BOOLEAN ALGEBRA: PART 1

- Principles
- Boolean functions

Boolean algebra: Axiomatic definition

A Boolean algebra consists of

A set of elements B

Binary operators {+ , •}   Boolean sum and product

A unary operation { ' } (or { $\overline{\ }$ })   example: A’ or $\overline{A}$

…and the following axioms

1. The set B contains at least two elements {a b} with a $\neq$ b
2. Closure: a+b is in B a•b is in B a’ is in B
3. Commutative: a+b = b+a a•b = b•a
4. Associative: a+(b+c) = (a+b)+c a•(b•c) = (a•b)•c
5. Two neutral elements denotes 0 and 1: a+0 = a a•1 = a
6. Distributive: a+(b•c)=(a+b)•(a+c) a•(b+c)=(a•b)+(a•c)
7. Complementarity: for every element a, a’ exists such that

   a+a' = 1 a•a' = 0

BOOLEAN ALGEBRA: EXHAUSTIVE DEFINITION

A SET OF TWO ELEMENTS denoted 0 and 1 and 3 operators defined below by Truth tables.

AND $a \bullet b$ or ab

a	b	ab
0	0	0
0	1	0
1	0	0
1	1	1

OR $a+b$

a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	1

NOT $\overline{a}$ or a'

a	a’
0	1
1	0

Digital (binary) logic is a Boolean algebra

Substitute

- {0, 1} for B
- AND for • Boolean Product. In logic this was $\wedge$
- OR for + Boolean Sum. In logic this was $\vee$
- NOT for ‘ Complement. In logic this was $\neg$

All the axioms hold for binary logic

Ambiguity:

$a + bc = a + (bc)$ or $a+bc = (a+b)c$?

Priority rule: “and” has priority (must be executed before) over “or”.

So $A + BC = A + (BC)$

Same rule as in standard arithmetic: multiplication has priority over addition.

Variables and their complements are sometimes called literals

Useful laws and theorems

Absorption:	$a + 1 = 1$	Dual: $a \cdot 0 = 0$
Idempotent:	$a + a = a$	Dual: $a \cdot a = a$
Idempotent:	$(a')' = a$	
Uniting:	$a \cdot b + a \cdot b' = a$	Dual: $(a+b) \cdot (a+b') = a$

Useful laws and theorems (con’t)

Absorption: a+a•b=a Dual: a•(a+b)=a

Absorption (#2): (a+b')•b=a•b Dual: (a•b')+b=a+b

de Morgan's: (a+b+...)'=a'•b'•... Dual: (a•b•...)'=a'+b'+...

Multiplying & factoring: (a+b)•(a'+c)=a•c+a'•b

Dual: a•b+a'•c=(a+c)•(a'+b)

Consensus: (a•b)+(b•c)+(a'•c)= a•b+a'•c

Dual: (a+b)•(b+c)•(a'+c)=(a+b)•(a'+c)

DEMONSTRATION USING TRUTH TABLE

a	b	(a+b)	(a+b’)	(a+b)(a+b’)
0	0	0	1	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

=

(a+b) (a+b’) = a

BOOLEAN FUNCTION: EXAMPLE

abc	F(a,b,c)
000	0
001	0
010	1
011	1
100	0
101	1
110	0
111	1

Truth Table

Algebraic format

F = A’B+AC

REMARK: easy to derive Truth Table from algebraic expression but going in the other direction is not straightforward

NAND $(a \bullet b)'$ or $(ab)'$

a	b	(ab)'
0	0	1
0	1	1
1	0	1
1	1	0

NOR $(a+b)'$

a	b	(a+b)'
0	0	1
0	1	0
1	0	0
1	1	0

XOR (Ou Exclusif) $a \oplus b$

a	b	$a \oplus b$
0	0	1
0	1	0
1	0	0
1	1	1

GENERAL AND (COMPLETE) MINTERM

DEFINITION: For a boolean function of **n** variables $x_{(n-1)},x_{(n-2)}.....x_2,x_1,x_0$, a product term in which each of the n variables appears **at most once** (either in its complemented or uncomplemented form) is called a **general minterm.**

Thus, a **general minterm** is a logical expression of n variables that employs only the complement operator and the disjunction (AND) operator.

