Sistemas Digitais I

LESI - 2º ano

Unit 3 - Boolean Algebra

João Miguel Fernandes
www.di.uminho.pt/~jmf

DEP. DE INFORMÁTICA
ESCOLA DE ENGENHARIA
UNIVERSIDADE DO MINHO

3. Boolean Algebra

- Summary -

- Binary Signals
- Combinational vs. Sequential Systems
- Gates
- Switching Algebra
- Axioms
- Theorems
- Duality
- Standard Representation
- Examples

3. Boolean Algebra

- Introduction -

- The success of computer technology is primarily based on simplicity of designing digital circuits and ease of their manufacture.
- Digital circuits are composed of basic processing elements, called gates, and basic memory elements, called flip-flops.
- The simplicity in digital circuit design is due to the fact that input and output signals of each gate or flip-flop can assume only two values, 0 and 1.
- The changes in signal values are governed by laws of Boolean algebra.
- The fact that Boolean algebra is finite and richer in properties than ordinary algebra leads to simple optimisation techniques for functions.
- In order to learn techniques for design of digital circuits, we must understand the properties of Boolean algebra.

3. Boolean Algebra

- Binary Signals (1) -

- Digital logic hides the analog world by mapping the infinite set of real values into 2 subsets (0 and 1).
- A logic value, 0 or 1, is often called a binary digit (bit).
- With n bits, 2^n different entities are represented.
- When using electronic circuits, digital designers often use the words “LOW” and “HIGH”, in place of “0” and “1”.
- The assignment of 0 to LOW and 1 to HIGH is called positive logic. The opposite assignment is called negative logic.
- Other technologies can be used to represent bits with physical states.

3. Boolean Algebra

- Binary Signals (2) -

	State Representing Bit	
Technology	***0***	***1***
Pneumatic logic	Fluid at low pressure	Fluid at high pressure
Relay logic	Circuit open	Circuit closed
Complementary metal-oxide semiconductor (CMOS) logic	0–1.5 V	3.5–5.0 V
Transistor-transistor logic (TTL)	0–0.8 V	2.0–5.0 V
Fiber optics	Light off	Light on
Dynamic memory	Capacitor discharged	Capacitor charged
Nonvolatile, erasable memory	Electrons trapped	Electrons released
Bipolar read-only memory	Fuse blown	Fuse intact
Bubble memory	No magnetic bubble	Bubble present
Magnetic tape or disk	Flux direction "north"	Flux direction "south"
Polymer memory	Molecule in state A	Molecule in state B
Read-only compact disc	No pit	Pit
Rewriteable compact disc	Dye in crystalline state	Dye in noncrystalline state

3. Boolean Algebra

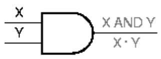
- Combinational vs. Sequential Systems -

- A combinational logic system is one whose outputs depend only on its current inputs.
- A combinational system can be described by a truth table.
- The outputs of a sequential logic circuit depend not only on the current inputs but also on the past sequence of inputs $\Rightarrow$ memory.
- A sequential system can be described by a state table.
- A combinational system may contain any number of logic gates but no feedback loops.
- A feedback loop is a signal path of a circuit that allows the output of a gate to propagate back to the input of that same gate.
- Feedback loops generally create sequential circuit behaviour.

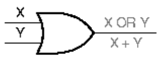
3. Boolean Algebra

- Gates (1) -

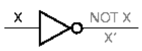
- Three basic gates (AND, OR, NOT) are sufficient to build any combinational digital logic system. They form a <u>complete set</u>.



X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1



X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1



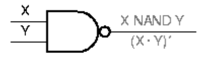
X	NOT X
0	1
1	0

- The symbols and truth tables for AND and OR may be extended to gates with any number of inputs.
- The <u>bubble</u> on the inverter output denotes "inverting" behaviour.

