ECE 697CE
Foundations of Computer Engineering

Lesson 3

Boolean Switching Functions (Continued)

Rationale

- Minimizing switching functions helps overcome limitations of switching algebra
- Minimization also creates more effective expressions

Objectives

- Apply principles of minimization to simplify switching functions
- Compare the properties of unate, monotonic, and symmetric functions

Poll: Prior Knowledge

The following is an example of what principle of Boolean Switching Functions?

$$[\ [f(x_1,\ x_2 \ldots .\, x_n, 0, 1, +, .)]'\ ]' = f(x_1,\ x_2 \ldots .\, x_n, 0, 1, +, .)$$

1. **Combinational Switching Logic**
2. **De Morgan’s Theorem**
3. **Law of Complementation**
4. **Consensus Theorem**

Prior Knowledge

- Let's review from Lesson 1: Boolean Switching Functions
  - Number representation
  - Switching logic & algebra
  - Simplification of expressions
  - De Morgan's Theorems
  - Threshold logic

Orchestrated Discussion (Hand Raise): Critical Thinking Exercise

- **Discuss two questions regarding Lesson 2.**

Minimization Example

- Example: $f(w,x,y,z) = \sum(1,5,6,7,11,12,13,15)$
- One irredundant form:

$$f = wxy' + wyz + w'xy + w'y'z$$

- Dotted box: xz is redundant

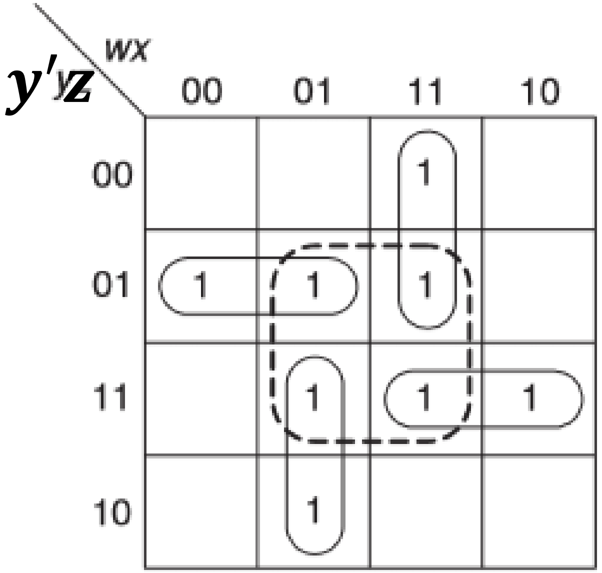


Fig : Map for $f = wxy' + wyz + w'xy + w'y'z$

Group Discussion and Report Back (Pen): Minimization

- **Minimize the following problem using the Karnaugh maps method.**

$$Z = f(A,B,C) = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}B + AB\bar{C} + AC$$

Minimization

- **Minimize the following problem using the Karnaugh maps method.**

$$Z = f(A,B,C) = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}B + AB\bar{C} + AC$$

C \ AB	00	01	11	10
0	1	1	1	
1		1	1	1

UMASS AMHERST

Don't Care Combinations

- Don't care combination ∅
  - Combination for which the value of the function is not specified
  - Either input combinations may be invalid
  - Or precise output value is of no importance
- A function with k don't cares corresponds to a class of 2k distinct functions
- Choose the function with the minimal representation
  - Assign 1 to some don't cares and 0 to others in order to increase the size of the selected cubes
  - No cube containing only don't care cells may be formed

Code Converter for Don't Care Combinations

- **Example: Code converter from BCD to excess-3**
- **Combinations 10 through 15 are don't cares**

Decimal number	BCD inputs w	x	y	z	Excess-3 outputs f_4	f_3	f_2	f_1
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Fig : Truth table for code converter

