CSE140: Components and Design Techniques for Digital Systems

Introduction

Prof. Tajana Simunic Rosing
Welcome to CSE 140!

• **Instructor:** Tajana Simunic Rosing
  • Email: tajana.teach@eng.ucsd.edu;
    – please put CSE140 in the subject line
  • Office Hours:
    – MF 2-3pm & by appointment; CSE 2118

• **TAs and Tutors**
  – Office hrs & contact info on cse140 website

• **Instructor’s Assistant:** James Martinez
  – Office: CSE 2225
  – Email: jtmartinez@eng.ucsd.edu

• **Discussion sessions:**
  – Fri 8:00p - 8:50p HSS 1330

• **Grades:** [http://ted.ucsd.edu](http://ted.ucsd.edu)

• **HW:** [https://www.gradescope.com/courses/33404/](https://www.gradescope.com/courses/33404/)

• **Announcements and online discussion:**
Class website

- [https://cseweb.ucsd.edu/classes/wi19/cse140-b/](https://cseweb.ucsd.edu/classes/wi19/cse140-b/)
  - Syllabus
  - TA/Tutor office hours and locations
  - Class policies
  - Course schedule:
    - required online ZyBook exercises prior to class
    - HWs & exams
Textbooks and Recommended Readings

• Required online book:
  – Online digital design by F. Vahid
    • Exercises to complete prior to class
    • Student access instructions:
      1. Sign up at ZyBooks.com using your TED email address
      2. Enter ZyBook code: UCSDCSE140Winter2019
      3. Click ‘Subscribe‘ The cost is $58.

• Recommended textbooks:
  – Digital Design by F. Vahid
  – Digital Design & Computer Arch.
    • by David & Sarah Harris
  – Contemporary Logic Design
    • by R. Katz & G. Borriello

• Lecture slides are derived from the slides designed for all three books
In Class We Will Use Clickers!

Vote on multiple choice questions in real time!

• Make sure to register yours with TED!
Grading

- **Standard grading scale:**
  - 90-100 = A-/A/A+;
  - 80-89.9=B-/B/B+,
  - 70-79.9=C-/C/C+,
  - 60-69.9=D
  - Less than 60=F.
  - Plusses and minuses given at the instructor's discretion.

- **Grade components:**
  - **Class participation** 5%
    - To get the full participation grade each session you have to respond to at least 75% of iClicker questions in class
  - **ZyBook questions** 5% (grades start with Chap 2)
    - Due before class, % completed correctly counts toward your grade
  - **Homework** 10%  (usually 4 problems, 2 graded)
  - **Exam #1** 20%
  - **Exam #2** 25%
  - **Exam #3** 35%
  - **Bonus Take Home Final:** 1% extra
Some Class Policies

• Academic Honesty
  – Studying together in groups is **encouraged**
  – Turned-in work must be **completely your own**.
  – Both “giver” and “receiver” are equally culpable
  – Cheating on HW/ exams: **F in the course**.
  – Any instance of cheating will be referred to Academic Integrity Office

• Late:
  – No makeup for missed exams or zybook; exceptions must be documented
    • e.g. proof of the death in the family, a letter from a doctor justifying why you had
      to miss an exam and a permission to speak directly to your doctor
  – HWs are due at the beginning of the class
    • Up to 1 day late HWs get 10% lower grade
    • More than 1 day late get no points

• Regrades
  – Request online within 24hrs of grades being posted. TAs/Tutors will review
    and add/subtract points as appropriate. If you are still not satisfied, contact
    the professor.
Abstraction:
A way to simplify by hiding details from other layers
Why Study Digital Design?

Look “under the hood” of your processors
You become a better programmer when you understand hardware your code runs on

Nvidia Tegra 2 die photo
The Scope of CSE140

- We start with Boolean algebra \( Y = A \) and \( B \)
- We end with a hardware design of a simple CPU

- What’s next? CSE141 – more complex CPU architectures
What Does “Digital” Mean?

- **Analog signal**
  - Infinite possible values
  - Ex: voltage on a wire created by microphone

- **Digital signal**
  - Finite possible values
  - Ex: button pressed on a keypad

Possible values:
- An analog signal can have infinite possibilities: 1.00, 1.01, 2.0000009, ...
- A digital signal can have only a few values: 0, 1, 2, 3, or 4.

