CMPE 1
Hands-on Computer Engineering

LAB 4
SPRING 2009
Last week:
Bread Board

each column connected

each row connected
each row connected
Resistor Color Code

- Black
- Brown
- Red
- Orange
- Yellow
- Green
- Blue
- Purple
- Grey
- White
- Gold
- Silver

162K Ohm ± 1%
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.
Computers in your Room

A. 0
B. 1-5
C. 6-10
D. 11-15
E. 16-20
F. 21+
Last Week’s Homework

- Transistors
  - A. Can we build computers from sand next week?
  - B. I sort of follow the electrons and holes.
  - C. Could you talk a little more about transistors?
This week: Logic!

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F(A,B)</th>
<th>Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>GND</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Vcc</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Vcc</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Vcc</td>
</tr>
</tbody>
</table>
This week: Gates!

A | B | F
---|---|---
0 | 0 | 0
0 | 1 | 0
1 | 0 | 0
1 | 1 | 1

A | B | F
---|---|---
0 | 0 | 1
0 | 1 | 1
1 | 0 | 1
1 | 1 | 0
What makes gates go: transistors as switches

<table>
<thead>
<tr>
<th>IN</th>
<th>P-type</th>
<th>N-type</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Closed</td>
<td>Open</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Open</td>
<td>Closed</td>
<td>0</td>
</tr>
</tbody>
</table>
What makes NAND gates go: transistors as switches

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</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>
Truth tables

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

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>out</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>

<table>
<thead>
<tr>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Simple logic functions can be expresses in a truth table form
- Why would we not want to use truth tables for large number of input functions?

To implement this function in logic we can use the basic gates AND, OR, and NOT

One method to do this is called “Sum of Products” another is “Product of Sums”

There are other methods, including way to reduce the size of the logic
## XOR Gate

<table>
<thead>
<tr>
<th></th>
<th></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>

1. Find the rows with the ‘1’ output
2. Write the product-form expression for the inputs in that row (0=inverted, 1=normal)
3. Combine the products in step 2 into a sum (OR the results of step 2)
Product of Sums

- Procedure:
  1. Find the rows with the ‘0’ output
  2. Write the sum-form expression for the inputs in that row (0=normal, 1=inverted)
  3. Combine the sums in step 2 into a product (AND the results of step 2)

- Note: we treat 0 and 1 reverse than for SoP
Can implement **ANY** truth table with AND, OR, NOT.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Sum of Products Method:

1. AND combinations that yield a "1" in the truth table.
2. OR the results of the AND gates.
3. Is not necessarily a minimal solution.
This week: Chips!

7400
Using a chip:
Connecting your chip to power and ground
Now you can connect the inputs to one of the gates to either power or ground.
Now you can connect the output to an LED so you can verify the truth table.
Other useful chip diagrams

7404(NOT)

7402(NOR)
Simple Electronics: Ring Oscillator

[Diagram of a ring oscillator]

10/21/2009
74HC04

Ground (Gnd)

Power
Resistor (Ohsms)

Capacitor (Farads)

4.7MΩ

0.1μF

27

CE1: Hands-on Computer Engineering

10/21/2009
Light Emitting Diode (LED)