CSE 241

Boolean Algebra

Boolean Algebra

- An algebra for symbolically representing problems in logic & analyzing them mathematically
- Based on work of George Boole
  - *An Investigation of the Laws of Thought*
  - Published in 1854
- Switching Circuit Theory
  - Forms foundation for digital systems
  - Boolean algebra applied to logic design
  - Uses
    - Describe terminal properties of a logic network
    - Verification
    - Manipulation
    - Simplification
- Mathematical system consisting of
  - Set of elements, B
    - $B \in (0,1)$
  - Binary Operators
    - +
    - •
  - Equality Sign (=)
  - Parenthesis ()
    - Order of operations
- More Definitions
  - Constant
    - An element $\in$ B
    - 0, 1
  - Variable
    - Symbol representing an arbitrary element
- Principle of Duality
  - If an expression in Boolean algebra is valid, the dual of the expression must also be valid.
  - To obtain the dual of an expression:
    - Replace every operator + with •
    - Replace every operator • with +
    - Replace every 1 with 0
    - Replace every 0 with 1
- Order of Precedence
  - Parenthesis, Not, •, +

- Notational Notes
  - The (•) operator is often omitted from an expression
    - The juxtaposition of two variables implies the (•) operator.
  - The complement (x') is often written with a bar (⁻) over the variable or expression to be complemented.

Theorems & Postulates

- Operations (+) and (•) are closed
  - $x + y \in B$
  - $x \bullet y \in B$
- There exist at least two elements
  - $x, y \in B$
  - $x \neq y$
- Complement
  - $x + x' = 1$
    - Dual: $x \bullet x' = 0$
  - Unary Operator
- Identity Elements
  - Identity elements exist, such that for every element $x \in B$
    - $0 + x = x + 0 = x$
      - ✓Dual: $x \bullet 1 = 1 \bullet x = x$
- Complements of Identity Elements
  - $0' = 1$
    - Dual: $1' = 0$
- Idempotent Law
  - $x + x = x$
    - Dual: $x \bullet x = x$
- Involution Law
  - $(x')' = x$
- Absorption Law
  - $x + xy = x$
    - Dual: $x(x + y) = x$
- Theorem
  - $x + x'y = x + y$
    - Dual: $x(x' + y) = xy$
- Commutative Law
  - $x + y = y + x$
    - Dual: $x \bullet y = y \bullet x$
- Associative Law
  - $x + (y + z) = (x + y) + z$
    - Dual: $x (yz) = (xy) z$

- Distributive Law
  - x (y + z) = (xy) + (xz)
    - Dual: x + (yz) = (x + y)(x + z)
- DeMorgan's Law
  - (x + y)' = x'y'
    - Dual: (xy)' = x' + y'
  - Extension to more variables
    - (w + x + y + z + ...)' = w'x'y'z'...
      - ✓Dual: (wxyz)'... = w' + x' + y' + z' ...
- Consensus Theorem
  - xy + x'z + yz = xy + x'z
    - Dual: (x + y)(x' + z)(y + z) = (x + y)(x' + z)

Complementing a Function

- To complement a function, either
  - apply DeMorgan's Theorem
  - take dual of function and complement each literal

The Truth Table

- A table listing the output for every possible combination of inputs for an *n*-input function
- Inputs
  - Enumerated on left
    - Count from 0…0 to 1…1 in binary to enumerate all values
- Outputs
  - Enumerated on right
- Columns
  - *n* + 1 (minimum)
    - Often intermediate values are listed instead of just the output of the function
- Rows
  - 2^n

Two-Valued Boolean Algebra

- A Boolean algebra where B = {0,1}, and operators • and +
- AND (•)
  - Alternate Symbols
    - ∩
    - ∧

x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

- OR (+)
  - Alternate Symbols
    - ∪
    - ∨

x	y	x + y
0	0	0
0	1	1
1	0	1
1	1	1

Terminology

- Negation
  - Not Operation
- Product
  - AND Operation
- Sum
  - OR Operation
- Literal
  - Each occurrence of a variable in its complemented or uncomplemented form
- Product Term
  - Literal
  - Product (conjunction) of literals
- Sum Terms
  - Literal
  - Sum (disjunction) of literals
- Boolean Formula or Expression
  - Boolean variables & constants are connected with operators to describe a particular function