That’s it.
Encoding Numbers – Base 10 & 2

- Each position represents a quantity; symbol in position means how many of that quantity
  - Base ten (decimal)
    - Ten symbols: 0, 1, 2, ..., 8, and 9
    - More than 9 -- next position
      - So each position power of 10
    - Nothing special about base 10 -- used because we have 10 fingers
  - Base two (binary)
    - Two symbols: 0 and 1
    - More than 1 -- next position
      - So each position power of 2
Bases Sixteen & Eight

- **Base sixteen**
  - Used as compact way to write binary numbers
  - Basic digits: 0-9, A-F
  - Known as *hexadecimal*, or just *hex*
- **Base eight**
  - Basic digits: 0-7
  - Known as *octal*

<table>
<thead>
<tr>
<th>hex</th>
<th>binary</th>
<th>hex</th>
<th>binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>9</td>
<td>1001</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>A</td>
<td>1010</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>B</td>
<td>1011</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>C</td>
<td>1100</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>D</td>
<td>1101</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>E</td>
<td>1110</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>F</td>
<td>1111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>octal</th>
<th>binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
</tr>
<tr>
<td>2</td>
<td>010</td>
</tr>
<tr>
<td>3</td>
<td>011</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
</tr>
</tbody>
</table>
Combinational circuit building blocks: Switches & CMOS transistors

Prof. Tajana Simunic Rosing
Switches

- Electronic switches are the basis of binary digital circuits
- Electrical terminology
  - **Voltage**: Difference in electric potential between two points
    - Analogous to water pressure
  - **Current**: Flow of charged particles
    - Analogous to water flow; dQ/dt
  - **Resistance**: Tendency of wire to resist current flow
    - \( V = I \times R \)  *Ohm’s Law*
  - **Capacitance**: ratio of change in electric charge due to change in electric potential
    - \( I = C \times \frac{dV}{dt} = \frac{dQ}{dt} \)
CMOS Switches

- CMOS circuit
  - Consists of N and PMOS transistors
  - Both N and PMOS are similar to basic switches

Silicon -- not quite a conductor or insulator: *Semiconductor*
Transistor Circuit Design

- **nMOS:**
  - Turns on when gate is connected to 1
  - When turned on, nMOS passes zeros well, but not ones, so connect source to GND
  - nMOS forms a pull-down network

- **pMOS:**
  - Turns on when gate is connected to 0
  - When turned on, pMOS passes ones well, but not zeros, so connect source to $V_{DD}$
  - pMOS forms a pull-up network

- **Note:** Vahid’s textbook shows some circuits with pMOS connected to GND and nMOS to Vdd: this is NOT normally done in practice!
CMOS Switches

The following is true for CMOS switches:
A. nMOS turns on when gate is connected to logic 1
B. pMOS is an open switch when gate is connect to logic 1
C. All of the above
D. None of the above
Logic gates: CMOS NOT Gate

\[ Y = \overline{A} \]

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

<table>
<thead>
<tr>
<th>A</th>
<th>P1</th>
<th>N1</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ON</td>
<td>OFF</td>
<td>↑</td>
</tr>
<tr>
<td>1</td>
<td>OFF</td>
<td>ON</td>
<td>φ</td>
</tr>
</tbody>
</table>

\[ V_{DD} \]
\[ \text{P1} \]
\[ \text{N1} \]
\[ \text{GND} \]

\[ R \cdot 2C \]
CMOS Two Input NAND Gate

\[ Y = \overline{AB} \]

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

\[ Y = \overline{AB} \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>P1</th>
<th>P2</th>
<th>N1</th>
<th>N2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ Y = \frac{R_{\text{tot}} \cdot 2G_D}{2} \]

\[ R_{\text{tot}} = \frac{R}{2} \]

\[ R_{\text{tot}} = 2R \]
CMOS Two Input AND Gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>P1</th>
<th>P2</th>
<th>N1</th>
<th>N2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>φ</td>
</tr>
</tbody>
</table>
Common Logic Gates

- **AND** gate:
  
  \[ a \land b = a \cdot b \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \land b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- **OR** gate:
  
  \[ a \lor b = a + b \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \lor 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>

- **NOT** gate:
  
  \[ a' = \overline{a} \]

<table>
<thead>
<tr>
<th>a</th>
<th>a'</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **BUF** gate:
  
  \[ a^{''} = a \]

