CS 211: Computer Architecture
Digital Logic

Topics:
- Converting truth tables to expressions
- Karnaugh maps
Converting Truth Table to Boolean Expression

Given a circuit, isolate the rows in which the output of the circuit should be true
Converting Truth Table to Boolean Expression

Given a circuit, isolate those rows in which the output of the circuit should be true.

A product term that contains exactly one instance of every variable is called a minterm.
Converting Truth Table to Boolean Expression

Given the expressions for each row, build a larger Boolean expression for the entire table.

- This is a sum-of-products (SOP) form.
Converting Truth Table to Boolean Expression

Finally build the circuit.

- **Problem:** SOP forms are often not minimal.
- **Solution:** Make it minimal. We’ll go over two ways.
First approach: algebraic

\[
\begin{align*}
\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC \\
BC(\bar{A} + A) + A\bar{B}C + AB\bar{C} \\
BC(1) + A\bar{B}C + AB\bar{C} \\
BC + A\bar{B}C + AB\bar{C} \\
B(C + A\bar{C}) + A\bar{B}C
\end{align*}
\]

\[
\begin{align*}
B(C + A) + A\bar{B}C \\
BC + AB + A\bar{B}C \\
BC + A(B + \bar{B}C) \\
BC + A(B + C) \\
BC + AB + AC
\end{align*}
\]
The Result

Output = $\overline{ABC} + \overline{ABC} + \overline{ABC} + ABC$

Output = $AB + BC + AC$
Karnaugh Maps or K-Maps

K-maps are a graphical technique to view minterms and how they relate.

The “map” is a diagram made up of squares, with each square representing a single minterm.

Minterms resulting in a “1” are marked as “1”, all others are marked “0”
2 Variable K-Map
2 Variable K-Map

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Output</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>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
2 Variable K-Map

\[ \begin{array}{cc}
\overline{B} & B \\
\hline
\overline{A} & A \\
A & A \\
\end{array} \]

A | B | Output
---|---|---
0  | 0 | 0  
0  | 1 | 1  
1  | 0 | 0  
1  | 1 | 1  

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

Finding Commonality

\[
\begin{array}{c|c|c}
A & B & 0 \\
\hline
0 & 1 & 1 \\
1 & 0 & 1 \\
\end{array}
\]

Out = \overline{B}

\[
\begin{array}{c|c|c}
\overline{A} & \overline{B} & B \\
\overline{A} & A & \overline{B} \\
\end{array}
\]

Out = \overline{A}
Finding the “best” solution

Grouping become simplified products.
Both are “correct”. “A+B” is preferred.
Simplify Example
Simplify Example
3 Variable K-Maps

- Note in higher maps, several variables occupy a given axis
- The sequence of 1s and 0s follow a **Gray Code Sequence**.
3 Variable K-Maps

Out = \overline{AB} \overline{C} + \overline{ABC}

Out = \overline{AB}

Out = 00 01 11 10

A

BC

A

B C B C B C

0 1 1 1 1 1 1 0

0 1

1
3 Variable K-Maps

![K-Map Diagram]

\[
\text{Out} = \overline{A} \overline{B} \overline{C} + \overline{A} \overline{B} \overline{C} + \overline{A} \overline{B} \overline{C} + \overline{A} \overline{B} \overline{C}
\]

\[
\text{Out} = \overline{A}
\]
3 Variable K-Maps

\[ \text{Out} = \overline{ABC} + \overline{A} BC + A \overline{BC} + ABC \]

\[ \text{Out} = C \]
3 Variable K-Maps

\[
\text{Out} = \overline{A}\overline{B}\overline{C} + \overline{A}BC + \overline{A}BC + \overline{A}B\overline{C} + ABC + AB\overline{C}
\]

\[
\text{Out} = \overline{A} + B
\]
3 Variable K-Maps

Out = \overline{ABC} + \overline{ABC} + \overline{ABC} + ABC

Out = \overline{C}
Back to our earlier example.....

The K-map and the algebraic produce the same result.
Up... up... and let's keep going

\[ \text{Out} = \overline{ABCD} + \overline{ABC}D + \overline{ABC}D + \overline{A}BCD \]

\[ \text{Out} = \overline{BD} \]
Few more examples

\[
\text{Out} = \overline{ABCD} + \overline{ABCD} + \overline{ABCD} + \overline{ABC}D
\]

\[
\text{Out} = \overline{BCD} + \overline{ABD} + \overline{ABC}D
\]
Few more examples

\[
\text{Out} = \overline{A \overline{B} \overline{C}} \overline{D} + \overline{A} \overline{B} \overline{C} \overline{D} + \overline{A} \overline{B} \overline{C} D + \overline{A} B \overline{C} D + \overline{A} B C D + A B \overline{C} D + A B C D
\]

\[
\text{Out} = \overline{A C} + \overline{A D} + B \overline{C} + B D
\]
Don’t Care Conditions

• Let \( F = AB + \overline{AB} \)
• Suppose we know that a disallowed input combo is \( A=1, B=0 \)
• Can we replace \( F \) with a simpler function \( G \) whose output matches for all inputs we do care about?
• Let \( H \) be the function with Don’t-care conditions for obsolete inputs

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F</th>
<th>H</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</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>X</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ G = AB + \overline{B} \]

• Both \( F \) & \( G \) are appropriate functions for \( H \)
• \( G \) can substitute for \( F \) for valid input combinations
Don’t Cares can Greatly Simplify Circuits

Sometimes “don’t cares” greatly simplify circuitry

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

\[ \overline{ABC}D + \overline{AB}C\overline{D} + ABCD + AB\overline{C}\overline{D} \text{ vs. } \overline{A} + C \]
# Formal Definition of Minterms

**e.g., Minterms for 3 variables A, B, C**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>minterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>m0 $\overline{A}\overline{B}\overline{C}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>m1 $\overline{A}BC$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>m2 $\overline{A}B\overline{C}$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>m3 $\overline{A}BC$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>m4 $A\overline{B}\overline{C}$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>m5 $A\overline{B}C$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>m6 $ABC$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>m7 $ABC$</td>
</tr>
</tbody>
</table>

- A product term in which all variables appear once, either complemented or uncomplemented (i.e., an entry in the truth table).