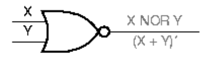
3. Boolean Algebra

- Gates (2) -

- Two more logic functions are obtained by combining NOT with an AND or OR function in a single gate.



X	Y	X NAND Y
0	0	1
0	1	1
1	0	1
1	1	0

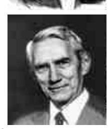


X	Y	X NOR Y
0	0	1
0	1	0
1	0	0
1	1	0

- The symbols and truth tables for NAND and NOR may also be extended to gates with any number of inputs.

3. Boolean Algebra

- Switching Algebra -

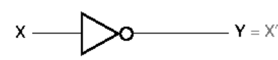


- In 1854, G. Boole (1815-1865) introduced the formalism that we use for the systematic treatment of logic which is now called <u>Boolean Algebra</u>.
- In 1938, C. Shannon (1916-2001) applied this algebra to prove that the properties of electrical switching circuits can be represented by a 2-valued Boolean Algebra, which is called <u>Switching Algebra.</u>
- Using this algebra, one can formulate propositions that are true or false, combine them to make new propositions and determine if the new propositions are true or false.
- We use a symbolic variable (ex. X) to represent the condition of a logic signal, which is in one of two possible values ("0" or "1").

3. Boolean Algebra

- Axioms (1) -

- The <u>axioms</u> (or postulates) of a mathematical system are a minimal set of basic definitions that we assume to be true.
- The first axioms embody the digital abstraction:
  (A1) X=0 if X≠1 (A1') X=1 if X≠0
- We stated these axioms as a pair, the only difference being the interchange of the symbols 0 and 1.
- This applies to all the axioms and is the basis of <u>duality</u>.
- The next axioms embody the complement notation:
  (A2) If X=0, then X'=1 (A2') If X=1, then X'=0

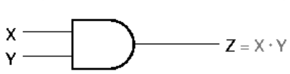


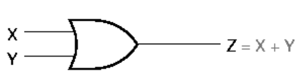
- We use a prime (') to denote an inverter's function.

3. Boolean Algebra

- Axioms (2) -

- The last three pairs of axioms state the formal definitions of the AND (logical multiplication) and OR (logical addition) operations:
  (A3) 0·0 = 0 (A3') 1+1 = 1
  (A4) 1·1 = 1 (A4') 0+0 = 0
  (A5) 0·1 = 1·0 = 0 (A5') 1+0 = 0+1 = 1





- By convention, in a logic expression involving both multiplication and addition, multiplication has precedence.
- The expression X·Y+Y·Z' is equivalent to (X·Y) + (Y·Z').
- The axioms (A1-A5, A1'-A5') completely define Boolean algebra.

3. Boolean Algebra

- Theorems (1) -

- <u>Theorems</u> are statements, known to be true, that allow us to manipulate algebraic expressions to have simpler analysis or more efficient synthesis of the corresponding circuits.
- Theorems involving a single variable:
  (T1) X+0 = X (T1') X·1 = X (Identities)
  (T2) X+1 = 1 (T2') X·0 = 0 (Null elements)
  (T3) X+X = X (T3') X·X = X (Idempotency)
  (T4) (X')' = X (Involution)
  (T5) X+X' = 1 (T5') X·X' = 0 (Complements)
- These theorems can be proved to be true. Let us prove T1:
  [X=0] 0+0=0 (true, according to A4')
  [X=1] 1+0=1 (true, according to A5')

3. Boolean Algebra
- Theorems (2) -

- Theorems involving two or three variables:

(T6)	X+Y = Y+X	(T6’)	X·Y = Y·X	(Commutativity)
(T7)	(X+Y)+Z = X+(Y+Z)	(T7’)	(X·Y)·Z = X·(Y·Z)	(Associativity)
(T8)	X·Y+X·Z = X·(Y+Z)	(T8’)	(X+Y)·(X+Z) = X+Y·Z	(Distributivity)
(T9)	X+X·Y = X	(T9’)	X·(X+Y) = X	(Covering)
(T10)	X·Y+X·Y’ = X	(T10’)	(X+Y)·(X+Y’) = X	(Combining)
(T11)	X·Y+X’·Z+Y·Z = X·Y+X’·Z			(Consensus)
(T11’)	(X+Y)·(X’+Z)·(Y+Z) = (X+Y)·(X’+Z)			

- Attention to theorem T8’ which is not true for integers and reals.
- T9 and T10 are used in the minimisation of logic functions.

