Lecture 10
Digital Logic Design
-- Basics
Logic Design

- From now, we’ll study how a modern processor is built starting with basic logic elements as building blocks.

- Why study logic design?
  - Understand what processors can do fast and what they can’t do fast (avoid slow things if you want your code to run fast!)
  - Background for more detailed hardware courses
Logic Gates

- Basic building blocks are logic gates.
  - In the beginning, did ad hoc designs, and then saw patterns repeated, gave names
  - Can build gates with transistors and resistors

- Then found theoretical basis for design
  - Can represent and reason about gates with truth tables and Boolean algebra
  - Assume know truth tables and Boolean algebra from a math or circuits course.
  - Section B.2 in the textbook has a review
Gates

A transistor inverter or A NOT gate

A NAND gate

A NOR gate
Gates and Boolean Algebra

Five basic gates

 Truth Table
Digit Display

Figure 2.4  Binary digit-display combinational system.
Building a Circuit

\[ C_{out} = \overline{A}BC_{in} + AB\overline{C}_{in} + A\overline{B}C_{in} + ABC_{in} \]

- Use And/Or gates to implement this expression
- How many gates do we use?
- Can we use less

\[ = BC_{in} + AC_{in} + AB(\overline{C}_{in} + C_{in}) \]
\[ = BC_{in} + AC_{in} + AB(1) \]
\[ = BC_{in} + AC_{in} + AB \]
Boolean Algebra 101
How to simplify Boolean expression?
Boolean Algebra (1)

- Using just ‘0’ and ‘1’ s (after George Boole)

- Truth Table
  - A Boolean function with n variables has only $2^n$ possible input combinations
  - A table with $2^n$ rows
Boolean Algebra (2)

- Algebra representations

(a) NOT

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

(b) NAND

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

(c) NOR

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

(d) AND

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

(e) OR

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>X</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>
Boolean Algebra (3)

Another example with 3 inputs

![Boolean Algebra Diagram]

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

(a)
Boolean Algebra (4)

- All Boolean functions can be implemented by logic gates (AND, OR, NOT) and vice versa.

- Boolean operations
  - AND: \( XY, \ X \land Y, X \& Y \)
  - OR: \( X+Y, X \lor Y, X | Y \)
  - NOT: \( \overline{X}, \ X', /X, \ \backslash X \)

- Order: NOT, AND, OR

\[
\overline{A} \cdot B + C = ((\overline{A}) \cdot B) + C
\]
\[
\overline{A} + B \cdot C = (\overline{A}) + (B \cdot C)
\]
Basics

- P1: \( X = 0 \) or \( X = 1 \)
- P2: \( 0.0 = 0 \)
- P3: \( 1 + 1 = 1 \)
- P4: \( 0 + 0 = 0 \)
- P5: \( 1.1 = 1 \)
- P6: \( 1.0 = 0.1 = 0 \)
- P7: \( 1 + 0 = 0 + 1 = 1 \)
Boolean Laws

- **T1 : Commutative Law**
  - $A + B = B + A$
  - $A \times B = B \times A$

- **T2 : Associate Law**
  - $(A + B) + C = A + (B + C)$
  - $(A \times B) \times C = A \times (B \times C)$
Boolean Laws

- **T3 : Distributive Law**
  - $A (B + C) = A B + A C$
  - $A + (B C) = (A + B) (A + C)$

