Review of Boolean Algebra and Karnaugh Maps

CS 2461

1

Theory of Combinational Logic Design ?

- Is there a well grounded theory behind design of boolean logic circuits/functions ?
- Equivalent circuits ?
- Efficient design ?
  - Fewest gates used

CS 2461

2

Boolean Algebra

- To describe behavior of combinational circuit
  - Truth table
  - Boolean algebraic expressions
  - Digital logic circuit/diagram
- Algebraic expression written according to laws of boolean algebra specifies not only what a combinational circuit does, but also how it does it!

Boolean Algebra – Definitions..recall from Discrete Math

- Boolean algebra has three operations defined over boolean variables:
  - OR (+), AND (.) and complement (’)
- Recall fundamental properties of Boolean algebra
  - These apply to anything that is a boolean algebra
    - Sets, digital logic circuits, …

Boolean Algebra– Fundamental Properties

- Commutative:
  - $x+y = y+x$ $\quad$ $x.y = y.x$
- Associative
  - $(x+y)+z = x+(y+z)$ $\quad$ $(x.y).z = x.(y.z)$
- Distributive
  - $x+(y.z) = (x+y).(x+z)$ $\quad$ $x.(y+z)=(x.y)+(x.z)$
- Identity
  - $x+0 = x$ $\quad$ $x.1 = x$
- Complement
  - $x + (x') = 1$ $\quad$ $x.(x') = 0$

Laws of Boolean algebra

- Duality property: each boolean property has a dual property
  - Exchange + and . Exchange 1 and 0
- Many useful properties/theorems can be proved from the 10 fundamental properties

Example: Idempotent Property

- Prove: $x + x = x$
- Proof: use only the 10 fundamental laws
- $x+x = (x+x).1$; From identity property
- $(x+x).1 = (x+x).(x+x')$; complement
- $(x+x).(x+x')= (x.x) + (x.x')$; distributive
- $x + (x.x')= x +0$; complement
- $x+0 = x$; identity property
  - QED
- The duality property is: $x.x =x$

CS 2461

7

Some useful properties. . .

- Zero theorem
  - $x+1 =1$ $x.0 = 0$
- Absorption property
  - $x + x.y = x$ $x.(x+y) = x$
- De Morgan's law
  - $(a.b)' = a' + b'$ $(a+b)' = a' . b'$
  - $(a.b.c)' = a'+b'+c'$ $(a+b+c)'= a'.b'.c'$
- Complement
  - $(x')' = x$
  -

CS 2461

8

Two Level Circuits

- Every boolean expression can be transformed to an AND-OR expression
  - Resulting in a 2 level circuit
- Advantage of 2 level circuit ?
  - Gate delays
    - Go through only two levels/layers of gates

CS 2461

9

Why this discussion of Boolean Algebra…

- Every boolean expression has a corresponding logic circuit diagram; and every logic circuit diagram has a corresponding boolean expression
  - One to one correspondence
- But a given truth table can have several corresponding implementations
- How to map from truth table to boolean expression ?
  - How to pick the "best" boolean expression ?

CS 2461

10

Simplification of boolean expressions

- The boolean expression/function $x(a,b,c,d) = a'bd' + a'c'd' + a'bc'd'$
- Can be simplified using absorption property to

  $a'bd' + (a'c'd')+(a'c'd')b = a'bd'+ a'c'd'$

CS 2461

11

Combinational Circuit Design: Truth Tables and Boolean expressions

- Given truth table we want to find an "efficient" implementation (i.e., circuit)
  - Efficient in speed
  - Efficient in number of gates
  - Simplicity of design
  - Canonical boolean expression
- Graphical method for designing two level circuits with 3 or 4 variables using minimum possible number of gates
  - What is this method ????

CS 2461

12

Canonical Expressions

- Consider boolean expression *x*, where
  $x(a,b,c) = abc + a'bc + ab$
  - First two are minterms since they contain all three input variables
- $abc + a'bc + ab = abc + a'bc + ab(c+c')$

$$= abc + a'bc + abc + abc'$$
$$= abc + a'bc + abc'$$

Truth table?

Example

a	b	c	x
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Transformation of the Boolean expression

- $abc + a'bc + abc'$
  $= (a + a')bc + abc'$
  $= bc + abc'$
- $a'bc + abc + abc'$
  $= a'bc + ab(c + c')$
  $= a'bc + ab$
- $a'bc + abc + abc'$
  $= a'bc + abc + abc + abc'$
  $= (a'+a)bc + ab\,(c+c')$
  $= bc + ab$

Distance between minterms

- Concept of "distance" between two minterms (Hamming distance):
  - Number of variables that are different
  - Distance(abc, abc')=1 only c and c' different
  - Distance(abc, a'bc')=2 both a and c are different
- Arrange 2-d truth table so that values in consecutive columns(rows) differ in one bit position

Truth table in 2-dimensions

b (b=1)

a \ bc	00	01	11	10
0	0	0	1	0
1	0	0	1	1

(a (a=1)

c (c=1)

$x_1 = bc$ $x_2 = ab$

Therefore, x = a'bc + abc + abc' = bc + ab

CS 2461

17

Karnaugh Maps

- Graphical way to represent boolean functions
  - Based on concept of distance
- Recognizing adjacent minterms is key to minimization of AND-OR expression
- K-map is a tool to minimize a two level circuit that it makes it easy to spot adjacent minterms
- Karnaugh Map is a truth table arranged so that adjacent entries represent minterms that differ by one.

CS 2461

18

Grouping minterms in K-Map

- Group 'cells' in K-map that are adjacent and have a value of 1 in the cell
  - Group of 2 cells in 3 variable K-map: is an AND of two variables
  - Group of 4 cells in 3 variable K-map: is single variable

Minimization using K-Maps

- Minimization procedure : determine best set of groups that will cover all the 1's in the K-map
  - "best" means the set that corresponds to a two-level circuit with the least number of gates and the least number of inputs per gate.
  - The number of groups equals number of AND gates
  - We want the smallest number of groups with each group as large as possible such that the groups cover all the 1's

Example

bc

b (b=1)

a	00	01	11	10
0	1	1	0	0
1	0	1	1	0

(a (a=1)

c (c=1)

21

Example

bc

b (b=1)

a	00	01	11	10
0	1	1	0	0
1	0	1	1	0

(a (a=1)

c (c=1)

22

Example

bc

b (b=1)

a	00	01	11	10
0	1	1	0	0
1	0	1	1	0

(a (a=1)

c (c=1)

a'b' + b'c + ac

CS 2461

23

Example

bc

b (b=1)

a	00	01	11	10
0	1	1	0	0
1	0	1	1	0

(a (a=1)

c (c=1)

a'b' + ac

CS 2461

24

4 variable Karnaugh map

ab \ cd	00	01	11	10
00	1			
01	1		1	
11	1	1	1	
10	1			

a b c d

CS 2461

25

Summary of Combinational Logic

- Combinational device/circuit: any circuit built using the basic gates
- Expressed as
  - Truth table
  - Digital circuit
  - Boolean function
- Any boolean function can be expressed as two level function
- Minimization procedure: Karnaugh Map
  - Try to minimize the number of gates, and inputs to gates, in a two level circuit

CS 2461

26