3. Boolean Algebra
- Theorems (3) -

- Several important theorems are true for an arbitrary number of variables.
- Theorems involving n variables:

(T12) X+X+ ... +X = X — Generalised Idempotency
(T12’) X·X· ... ·X = X

(T13) $(X_1 \cdot X_2 \cdot \ldots \cdot X_n)' = X_1'+X_2'+ \ldots +X_n'$ — DeMorgan’s theorems
(T13’) $(X_1+X_2+ \ldots +X_n)' = X_1' \cdot X_2' \cdot \ldots \cdot X_n'$

(T14) $[F(X_1,X_2,\ldots,X_n,+,\cdot)]' = F(X_1',X_2',\ldots,X_n',\cdot,+)$ — Generalised DeMorgan’s th.

(T15) $F(X_1,X_2,\ldots,X_n) = X_1 \cdot F(1,X_2,\ldots,X_n) + X_1' \cdot F(0,X_2,\ldots,X_n)$ — Shannon’s expansion theorems
(T15’) $F(X_1,X_2,\ldots,X_n) = [X_1+F(0,X_2,\ldots,X_n)] \cdot [X_1'+F(1,X_2,\ldots,X_n)]$

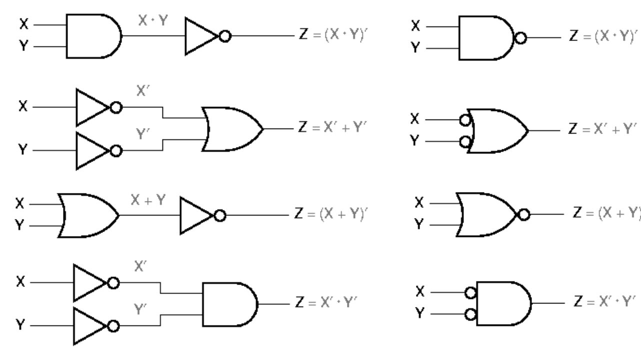
3. Boolean Algebra
- Theorems (4) -

- DeMorgan’s theorem (T13 and T13’) for n=2:
  (X·Y)’ = X’+Y’
  (X+Y)’ = X’·Y’
- DeMorgan’s theorem gives a procedure for complementing a logic function.
- DeMorgan's theorem can be used to convert AND/OR expressions to OR/AND expressions.
- Example:
  Z = A’ B’ C + A’ B C + A B’ C + A B C’
  Z’ = (A + B + C’) · (A + B’ + C’) · (A’ + B + C’) · (A’ + B’ + C)

Augustus De Morgan (1806-1871)

3. Boolean Algebra
- Theorems (5) -

Equivalent gates according to DeMorgan’s theorem



3. Boolean Algebra
- Theorems (6) -

- Since Boolean algebra has only two elements, we can also show the validity of these theorems by using truth tables.
- To do this, a truth table is built for each side of the equation that appears in the theorem.
- Then both sides of the equation are checked to see if they yield identical results for all the combinations of variable values.
- Let us prove DeMorgan’s theorem (T13 and T13’) for n=2:
  (X+Y)’ = X’·Y’ (X·Y)’ = X’+Y’

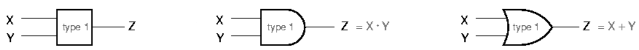
X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X+Y}$	$\overline{X}\cdot\overline{Y}$
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X\cdot Y}$	$\overline{X}+\overline{Y}$
0	0	1	1	1	1
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	0	0