- **Output functions**

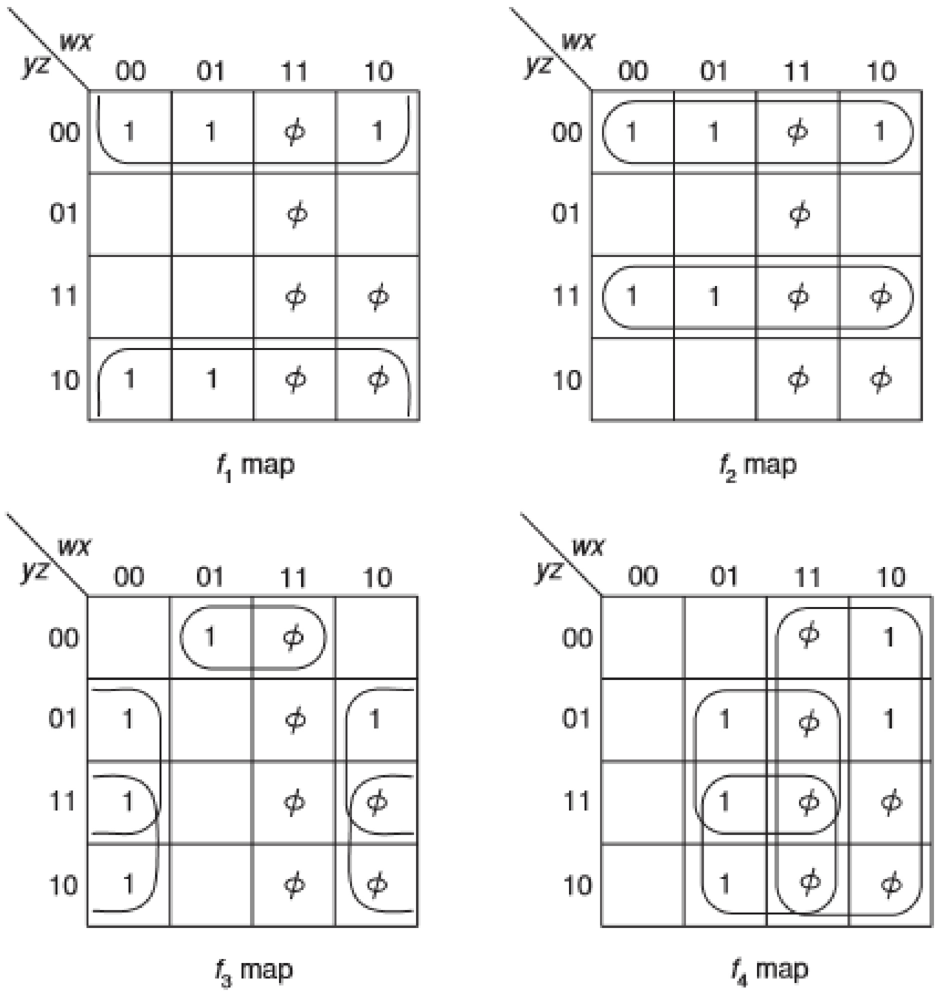
$$f_1 = \sum(0,2,4,6,8) + \sum_{\emptyset}(10,11,12,13,14,15) \quad f_3 = \sum(1,2,3,4,9) + \sum_{\emptyset}(10,11,12,13,14,15)$$

$$f_2 = \sum(0,3,4,7,8) + \sum_{\emptyset}(10,11,12,13,14,15) \quad f_4 = \sum(5,6,7,8,9) + \sum_{\emptyset}(10,11,12,13,14,15)$$