DEFINITION (Wikipedia): For a boolean function of **n** variables $x_{(n-1)},x_{(n-2)}.....x_2,x_1,x_0$, a product term in which each of the **n** variables appears **exactly once** (either in its complemented or uncomplemented form) is called a **(complete) minterm.**

COMPLETE MINTERM DEFINED BY A VECTOR (2)

Let us consider a n bit vector $V = (v_{(n-1)}, v_{(n-2)} \ldots .. v_2, v_1, v_0)$, the complete minterm associated with V is defined by the formula

$$Min_V(x_{(n-1)}, x_{(n-2)} \ldots .. x_2, x_1, x_0) = (1 \oplus x_{(n-1)} \oplus v_{(n-1)})\,(1 \oplus x_{(n-2)} \oplus v_{(n-2)}) \ldots .\,(1 \oplus x_1 \oplus v_1)\,(1 \oplus x_0 \oplus v_0)$$

MINTERM/MAXTERM: VECTOR /NUMBER

VECTOR	Minterm	Number	Name
(0,0,0,1,0)	$x'_4.x'_3.x'_2.x_1.x'_0$	2	m_2
(1,0,0,0,1)	$x_4.x'_3.x'_2.x'_1.x_0$	17	m_{17}
(1,0,0,1,1)	$x_4.x'_3.x'_2.x_1.x_0$	19	m_{19}
(0,1,1,1,0)	$x'_4.x_3.x_2.x_1.x'_0$	14	m_{14}

VECTOR	Maxterm	Number	Name
(0,0,0,1,0)	$x_4+x_3+x_2+x'_1+x_0$	2	M_2
(1,0,0,0,1)	$x'_4+x_3+x_2+x_1+x'_0$	17	M_{17}
(1,0,0,1,1)	$x'_4+x_3+x_2+x'_1+x'_0$	19	M_{19}
(0,1,1,1,0)	$x_4+x'_3+x'_2+x'_1+x_0$	14	M_{14}

COMPLETE MINTERMS 3 VARIABLES

VECTOR	Minterm	Number	Name
(0,0,0)	$x'_2.x'_1.x'_0$	0	m_0
(0,0,1)	$x'_2.x'_1.x_0$	1	m_1
(0,1,0)	$x'_2.x_1.x'_0$	2	m_2
(0,1,1)	$x'_2.x_1.x_0$	3	m_3
(1,0,0)	$x_2.x'_1.x'_0$	4	m_4
(1,0,1)	$x_2.x'_1.x_0$	5	m_5
(1,1,0)	$x_2.x_1.x'_0$	6	m_6
(1,1,1)	$x_2.x_1.x_0$	7	m_7

COMPLETE MAXTERMS 3 VARIABLES

VECTOR	Maxterm	Number	Name
(0,0,0)	$x_2+x_1+x_0$	0	M_0
(0,0,1)	$x_2+x_1+x'_0$	1	M_1
(0,1,0)	$x_2+x'_1+x_0$	2	M_2
(0,1,1)	$x_2+x'_1+x'_0$	3	M_3
(1,0,0)	$x'_2+x_1+x_0$	4	M_4
(1,0,1)	$x'_2+x_1+x'_0$	5	M_5
(1,1,0)	$x'_2+x'_1+x_0$	6	M_6
(1,1,1)	$x'_2+x'_1+x'_0$	7	M_7

Theorem: Let F be a Boolean function and Sup (F) (called support of F) the set of Boolean vectors for which F value is 1, then

$$F = \sum_{V\ in\ Sup(F)} Min_V$$

Theorem: Let F be a Boolean function and ASup(F) (called AntiSupport of F) the set of Boolean vectors for which F value is 0, then

$F = \prod_{V\ in\ ASup(F)} Max_V$

/

MINTERM CANONICAL FORM (EXAMPLE)

$$m_1 + m_2 + m_3 + m_5 + m_6 = F(a,b,c) = \sum m(1,2,3,5,6)$$

a'b'c + a'bc' + a'bc + ab'c + abc'

	abc	F(a,b,c)
0	000	0
1	001	1
2	010	1
3	011	1
4	100	0
5	101	1
6	110	1
7	111	0

MAXTERM CANONICAL FORM (EXAMPLE)

$M_0 \quad . \quad M_4 \quad . \quad M_7 \quad = F(a,b,c) = \prod M(0,4,7)$

(a+b+c) (a'+b+c) + (a'+b'+c')

	abc	F(a,b,c)
0	000	0
1	001	1
2	010	1
3	011	1
4	100	0
5	101	1
6	110	1
7	111	0

Acknowledgements

A lot of material was found in Wikipedia.

Some of these slides were inspired by slides developed by:

- University of Washington Computer Science & Engineering (CSE 370)
- Y.N. Patt (Univ of Texas Austin)
- S. J. Patel (Univ of Illinois Urbana Champaign)
- Walid A. Najjar (Univ California Riverside)
- Brian Linard (Univ California Riverside)
- G.T. Byrd (Univ North Carolina)