3. Boolean Algebra
- Duality (1) -

- Theorems were presented in pairs.
- The prime version of a theorem is obtained from the unprimed version by swapping “0” and “1”, and “·” and “+”.
- Principle of Duality: Any theorem or identity in Boolean algebra remains true if 0 and 1 are swapped and · and + are swapped throughout.
- Duality is important because it doubles the usefulness of everything about Boolean algebra and manipulation of logic functions.
- The dual of a logic expression is the same expression with + and · swapped: $F^D(X_1,X_2,\ldots,X_n,+,\cdot,') = F(X_1,X_2,\ldots,X_n,\cdot,+,')$.
- Do not confuse duality with DeMorgan’s theorems!
  $[F(X_1,X_2,\ldots,X_n,+,\cdot)]' = F(X_1',X_2',\ldots,X_n',\cdot,+)$
  $[F(X_1,X_2,\ldots,X_n)]' = F^D(X_1',X_2',\ldots,X_n')$

3. Boolean Algebra
- Duality (2) -

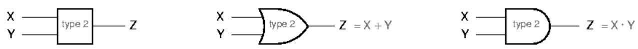


X	Y	Z
LOW	LOW	LOW
LOW	HIGH	LOW
HIGH	LOW	LOW
HIGH	HIGH	HIGH

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	Z
1	1	1
1	0	1
0	1	1
0	0	0

Electric function　　Positive-logic　　Negative-logic



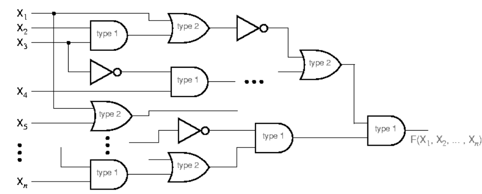
X	Y	Z
LOW	LOW	LOW
LOW	HIGH	HIGH
HIGH	LOW	HIGH
HIGH	HIGH	HIGH

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

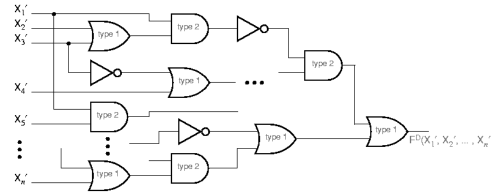
X	Y	Z
1	1	1
1	0	0
0	1	0
0	0	0

3. Boolean Algebra
- Duality (3) -

- Positive-logic



- Negative-logic



3. Boolean Algebra
- Standard Representation (1) -

- The most basic representation of a logic function is a truth table.
- A <u>truth table</u> lists the output of the circuit for every possible input combination.
- There are 2^n rows in a truth table for an n-variable function.

Row	X	Y	Z	F
0	0	0	0	F(0,0,0)
1	0	0	1	F(0,0,1)
2	0	1	0	F(0,1,0)
3	0	1	1	F(0,1,1)
4	1	0	0	F(1,0,0)
5	1	0	1	F(1,0,1)
6	1	1	0	F(1,1,0)
7	1	1	1	F(1,1,1)

Row	X	Y	Z	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

- There are 2^8 ($8=2^3$) different logic functions for 3 variables.

3. Boolean Algebra
- Standard Representation (2) -

- Truth tables can be converted to algebraic expressions.
- A <u>literal</u> is a variable or the complement of a variable. Ex: X, Y, X'.
- A <u>product term</u> is a single literal or a logical product of two or more literals. Ex: Z', W·X·Y, W·X'·Y'.
- A <u>sum-of-products</u> (SOP) is a logical sum of product terms.
  Ex: Z' + W·X·Y.
- A <u>sum term</u> is a single literal or a logical sum of two or more literals. Ex: Z', W+X+Y, W+X'+Y'.
- A <u>product-of-sums</u> (POS) is a logical product of sum terms.
  Ex: Z' · (W+X+Y).