- Each minterm evaluates to 1 for exactly one variable assignment, 0 for all others.

- Denoted by $mX$ where $X$ corresponds to the variable assignment for which $mX = 1$. 
Minterm Example

(variables appear once in each minterm)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
<th>F’</th>
<th>minterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>m0: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>m1: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>m2: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>m3: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>m4: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>m5: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>m6: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>m7: ( \overline{A} \overline{B} \overline{C} )</td>
</tr>
</tbody>
</table>

\[
F = \overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC} = m0 + m1 + m2 + m4 + m5 = \Sigma m(0,1,2,4,5)
\]

\[
\overline{F} = AB \overline{C} + A \overline{B} \overline{C} + ABC = m3 + m6 + m7 = \Sigma m(3,6,7)
\]
Minterm Example

Circuits may be specified in the form $\sum m(0, 2, 4, 5, 6, 7, 13, 15)$, where the indices correspond to positions in a Karnaugh map:

<table>
<thead>
<tr>
<th></th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>
Minterm Example

Circuits may be specified in the form $\sum \text{m}(0, 2, 4, 5, 6, 7, 13, 15)$, where the indices correspond to positions in a Karnaugh map:

<table>
<thead>
<tr>
<th>AB</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>1 0 0 1</td>
</tr>
<tr>
<td>01</td>
<td>1 1 1 1</td>
</tr>
<tr>
<td>11</td>
<td>0 1 1 0</td>
</tr>
<tr>
<td>10</td>
<td>0 0 0 0</td>
</tr>
</tbody>
</table>
## Formal Definition of Maxterms

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>maxterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>M0 : (A+B+C)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>M1 : (A+B+\bar{C})</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>M2 : (\bar{A}+B+C)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>M3 : (\bar{A}+\bar{B}+\bar{C})</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>M4 : (\bar{A}+B+C)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>M5 : (\bar{A}+B+\bar{C})</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>M6 : (\bar{A}+\bar{B}+C)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>M7 : (\bar{A}+\bar{B}+\bar{C})</td>
</tr>
</tbody>
</table>

- A sum term in which all variables appear once, either complemented or uncomplemented.

- Each maxterm evaluates to 0 for exactly one variable assignment, 1 for all others.

- Denoted by \(M_X\) where \(X\) corresponds to the variable assignment for which \(M_X = 0\).
## Maxterm Example

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>maxterm</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>A+B+C</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>A+B+\bar{C}</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>A+\bar{B}+C</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>A+\bar{B}+\bar{C}</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\bar{A}+B+C</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>\bar{A}+B+\bar{C}</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>\bar{A}+\bar{B}+C</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>\bar{A}+\bar{B}+\bar{C}</td>
<td>0</td>
</tr>
</tbody>
</table>

F = (A+B+C) (A+B+\bar{C}) (\bar{A}+B+C)

= (M3) (M6) (M7)

= \prod M(3,6,7)
Maxterm Example

Then we can find the usual sum of products:

\[ F = BD + \overline{A}\overline{D} \]
Maxterm Example

Or the product of sums:

\[ F = (\overline{A} + D)(B + \overline{D}) \]
Converting Between Canonical Forms

DeMorgans: same terms
Product of Sums Example

\[ \overline{F} = YZ + XZ + YX \]

DeMorgan’s

\[ F = (\overline{Y} + \overline{Z})(\overline{Z} + \overline{X})(\overline{Y} + \overline{X}) \]
K-maps and Implicants

- **Implicant**: A product term, which, viewed in a K-Map is a $2^i \times 2^j$ size "rectangle" (possibly wrapping around) where $i=0,1,2$, $j=0,1,2$. 

![K-map diagram]
Implicants

- **Implicant**: a product term, which, viewed in a K-Map is a $2^i \times 2^j$ size “rectangle” (possibly wrapping around) where $i = 0, 1, 2$, $j = 0, 1, 2$.

Note: bigger rectangles = fewer literals
More Implicant Terminology

Implicant: product term, which when viewed in a K-map, is a rectangle of 1s

Prime implicant: an implicant not contained in another implicant

Essential prime implicant: a prime implicant that is the only prime implicant to cover some minterm
Example

- List all of the prime implicants for this function
- Is any of them an essential prime implicant?
- What is a simplified expression for this function?
Example

- Step 1: Identify all PIs and essential PIs
- Step 2: Include all Essential PIs in the circuit (Why?)
- Step 3: If any 1-valued minterms are uncovered by EPIs, choose PIs that are “big” and do a good job covering
- Selection Rule: a heuristic for usually choosing “good” PIs: choose the PIs that minimize overlap with one another and with EPIs

Red bounds are EPIs (solo-covered minterm shown in red)

Also need (purple or blue) and (yellow or green)

All blue PIs or all green PIs cover

No EPIs!
Design Example

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vx</td>
<td>Wx</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
</tbody>
</table>

e.g., what outputs “lights up” when input V=4?
Design Example

For what values does output f “light up” for?
Design Example

\[ f = W + \bar{Y}\bar{Z} + X\bar{Z} + \bar{X}\bar{Y} = W + (X+\bar{Y})\bar{Z} + \bar{X}\bar{Y} \]