Code Converter for Don’t Care Combinations

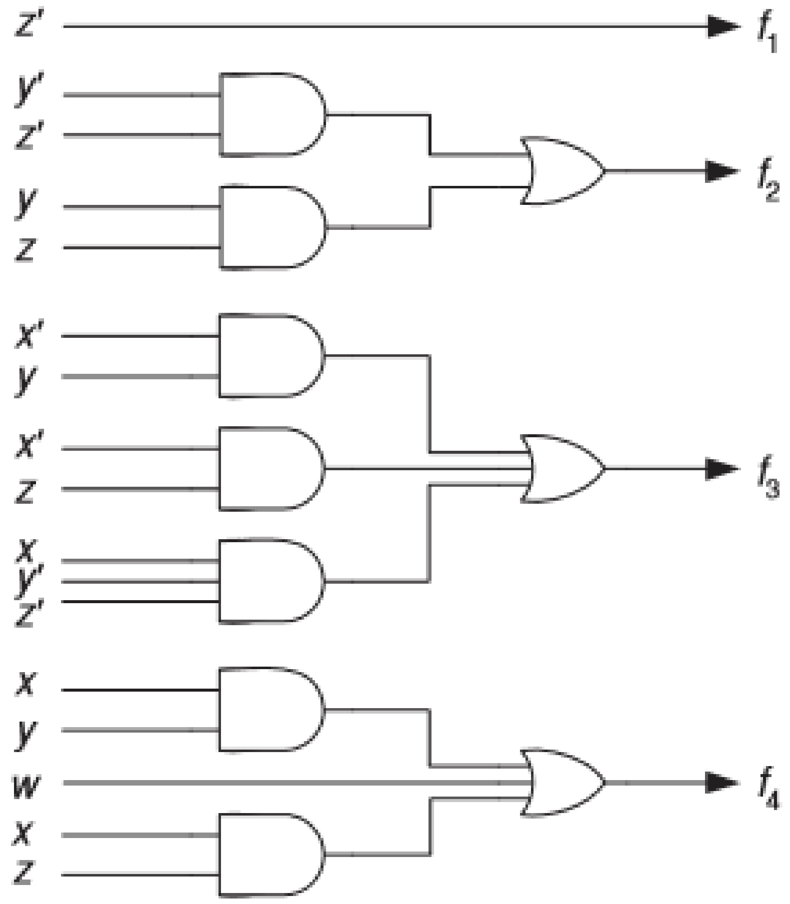
- **Minimal functions from Maps**

$$f_1 = z'$$
$$f_2 = y'z' + yz$$
$$f_3 = x'y + x'z + xy'z'$$
$$f_4 = w + xy + xz$$



Logical Network for Code Converter

- **Two-level AND-OR realization**



Unate Functions

Consider a function $\mathbf{f}(\mathbf{x_1}, \mathbf{x_2}, \dots \mathbf{x_n})$

If $\mathbf{f}$ is either positive or negative in $\mathbf{x_j}$, it is said to be UNATE in $\mathbf{x_j}$

- Examples:
  1. $f(x_1, x_2, \dots x_n) = x_1{}' x_2 + x_2{}' x_3$
     - Function *f* is
       - Unate and negative in x_1
       - Unate and positive in x_3
       - Not Unate in x_2

Unate Functions

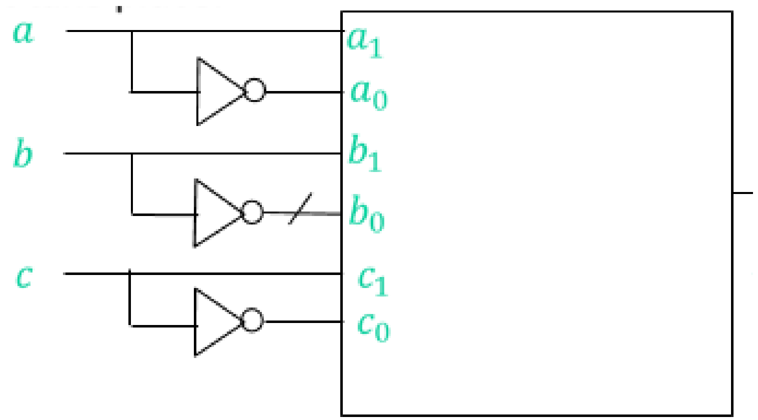
Consider a function $f(x_1, x_2, \dots x_n)$

If f is unate in each of its variables, f is said to be UNATE

- Examples:
  1. $f(x_1, x_2, \dots x_n) = x_1{}'x_2 + x_1x_2x_3{}'$
     - Function f is Unate as it can be simplified to
       $$f(x_1, x_2, \dots x_n) = x_1{}'x_2 + x_2x_3{}'$$
     - Positive Unate in x_2, negative Unate in x_1 and x_3
  2. $f(x_1, x_2, \dots x_n) = x_1x_2{}' + x_1{}'x_2$
     - Function f is not Unate in either variable

Fault Detection in Functions

- Let $f = ab'c + a'b'c + abc'$: Binate function
- Make it Unate by the following substitutions
  - $a_0 = a', a_1 = a$
  - $b_0 = b', b_1 = b$
  - $b_0 = c', c_1 = c$
- F=$a_1b_0c_1 + a_0b_0c_1 + a_1b_1c_0$: Unate function
- Unidirectional change in output



$$f = 0\ for\ (a, b, c) = (0, 1, 1)$$

$$f = 1\ for\ (a, b, c) = (0, 1, 1) and\ b_0 = 1$$

Symmetric Functions

- A function of n variables $x_1, x_2, \dots x_n$ is symmetric if and only if the interchange of any pair of variables leaves the function identically the same
- Examples:
- $xy' + x'y$ is symmetric in $x\ and\ y$
- $xyz + x'y'z'$ is symmetric in $x, y\ and\ z$

- If a function is symmetric, then the number of variables taking the value 1 determines the function rather than the specific assignment of 1's among the variables

Poll: Function Classification

Classify the following function:

$$k(A, B, C) = A'B + A'C$$

1. Unate
2. Binate
3. Symmetric

Summary of this Lesson

- Minimization of Boolean functions
- Don’t care conditions
- Special functions (Unate, Symmetric)

Post-work for Lesson 3

Homework

- **After the Live Lecture, you will complete and submit a homework assignment. Go to the online classroom to view and submit the assignment.**

To Prepare for the Next Lesson

- Read the Required Readings for Lesson 4.
- Complete the Pre-work for Lesson 4.
- Start working on the Project.

Go to the online classroom for details.