<table>
<thead>
<tr>
<th>a</th>
<th>a''</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- **NAND** gate:
  
  \[ a \land \overline{b} = \overline{a \lor b} \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \land \overline{b}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **NOR** gate:
  
  \[ a \lor \overline{b} = \overline{a \land b} \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \lor \overline{b}</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>

- **XOR** gate:
  
  \[ a \oplus b = \overline{a \land b} \land \overline{a \lor b} \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \oplus 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>0</td>
</tr>
</tbody>
</table>

- **XNOR** gate:
  
  \[ a \iff b = \overline{a \oplus b} \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a \iff b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Boolean algebra

- \( B = \{0, 1\} \)
- Variables represent 0 or 1 only
- Operators return 0 or 1 only
- Basic operators
  - Intersection: \( \cdot \) is logical **AND**: \( a \ \text{AND} \ b \) returns 1 only when \( a=1 \ \& \ b=1 \)
  - Union: \( + \) is logical **OR**: \( a \ \text{OR} \ b \) returns 1 if either \( a=1 \) or \( b=1 \) (or both)
  - Complement: \( ' \) is logical **NOT**: \( \text{NOT} \ a \) returns the opposite of \( a \)

**AND**

\[
\begin{array}{ccc}
 a & b & \text{AND} \\
 0 & 0 & 0 \\
 0 & 1 & 0 \\
 1 & 0 & 0 \\
 1 & 1 & 1 \\
\end{array}
\]

**OR**

\[
\begin{array}{ccc}
 a & b & \text{OR} \\
 0 & 0 & 0 \\
 0 & 1 & 1 \\
 1 & 0 & 1 \\
 1 & 1 & 1 \\
\end{array}
\]

**NOT**

\[
\begin{array}{ccc}
 a & \text{NOT} \\
 0 & 1 \\
 1 & 0 \\
\end{array}
\]

- Derived operators:

**NAND**

\[
\begin{array}{ccc}
 a & b & \text{NAND} \\
 0 & 0 & 1 \\
 0 & 1 & 1 \\
 1 & 0 & 1 \\
 1 & 1 & 0 \\
\end{array}
\]

**NOR**

\[
\begin{array}{ccc}
 a & b & \text{NOR} \\
 0 & 0 & 1 \\
 0 & 1 & 0 \\
 1 & 0 & 0 \\
 1 & 1 & 0 \\
\end{array}
\]

**XOR**

\[
\begin{array}{ccc}
 a & b & \text{XOR} \\
 0 & 0 & 0 \\
 0 & 1 & 1 \\
 1 & 0 & 1 \\
 1 & 1 & 0 \\
\end{array}
\]

**XNOR**

\[
\begin{array}{ccc}
 a & b & \text{XNOR} \\
 0 & 0 & 1 \\
 0 & 1 & 0 \\
 1 & 0 & 0 \\
 1 & 1 & 1 \\
\end{array}
\]

**BUFFER**

\[
\begin{array}{ccc}
 a & \text{BUF} \\
 0 & 0 \\
 1 & 1 \\
\end{array}
\]
X and Y are Boolean variables with X=1, Y=0
What is $X + X + Y$?
A. 0
B. 1
C. 2
D. None of the above
Universal Gate: NAND

Any logic function can be implemented using just **NAND** gates. Boolean algebra’s basic operators are AND, OR and NOT.

\[ A \cdot B = \overline{A + B} \]
Universal Gate: NOR

Any logic function can be implemented using just **NOR gates**. Boolean algebra needs AND, OR and NOT
# Boolean Axioms & Theorems

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Dual</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>A1’</td>
<td>Binary field</td>
</tr>
<tr>
<td>A2</td>
<td>A2’</td>
<td>NOT</td>
</tr>
<tr>
<td>A3</td>
<td>A3’</td>
<td>AND/OR</td>
</tr>
<tr>
<td>A4</td>
<td>A4’</td>
<td>AND/OR</td>
</tr>
<tr>
<td>A5</td>
<td>A5’</td>
<td>AND/OR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem</th>
<th>Dual</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>T1’</td>
<td>Identity</td>
</tr>
<tr>
<td>T2</td>
<td>T2’</td>
<td>Null Element</td>
</tr>
<tr>
<td>T3</td>
<td>T3’</td>
<td>Idempotency</td>
</tr>
<tr>
<td>T4</td>
<td></td>
<td>Involution</td>
</tr>
<tr>
<td>T5</td>
<td>T5’</td>
<td>Complements</td>
</tr>
</tbody>
</table>
### Boolean theorems of multiple variables