3. Boolean Algebra
- Standard Representation (3) -

- A <u>normal term</u> is a product or sum term in which no variable appears more than once.
  Examples (**non**-normal terms): W·X·X'·Z', W'+Y'+Z+W'.
- A n-variable <u>minterm</u> is a normal product term with n literals. Examples (with 4 variables): W·X·Y·Z', W'·X'·Y·Z.
- A n-variable <u>maxterm</u> is a normal sum term with n literals. Examples (with 4 variables): W+X+Y+Z', W'+X'+Y+Z.
- There is a correspondence between the truth table and minterms and maxterms.
- A minterm is a product term that is 1 in one row of the truth table.
- A maxterm is a sum term that is 0 in one row of the truth table.

3. Boolean Algebra
- Standard Representation (4) -

Minterms and maxterms for a 3-variable function F(X,Y,Z)

Row	X	Y	Z	F	Minterm	Maxterm
0	0	0	0	F(0,0,0)	X'·Y'·Z'	X + Y + Z
1	0	0	1	F(0,0,1)	X'·Y'·Z	X + Y + Z'
2	0	1	0	F(0,1,0)	X'·Y·Z'	X + Y'+ Z
3	0	1	1	F(0,1,1)	X'·Y·Z	X + Y'+ Z'
4	1	0	0	F(1,0,0)	X·Y'·Z'	X'+ Y + Z
5	1	0	1	F(1,0,1)	X·Y'·Z	X'+ Y + Z'
6	1	1	0	F(1,1,0)	X·Y·Z'	X'+ Y'+ Z
7	1	1	1	F(1,1,1)	X·Y·Z	X'+ Y'+ Z'

3. Boolean Algebra
- Standard Representation (5) -

- An n-variable minterm can be represented by an n-bit integer (the <u>minterm number</u>).
- In minterm i, a variable appears complemented if the respective bit in the binary representation of i is 0; otherwise it is uncomplemented.
- For example, row 5 (101) is related to minterm $X \cdot Y' \cdot Z$.
- In maxterm i, a variable appears complemented if the corresponding bit in the binary representation of i is 1; otherwise it is unprimed.
- For example, row 5 (101) is related to maxterm $X'+Y+Z'$.
- To specify the minterms and maxterms, it is mandatory to know the number of variables in the function and their order.

3. Boolean Algebra
- Standard Representation (6) -

Row	X	Y	Z	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

- Based on the correspondence between the truth table and the <u>minterms</u>, an algebraic representation of a logic function can be created.
- The <u>canonical sum</u> of a logic function is a sum of the minterms corresponding to truth table rows for which the function is 1.
- From the table:
  $F = \sum_{X,Y,Z}(0,3,4,6,7) = X' \cdot Y' \cdot Z' + X' \cdot Y \cdot Z + X \cdot Y' \cdot Z' + X \cdot Y \cdot Z' + X \cdot Y \cdot Z$
- The notation $\sum_{X,Y,Z}(0,3,4,6,7)$ is a <u>minterm list</u> and means the sum of minterms 0,3,4,6, and 7, with variables X, Y, and Z.
- The minterm list is also known as the <u>on-set</u> for the logic function.

3. Boolean Algebra
- Standard Representation (7) -

Row	X	Y	Z	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

- Based on the correspondence between the truth table and the <u>maxterms</u>, an algebraic representation of a logic function can be created.
- The <u>canonical product</u> of a function is a product of the maxterms corresponding to input combinations for which the function is 0.
- From the table:
  $F = \prod_{X,Y,Z}(1,2,5) = (X+Y+Z') \cdot (X+Y'+Z) \cdot (X'+Y+Z')$
- The notation $\prod_{X,Y,Z}(1,2,5)$ is a <u>maxterm list</u> and means the product of maxterms 1,2, and 5, with variables X, Y, and Z.
- The maxterm list is also known as the <u>off-set</u> for the logic function.