- **T4 : Identity Law**
  - $A + A = A$
  - $A A = A$
Laws

- **T6: Redundancy Law**
  - $A + A B = A$
  - $A (A + B) = A$

- **T7**
  - $0 + A = A$
  - $0 A = 0$

- **T8**
  - $1 + A = 1$
  - $1 A = A$
## Other Laws

<table>
<thead>
<tr>
<th>Name</th>
<th>AND form</th>
<th>OR form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity law</td>
<td>$1A = A$</td>
<td>$0 + A = A$</td>
</tr>
<tr>
<td>Null law</td>
<td>$0A = 0$</td>
<td>$1 + A = 1$</td>
</tr>
<tr>
<td>Idempotent law</td>
<td>$AA = A$</td>
<td>$A + A = A$</td>
</tr>
<tr>
<td>Inverse law</td>
<td>$A\bar{A} = 0$</td>
<td>$A + \bar{A} = 1$</td>
</tr>
<tr>
<td>Commutative law</td>
<td>$AB = BA$</td>
<td>$A + B = B + A$</td>
</tr>
<tr>
<td>Associative law</td>
<td>$(AB)C = A(BC)$</td>
<td>$(A + B) + C = A + (B + C)$</td>
</tr>
<tr>
<td>Distributive law</td>
<td>$A + BC = (A + B)(A + C)$</td>
<td>$A(B + C) = AB + AC$</td>
</tr>
<tr>
<td>Absorption law</td>
<td>$A(A + B) = A$</td>
<td>$A + AB = A$</td>
</tr>
<tr>
<td>De Morgan's law</td>
<td>$\bar{A}B = \bar{A} + \bar{B}$</td>
<td>$\bar{A} + B = \bar{A}\bar{B}$</td>
</tr>
</tbody>
</table>
Example

\[ C_{\text{out}} = \overline{A}BC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = ABC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = ABC_{\text{in}} + ABC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = (\overline{A} + A)BC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = (1)BC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = BC_{\text{in}} + A\overline{B}C_{\text{in}} + AB\overline{C}_{\text{in}} + ABC_{\text{in}} \]

\[ = BC_{\text{in}} + A\overline{B}C_{\text{in}} + AB(\overline{C}_{\text{in}} + C_{\text{in}}) \]

\[ = BC_{\text{in}} + AC_{\text{in}} + AB(1) \]

\[ = BC_{\text{in}} + AC_{\text{in}} + AB \]

Idempotent theorem + commutative law

distributive law

complementarity law

identity law
Question 1

Simplify the Boolean expression

\((A+B+C)(D+E)' + (A+B+C)(D+E)\) and choose the best answer.

- A + B + C
- D + E
- A'B'C'
- D'E'
- None of the above
Given the function $F(X,Y,Z) = XZ + Z(X' + XY)$, the equivalent most simplified Boolean representation for $F$ is:

- $Z + YZ$
- $Z + XYZ$
- $XZ$
- $X + YZ$
- None of the above
Question 3

An equivalent representation for the Boolean expression $A' + 1$ is

- $A$
- $A'$
- $1$
- $0$
Question 4

Simplification of the Boolean expression

\[ AB + ABC + ABCD + ABCDE + ABCDEF \]
yields which of the following results?

- ABCDEF
- AB
- AB + CD + EF
- A + B + C + D + E + F
- A + B(C+D(E+F))
Question 5

Given that $F = A'B' + C' + D' + E'$, which of the following represent the only correct expression for $F'$?

- $F' = A + B + C + D + E$
- $F' = ABCDE$
- $F' = AB(C + D + E)$
- $F' = AB + C' + D' + E'$
- $F' = (A + B)CDE$
Deriving Complement of a Boolean Expression

- DeMorgan’s LAW: The complement of a Boolean expression is formed by replacing all literals by their compliments; ANDs become ORs, ORs become ANDs

\[
\overline{X + Y} = \overline{X} \cdot \overline{Y} \quad \text{and} \quad \overline{X \cdot Y} = \overline{X} + \overline{Y}
\]
Using DeMorgan’s Law

\[ \bar{X} + Y = \bar{X} \cdot \bar{Y} \]
\[ X \cdot Y = \bar{X} + \bar{Y} \]

\[ \bar{X} + Y \cdot Z = ? \]
Question 5

Given that \( F = A'B' + C' + D' + E' \), which of the following represent the only correct expression for \( F' \)?

- \( F' = A + B + C + D + E \)
- \( F' = ABCDE \)
- \( F' = AB(C + D + E) \)
- \( F' = AB + C' + D' + E' \)
- \( F' = (A + B)CDE \)
Question 6

Given the function \( F(A,B,X,Y) = AB + X'Y \), the most simplified Boolean representation for \( F' \) is

- \((AB)' + (X'Y)\)
- \(A'B' + XY'\)
- \((A'+ B')(X + Y')\)
- \((AB + X'Y)'\)
- \((AB)'(X'Y)'\)
Question 7

Given the function $F(X,Y,Z) = XZ + Z(X' + XY)$, the equivalent, most simplified Boolean representation for $F$ is

- $Z + YZ$
- $Z + XYZ$
- $XZ$
- $Z$
- none of the above
Question 8

Simplification of the Boolean expression AB + A (BC)' yields which of following results?

- A
- BC
- B
- AB
- (BC)'