<table>
<thead>
<tr>
<th>Theorem</th>
<th>Dual</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>T6</td>
<td>$B \cdot C = C \cdot B$</td>
<td>T6’ $B + C = C + B$</td>
</tr>
<tr>
<td>T7</td>
<td>$(B \cdot C) \cdot D = B \cdot (C \cdot D)$</td>
<td>T7’ $(B + C) + D = B + (C + D)$</td>
</tr>
<tr>
<td>T8</td>
<td>$(B \cdot C) + B \cdot D = B \cdot (C + D)$</td>
<td>T8’ $(B + C) \cdot (B + D) = B + (C \cdot D)$</td>
</tr>
<tr>
<td>T9</td>
<td>$B \cdot (B + C) = B$</td>
<td>T9’ $B + (B \cdot C) = B$</td>
</tr>
<tr>
<td>T10</td>
<td>$(B \cdot C) + (B \cdot \overline{C}) = B$</td>
<td>T10’ $(B + C) \cdot (B + \overline{C}) = B$</td>
</tr>
<tr>
<td>T11</td>
<td>$(B \cdot C) + (\overline{B} \cdot D) + (C \cdot D)$</td>
<td>T11’ $(B + C) \cdot (\overline{B} + D) \cdot (C + D)$</td>
</tr>
<tr>
<td>T12</td>
<td>$B_0 \cdot B_1 \cdot B_2 ... = (\overline{B_0} + \overline{B_1} + \overline{B_2} ...)$</td>
<td>T12’ $B_0 + B_1 + B_2 ... = (\overline{B_0} \cdot \overline{B_1} \cdot \overline{B_2})$</td>
</tr>
</tbody>
</table>
Boolean Duality

• Derived by replacing $\cdot$ by $+$, $+$ by $\cdot$, 0 by 1, and 1 by 0 & leaving variables unchanged

$$X + Y + ... \iff X \cdot Y \cdot ...$$

• Generalized duality:

$$f (X_1,X_2,...,X_n,0,1,+,\cdot) \iff f(X_1,X_2,...,X_n,1,0,\cdot,+$$

• Any theorem that can be proven is also proven for its dual!  Note: this is NOT deMorgan’s Law
Covering Theorem Explained

- Covering Theorem: \( A^*(A+B) = A + A^*B = A \)

Venn Diagrams
Combining Theorem Explained

• Combining Theorem: \[ AB + AB' = (A+B) \cdot (A+B') = A \]
Proving theorems with Boolean Algebra

• Using the axioms of Boolean algebra:
  – e.g., prove the consensus theorem: \( X \cdot Y + X \cdot Y' = X \)
    
    | Distributivity | \( X \cdot Y + X \cdot Y' = X \cdot (Y + Y') \) |
    | Complementarity | \( X \cdot (Y + Y') = X \cdot (1) \) |
    | Identity       | \( X \cdot (1) = X \) |

  – e.g., prove the covering theorem: \( X + X \cdot Y = X \)
    
    | Identity       | \( X + X \cdot Y = X \cdot 1 + X \cdot Y \) |
    | Distributivity | \( X \cdot 1 + X \cdot Y = X \cdot (1 + Y) \) |
    | Identity       | \( X \cdot (1 + Y) = X \cdot (1) \) |
    | Identity       | \( X \cdot (1) = X \) |
Combinational circuit building blocks: Boolean Algebra Cont.

Prof. Tajana Simunic Rosing
Where we are....

• Last time:
  – Number representation, CMOS switches & basic gates
  – Started Boolean algebra

• Plan for today:
  – Boolean algebra, Logic representations

• Deadlines coming up:
  – HW assigned, due next Monday at the beginning of the class
    • 4 problems, 2 graded
  – Zybook due at the beginning of the class

• Updates
  – Check your participation score on TED
    • Grading started last Friday
Consensus Theorem of 3 Variables

