Lecture 9: Universal Gates

CSE 140: Components and Design Techniques for Digital Systems

Diba Mirza
Dept. of Computer Science and Engineering
University of California, San Diego
Announcements

• Today is the last day to register clickers on TED
• Midterm - All material until HW2
• K-map wrap up
  – The decision tree method provides all solutions with the minimum number of terms
  – If these solutions differ in the total number of literals, choose the ones that have the min. number of literals as the “possible solutions”

We will use K-maps later in the course, so don’t forget about them
Combinational Logic: Other Types of Gates

- Universal Set of Gates
- Other Types of Gates
  1) XOR
  2) NAND / NOR
  3) Block Diagram Transfers
Universal Set of Gates: Motivation

- **AND, OR, NOT**: Logic gates related to reasoning from Aristotle (384-322 BC)
- **NAND, NOR**: Inverted AND, Inverted OR gates. VLSI technologies. All gates are inverted.
- **Multiplexer + input table**: FPGA technology. Table based logic for programmability.

In the future, we may have new sets of gates due to new technologies.
Given a set of gates, can this set of gates cover all possible switching functions?
Universal Set

Universal Set: A set of gates such that every switching function can be implemented with the gates in this set.

Examples

{AND, OR, NOT}

{AND, NOT}

{OR, NOT}
Universal Set

Universal set is a powerful concept to identify the coverage of a set of gates afforded by a given technology.

If the set of gates can implement AND, OR, and NOT gates, the set is universal.
Universal Set

Universal Set: A set of gates such that every Boolean function can be implemented with the gates in this set.

Examples

\{AND, OR, NOT\}

\{AND, NOT\} OR can be implemented with AND & NOT gates

\[ a + b = (a' \cdot b')' = \overline{\overline{a+b}} = \overline{\overline{\overline{a}} \cdot \overline{\overline{b}}} \]
Universal Set

Universal Set: A set of gates such that every Boolean function can be implemented with the gates in this set.

Examples

\{\text{AND, OR, NOT}\}

\{\text{AND, NOT}\} \text{ OR can be implemented with AND \& NOT gates} \quad a+b = (a' \ b')'

\{\text{OR, NOT}\} \text{ AND can be implemented with OR \& NOT gates} \quad a \cdot b = (\bar{a} + \bar{b})
Universal Set

Universal Set: A set of gates such that every Boolean function can be implemented with the gates in this set.

Examples

\{\text{AND, OR, NOT}\}

\{\text{AND, NOT}\} \text{ OR can be implemented with AND \& NOT gates} \quad a+b = (a' b')'

\{\text{OR, NOT}\} \text{ AND can be implemented with OR \& NOT gates} \quad ab = (a'+b')'

\{\text{XOR}\} \text{ is not universal}

\{\text{XOR, AND}\} \text{ is universal}
iClicker

Is the set \{AND, OR\} (but no NOT gate) universal?
A. Yes
B. No
iClicker

Is the set \{f(x,y)=x.y'\} universal?
A. Yes
B. No

To show universal:
- Show \( \neg x\neg y \) and \( \neg x\vee y \) or not
- \( \neg x \)
- \( x \vee y \)
- De Morgan's
- Access to input 0, 1 signal
- OR \( \Leftrightarrow \) AND & NOT?
Is NAND universal?

1. Implementing NOT using NAND

![Diagram of NOT using NAND]

2. Implementing AND using NAND

![Diagram of AND using NAND]

3. Implementing OR using NAND

![Diagram of OR using NAND]
Is NOR universal?

1. Implementing NOT using NOR

![Diagram of NOT using NOR]

2. Implementing OR using NOR

![Diagram of OR using NOR]

3. Implementing AND using NOR
Is \{\text{XOR, AND}\} universal?

1. Implementing NOT using XOR

\[ x \oplus y = \overline{x}y + xy \]

\[ x \oplus 1 = \]

\[ X \oplus 1 = X.1' + X'.1 = X' \text{ if constant "1" is available.} \]

2. Implementing OR using XOR and AND

Same as implementing OR using AND and NOT except NOT is implemented using XOR as shown above
Other Types of Gates: Properties and Usage

1) XOR  \( X \oplus Y = XY' + X'Y \)

It is a parity function (examples)
Useful for testing because the flipping of a single input changes the output.

<table>
<thead>
<tr>
<th>id</th>
<th>x</th>
<th>y</th>
<th>( x \oplus y )</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \begin{array}{c|c|c}
  x=0 & x=1 \\
  \hline
  y=0 & 0 & 1 \\
  y=1 & 1 & 0 \\
\end{array} \]
Other Types of Gates: Properties and Usage

1) XOR  \( X \oplus Y = XY' + X'Y \)

(a) Commutative  \( X \oplus Y = Y \oplus X \)
(b) Associative  \( (X \oplus Y) \oplus Z = X \oplus (Y \oplus Z) \)
(c) 1 \( \oplus X = X', \quad 0 \oplus X = 0X' + 0'X = X \)
(d) X \( \oplus X = 0, \quad X \oplus X' = 1 \)
e) If \( ab = 0 \), then \( a \oplus b = a + b \)

Proof: If \( ab = 0 \), then
\[
a = a (b+b') = ab+ab' = ab'
b = b (a + a') = ba + ba' = a' b
\]

\( a+b = ab' + a' b = a \oplus b \)

f) \( f(x,y)=x \oplus xy' \oplus x'y \oplus (x + y) \oplus x \) = ?

(Priority of operations: AND, \( \oplus \), OR)

Hint: We apply Shannon’s Expansion.
\[
f(x,y) = \bar{x} f(0,y) + x f(1,y)
\]
Simplify using Shannon’s expansion

Simplify the function
\[ f(X,Y) = X \oplus XY' \oplus X' \oplus (X+Y) \oplus X \]

Case \( X = 1 \): 
\[ f(1, Y) = 1 \oplus Y' \oplus 0 \oplus 1 \oplus 1 = Y \]
Case \( X = 0 \): 
\[ f(0, Y) = 0 \oplus 0 \oplus Y \oplus Y \oplus 0 = 0 \]

Thus, using Shannon’s expansion, we have 
\[ f(X, Y) = Xf(1,Y)+X'f(0,Y) = XY \]

Use identities from previous slides:
\[ Y \oplus 1 = Y \]
\[ Y \oplus 0 = Y \]
\[ 0 \oplus 0 = 0 \]
\[ 0 \oplus 1 = 1 \]
\[ Y \oplus Y = 0 \]
\[ Y \oplus \overline{Y} = 1 \]
XOR gates

iClicker: \( a + (b \oplus c) \) = \((a+b) \oplus (a+c)\)?

A. Yes

B. No

To disprove, set \( a = 1 \) & show that the equality doesn't hold.
2) NAND, NOR gates

NAND, NOR gates are not associative

Let \( a \mid b = (ab)' \)

\[ (a \mid b) \mid c \neq a \mid (b \mid c) \]
3) Block Diagram Transformation

a) Reduce # of inputs.
b. DeMorgan’s Law

\[(a+b)' = a' \cdot b'\]

\[(ab)' = a' + b'\]
c. Sum of Products (Using only NAND gates)

Sum of Products (We need more gates if we using NOR gates)
d. Product of Sums (NOR gates only)

We will need more gates if we used only NAND gates.
NAND, NOR gates

Remark:
Two level NAND gates: Sum of Products
Two level NOR gates: Product of Sums
Part II. Sequential Networks

Memory / Timesteps

Flip flops
Specification
Implementation
Reading

[Harris] Chapter 3, 3.1, 3.2