3. Boolean Algebra
- Standard Representation (8) -

- It is easy to convert between a minterm list and a maxterm list.
- For a function of n variables, the minterms and maxterms are in the set $\{0, 1, \ldots, 2^n-1\}$.
- A minterm or maxterm list contains a subset of these numbers.
- To switch between the lists, one takes the set complement.
- Examples:
  $\sum_{A,B,C}(0,1,2,3) = \prod_{A,B,C}(4,5,6,7)$
  $\sum_{X,Y}(1) = \prod_{X,Y}(0,2,3)$
  $\sum_{W,X,Y,Z}(1,2,3,5,8,12,13) = \prod_{W,X,Y,Z}(0,4,6,7,9,10,11,14,15)$

3. Boolean Algebra
- Standard Representation (9) -

- We have learned 5 possible representations for a combinational logic function.
  - **A truth table;**
  - **An algebraic sum of minterms (the canonical sum);**
  - **A minterm list, using the $\sum$ notation;**
  - **An algebraic product of maxterms (the canonical product);**
  - **A maxterm list, using the $\prod$ notation;**
- Each one of these representations specifies exactly the same information.
- Given any of them, we can derive the other four using a simple mechanical process.

3. Boolean Algebra
- Examples (1) -

- Ex.1: Let $F = X \cdot Y + X \cdot Y' \cdot Z + X' \cdot Y \cdot Z$. Derive the expression for F' in the product of sums form.
- $F' = (XY + XY'Z + X'YZ)'$
  $= (XY)' \cdot (XY'Z)' \cdot (X'YZ)'$
  $= (X'+Y')(X'+Y+Z')(X+Y'+Z')$
- Ex.2: Express the function $G(X,Y,Z) = X + Y \cdot Z$ as a sum of minterms.
- $G = X + Y \cdot Z$
  $= X \cdot (Y+Y') \cdot (Z+Z') + Y \cdot Z \cdot (X+X')$
  $= XYZ + XYZ' + XY'Z + XY'Z' + XYZ + X'YZ$
  $= X'YZ + XY'Z' + XY'Z + XYZ' + XYZ$
  $= \sum_{X,Y,Z}(3,4,5,6,7)$

3. Boolean Algebra

- Examples (2) -

- Ex.3: Derive the product-of-maxterms form for H = X’·Y’ + X·Z.
- **H = X’Y’ + XZ**
  = (X’Y’+X)(X’Y’+Z)
  = (X’+X)(Y’+X)(X’+Z)(Y’+Z)
  = (X+Y’)(X’+Z)(Y’+Z)
- Each OR term in the expression is missing one variable:
  - **X+Y’ = X+Y’+ZZ’ = (X+Y’+Z)(X+Y’+Z’)**
  - **X’+Z = X’+Z+YY’ = (X’+Y+Z)(X’+Y’+Z)**
  - **Y’+Z = Y’+Z+XX’ = (X+Y’+Z)(X’+Y’+Z)**
- Finally we combine these terms:
  H = (X+Y’+Z)(X+Y’+Z’)(X’+Y+Z)(X’+Y’+Z)
  $\prod_{X,Y,Z}(2,3,4,6)$

3. Boolean Algebra

- Examples (3) -

- Ex.4: Derive the product-of-maxterms form for H = X’·Y’ + X·Z.

X	Y	Z	H
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

- From the table, we obtain:
  H = $\prod_{X,Y,Z}(2,3,4,6)$
  H = $\sum_{X,Y,Z}(0,1,5,7)$
- Compare this solution with the solution of ex.3.

- Ex.5: Derive a standard form with a reduced number of operators for J = XYZ + XYZ’ + XY’Z + X’YZ.
- **J = XYZ + XYZ’ + XYZ + XY’Z + XYZ + X’YZ**
  = XY(Z+Z’) + X(Y+Y’)Z + (X+X’)YZ
  = XY+XZ+YZ