Boolean Formula Manipulation

- Complementing a Function
- Expanding About a Variable
  - $f(x_1,\ldots,x_i,\ldots,x_n) = x_i\, f(x_1\ldots, 1, \ldots, x_n) + x_i'\, f(x_1, \ldots, 0, \ldots, x_n)$
  - $f(x_1,\ldots,x_i,\ldots,x_n) = [x_i + f(x_1\ldots, 0, \ldots, x_n)]\,[x_i' + f(x_1, \ldots, 1, \ldots, x_n)$
- Equation Simplification
  - Reduction of the number of literals
  - Reduction Theorems
    - $x_i\, f(x_1,\ldots,x_i,\ldots,x_n) = x_i\, f(x_1\ldots, 1, \ldots, x_n)$
    - $x_i + f(x_1,\ldots,x_i,\ldots,x_n) = x_i + f(x_1\ldots, 0, \ldots, x_n)$
    - $x_i'\, f(x_1,\ldots,x_i,\ldots,x_n) = x_i'\, f(x_1\ldots, 0, \ldots, x_n)$
    - $x_i' + f(x_1,\ldots,x_i,\ldots,x_n) = x_i' + f(x_1\ldots, 1, \ldots, x_n)$

Examples

- Determine the dual of $F = xy + z$
  - $(x + y)z$

- Determine F', where $F = (x + y')z + y$
  - $F' = (x'y + z')\, y'$

- Simplify: $F = x'yz' + xyz' + xyz + xyz'$
  - ☞ $F = yz' + xy$

- Simplify: $F = x'y'z + x'yz + xy'z + xyz$
  - ☞ $F = z$

- Simplify: $F = w'x'z' + xy'z + wxy'z' + w'xy'z + x'z'$
  - ☞ $F = x'z' + xy'z + wxy'$
  - ☞ $F = x'z' + xy'z + wy'z'$

- Simplify: $F = y'z' + x'yz + x'y + xyz + xz'$
  - ☞ $F = y + z'$

Canonical Forms

- The standard form of an equation consists of product or sum terms
  - ☞Referred to as the ***canonical form***
  - ☞Two forms
    - ⇘POS
    - ⇘SOP

Two Canonical Forms

- SOP & POS Forms
  - ☞Sum of Products (SOP)
    - ⇘Formed by summing products terms
      - ✓Each product formed by ANDing literals
    - ⇘Example
      - ✓f(a,b,c,d) = a’b + ac’ + abcd
  - ☞Product of Sums (POS)
    - ⇘Formed by taking the product of sum terms
      - ✓Each sum formed by ORing literals
    - ⇘Example
      - ✓f(a,b,c,d) = (a+b) (b+c+d’) (a+b’+c’)
  - ☞Note
    - ⇘A sum is formed by using the OR operator
    - ⇘A product is formed by using the AND operator

Canonical Sum of Products

- *aka*
  - ☞Canonical SOP
  - ☞Standard SOP
  - ☞Disjunctive Normal Form
  - ☞Disjunctive Canonical Formula
  - ☞Minterm Expansion
  - ☞Minterm Canonical Formula
- Minterm
  - ☞An product of literals in which each variable is represented once and only once in either its complemented or uncomplemented form.
- Minterms are ORed to form the canonical SOP
- Shorthand Notation
  - ☞Each minterm is represented by an *n*-bit binary code as follows
    - ⇘Let an uncomplented variable represent 1
    - ⇘Let a complemented variable represent 0
  - ☞Each minterm is represented by m_i
    - ⇘where i is the decimal integer equivalent of the binary code representing the minterm
  - ☞If the minterm, m_i , evaluates to 1, the minterm is included in the expression
  - ☞Hence, the function, f(a,b,c) = $\sum m_i$
    - ⇘where m_i is a minterm that evaluates to 1

- ☞Example
  - ⇘Consider f(a,b,c) = a’bc’ + a’bc + ab’c
  - ⇘Truth Table

a b c	f(a,b,c)	Minterm
0 0 0	0	0
0 0 1	0	1
0 1 0	1	2
0 1 1	1	3
1 0 0	0	4
1 0 1	1	5
1 1 0	0	6
1 1 1	0	7

  - ⇘$f(a,b,c) = \sum m(2,3,5) = m_2 + m_3 + m_5$
  - ⇘$f'(a,b,c) = \sum m(0,1,4,6,7) = m_0 + m_1 + m_4 + m_6 + m_7$
- ☞Minterm list form
  - ⇘The shorthand notation represented above as
    $f(a,b,c) = \sum m(2,3,5)$

Canonical Product of Sums

- *aka*
  - ☞Canonical POS
  - ☞Standard POS
  - ☞Conjunctive Normal Form
  - ☞Conjunctive Canonical Formula
  - ☞Maxterm Expansion
  - ☞Maxterm Canonical Formula
- Maxterm
  - ☞An sum of literals in which each variable is represented once and only once in either its complemented or uncomplemented form.
- Maxterms are ANDed to form the canonical POS
- Shorthand Notation
  - ☞Each maxterm is represented by an *n*-bit binary code as follows
    - ⇘Let an uncomplemented variable represent 0
    - ⇘Let a complemented variable represent 1
  - ☞Each maxterm is represented by M_i
    - ⇘where *i* is the decimal integer equivalent of the binary code representing the maxterm
  - ☞If the maxterm, M_i , evaluates to 0, the maxterm is included in the expression
  - ☞Hence, the function, $f(a,b,c) = \prod M_i$
    - ⇘where M_i is a maxterm that evaluates to 0
  - ☞Example
    - ⇘Consider f(a,b,c) = (a+b+c) (a+b’+c) (a’+b+c) (a’+b’+c’)
    - ⇘Truth Table