- Consensus Theorem:

\[ AB + B'C + AC = AB + B'C \]
Proof of Consensus Theorem with Boolean Algebra

- Consensus Theorem:
  \[(X \cdot Y) + (Y \cdot Z) + (X' \cdot Z) = X \cdot Y + X' \cdot Z\]

\[
(X \cdot Y) + (Y \cdot Z) + (X' \cdot Z)
\]

identity
\[
(X \cdot Y) + (1) \cdot (Y \cdot Z) + (X' \cdot Z)
\]
complementarity
\[
(X \cdot Y) + (X' + X) \cdot (Y \cdot Z) + (X' \cdot Z)
\]
distributivity
\[
(X \cdot Y) + (X' \cdot Y \cdot Z) + (X \cdot Y \cdot Z) + (X' \cdot Z)
\]
commutativity
\[
(X \cdot Y) + (X \cdot Y \cdot Z) + (X' \cdot Y \cdot Z) + (X' \cdot Z)
\]
factoring
\[
(X \cdot Y) \cdot (1 + Z) + (X' \cdot Z) \cdot (1 + Y)
\]
null
\[
(X \cdot Y) \cdot (1) + (X' \cdot Z) \cdot (1)
\]
identity
\[
(X \cdot Y) + (X' \cdot Z) \checkmark
\]
Applying Boolean Algebra Theorems

Which of the following is CB+BA+C’A equal to?

A. AB+AC’
B. BC+AC’
C. AB+BC
D. None of the above
DeMorgan’s Theorem  
(Bubble Pushing)

- $Y = \overline{AB} = \overline{A} + \overline{B}$
- $Y = \overline{A + B} = \overline{A} \cdot \overline{B}$
Example of Transforming Circuits with Bubble Pushing

\[ 6 + 2 + 4 + 2 + 4 + 2 + 2 = 22 \]
Implement using only NORs

- Hint: DeMorgan’s Theorem

\[ F = X'Y + Z \]
Proving Theorems with Perfect Induction

- Using perfect induction = complete the truth table:
  - e.g., de Morgan’s:

\[(X + Y)' = X' \cdot Y'\]
NOR is equivalent to AND
with inputs complemented

\[(X \cdot Y)' = X' + Y'\]
NAND is equivalent to OR
with inputs complemented

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X'</th>
<th>Y'</th>
<th>((X + Y))'</th>
<th>(X' \cdot Y')</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X'</th>
<th>Y'</th>
<th>((X \cdot Y))'</th>
<th>(X' + Y')</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Summary

• What we reviewed thus far:
  – Number representations
  – Switches, logic gates
  – Boolean algebra
  – Using Boolean algebra to simplify Boolean equations
    • There is an easier way!

• What is next:
  – Combinational logic:
    • Minimization the easier way 😊
    • Designs of common combinational circuits
      – Adders, comparators, subtractors, multipliers, etc.
Note #1: Basic Needs Resources

• Are you eating properly? Do you have adequate access to nutritious food? Do you have stable housing? Are you homeless or couch surfing?

• If you or someone you know is suffering from food and/or housing insecurities, please note:
  • The **Triton Food Pantry** (in the old Student Center), [https://www.facebook.com/tritonfoodpantry/](https://www.facebook.com/tritonfoodpantry/), is free and anonymous, and includes produce.
  • Financial aid resources, the possibility of emergency grant funding, and off-campus housing referral resources are available.
  • CAPS and college deans can connect students to the above resources, as well as other community resources and support.
The Office for the Prevention of Harassment & Discrimination (OPHD) provides assistance to students, faculty, and staff regarding reports of bias, harassment, and discrimination. OPHD is the UC San Diego Title IX office. Title IX of the Education Amendments of 1972 is the federal law that prohibits sex discrimination in educational institutions that are recipients of federal funds. Jacobs School students have the right to an educational environment that is free from harassment and discrimination.

Students have options for reporting incidents of sexual violence and sexual harassment. Sexual violence includes sexual assault, dating violence, domestic violence, and stalking. Information about reporting options may be obtained at OPHD at (858) 534-8298, ophd@ucsd.edu or http://ophd.ucsd.edu. Students may receive confidential assistance at CARE at the Sexual Assault Resource Center at (858) 534-5793, sarc@ucsd.edu or http://care.ucsd.edu or Counseling and Psychological Services (CAPS) at (858) 534-3755 or http://caps.ucsd.edu.

Students may feel more comfortable discussing their particular concern with a trusted employee. This may be a Jacobs School student affairs staff member, a department Chair, a faculty member or other University official. These individuals have an obligation to report incidents of sexual violence and sexual harassment to OPHD. This does not necessarily mean that a formal complaint will be filed.

If you find yourself in an uncomfortable situation, ask for help. The Jacobs School is committed to upholding University policies regarding nondiscrimination, sexual violence and sexual harassment.