a b c	f(a,b,c)	Maxterm
0 0 0	0	0

0 0 1	1	1
0 1 0	0	2
0 1 1	1	3
1 0 0	0	4
1 0 1	1	5
1 1 0	1	6
1 1 1	0	7

- $f(a,b,c) = \prod M(0,2,4,7) = M_0 M_2 M_4 M_7$
- $f'(a,b,c) = \prod M(1,3,5,6) = M_1 M_3 M_5 M_6$

☞Maxterm list form
- The shorthand notation represented above as
  $f(a,b,c) = \prod M(0,2,4,7)$

Derivation of Minterm & Maxterm

- Minterm
  - ☞The function not equal to 0 with a minimum number of 1's in the truth table
- Maxterm
  - ☞The function not equal to 1 with a minimum number of 0's in the truth table
- Note
  - ☞For a function, F, $M_j = m_j'$

Summary

- 2^n minterms (maxterms) exist for *n* Boolean variables
  - ☞These minterms (maxterms) can be represented by the binary numbers 0 through 2^n-1
- Any Boolean function can be represented as a logical sum (product) of minterms (maxterms)
- The complement of a function, F', consists of those minterms (maxterms) not included in the original function, F.
  - ☞Example
    - $F(w,x,y,z) = \sum m(0,1,6,10,11,14,15)$
      - ✓ $F'(w,x,y,z) = \sum m(2,3,4,5,7,8,9,12,13)$
    - $F(x,y,z) = \Pi M(2,6,7)$
      - ✓ $F'(x,y,z) = \Pi M(0,1,3,4,5)$
- A function, F, which includes all 2^n possible minterms (maxterms) is equal to 1 (0).

Conversion Between Canonical Forms

- Canonical SOP to canonical POS
  - ☞Write expression in minterm list form
  - ☞Replace $\sum$ with $\prod$
  - ☞Replace minterm numbers with those not used in list
  - ☞Replace m with M
  - ☞Example
    - $f(a,b,c) = \sum m(1,3,6) = \prod M(0,2,4,5,7)$

- Canonical POS to canonical SOP
  - Write expression in maxterm list form
  - Replace $\prod$ with $\sum$
  - Replace maxterm numbers with those not used in list
  - Replace M with m
  - Example
    - $f(a,b,c,d)=\prod M(0,2,4,7,10,12,13,15)=\sum m(1,3,5,6,8,9,11,14)$

Conversion From Noncanonical to Canonical Form

- SOP
  - Apply distributive law to get an SOP represenation
  - Add literals to create minterms by repeatedly applying
    - $xy + xy' = x$
  - Eliminate redundant terms
- POS
  - Apply distributive law to get an POS represenation
  - Add literals to create maxterms by repeatedly applying
    - $(x+y)(x+y') = x$
  - Eliminate redundant terms
- Example
  - $f(a,b,c) = a(b+c') + a'bc$
  - $f(a,b,c) = ab + ac' + a'bc$
  - $f(a,b,c) = ab(c+c') + ac'(b+b') + a'bc$
  - $f(a,b,c) = abc + abc' + abc' + ab'c' + a'bc$
  - $f(a,b,c) = abc + abc' + ab'c' + a'bc$
  - $f(a,b,c) = \sum m(3,4,6,7)$
- Shannon's Expansion Theorem can also be applied

Incompletely Specified Functions

- Functions may not be completely specified, allowing certain minterms or maxterms to be undefined in truth table.
- Why?
  - Certain input patterns may never be applied.
  - Where all input patterns do occur, only an output of 0 or 1 may be only required for certain input patterns.
- For canonical SOP or POS, don't care minterms or maxterms are specified as dc_i or d_i.
- Allows flexibility for optimal designs.
- Once circuit is designed, the circuit will have a defined value for all don't care conditions.
- Examples
  - $f(a,b,c) = \sum m(1,3,6) + dc(4,5,7)$
  - $f(x,y,z) = \prod M(0,2,4,6,7) + d(1)$

References

- M. Morris Mano and Charles R. Kime, *Logic and Computer Design Fundamentals*, Prentice Hall, Inc., 2000
- Victor P. Nelson, H. Troy Nagle, Bill D. Carroll, and J. David Irwin, *Digital Logic Circuit Analysis & Design*, Prentice-Hall, Inc., 1995
- Donald D. Givone, *Digital Principles and Design*, McGraw-Hill, 2003