SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Boolean Expressions and Operators

Prof. Daniele Gorla

Boolean Expressions

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

A *boolean expression* is a sequence composed of constants, variables, parenthesis and operators, inductively defined as follows:

Let V be a numerable set of variables; then

- $0, 1 \in \mathrm{BE}$;
- if $x \in V$, then $x \in \mathrm{BE}$;
- if $\mathrm{E} \in \mathrm{BE}$, then $\overline{\mathrm{E}}$, $(\mathrm{E}) \in \mathrm{BE}$;
- if $\mathrm{E}_1, \mathrm{E}_2 \in \mathrm{BE}$, then $\mathrm{E}_1+\mathrm{E}_2$, $\mathrm{E}_1 \cdot \mathrm{E}_2 \in \mathrm{BE}$.

2

Dual and Complementary Expressions

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Dual Expression: obtained by swapping 0 and 1 , + and ·

Complementary Expression: like the dual one, but also complement all variables (it is obtained through De Morgan and involution)

Example: $\mathrm{E} = (x+0) \cdot y + 1 \cdot z$

Dual:

$$(x \cdot 1 + y) \cdot (0 + z)$$

Complementary:

$$\overline{(x+0)\cdot y + 1\cdot z} = \overline{(x+0)\cdot y} \cdot \overline{1\cdot z} = (\overline{x+0} + \overline{y})\cdot(\overline{1} + \overline{z})$$

$$= (\overline{x}\cdot\overline{0} + \overline{y})\cdot(0+\overline{z}) = (\overline{x}\cdot 1 + \overline{y})\cdot(0+\overline{z})$$

3

Equivalence of Boolean Expressions

SAPIENZA UNIVERSITÀ DI ROMA DIPARTIMENTO DI INFORMATICA

Def.: E_1 and E_2 are *equivalent* if they have the same value under the same assignment of boolean values to their variables.

Check: 1. through formal proofs
2. through perfect induction
   - Consider all possible assignments to variables
   - Incrementally compute the value of the expression for every assignment

Example: $x + xy = x + xz$

- $x + xy = x\,(1 + y) = x = x\,(1 + z) = x + xz$
-

x y z	xy	$x+xy$	xz	$x+xz$
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	0	0	0
0 1 1	0	0	0	0
1 0 0	0	1	0	1
1 0 1	0	1	1	1
1 1 0	1	1	0	1
1 1 1	1	1	1	1

4

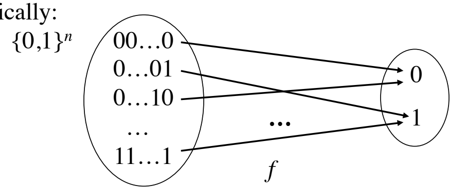
Boolean Functions

Hence, a BE identifies a *boolean function*, i.e. a law that, according to the variable values, univoquely returns a boolean value:

$$f : \{0,1\}^n \rightarrow \{0,1\}$$

numero di variabili

Graphically:



Remark: two BEs are equivalent if they identify the same BF

Truth Tables

SAPIENZA Università di Roma Dipartimento di Informatica

A *boolean function* can be represented through a **truth table** that fully describes the association between the elements of the domain and those of the codoman.

Given n variables, a truth table is made up of 2 parts:

- In the leftmost part, there are all the 2^n possible combinations of boolean values assignable to variables
- In the rightmost part, there is a column of 0s and 1s such that the value in the row i is the value of the i-th n-tupla of boolean values assignable to the variables.

Example (function associated to the BE $x \cdot y$):

x	y	f
0	0	0
0	1	0
1	0	0
1	1	1

Constant and Unary BFs

SAPIENZA Università di Roma Dipartimento di Informatica

$$f : \{0,1\}^n \rightarrow \{0,1\}$$

If $n = 0$, f is a constant, that can be either 0 or 1

If $n = 1$, we have four possible BFs:

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1
	constant0	identity	complement (NOT)	constant1

Binary BFs

SAPIENZA Università di Roma Dipartimento di Informatica

If $n = 2$, we have 16 possible functions:

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
		constant0	• (AND)		x		y	XOR $\oplus$	+ (OR)	NOR	XNOR	$\overline{y}$		$\overline{x}$		NAND	constant1

Functions to $\{0,1\}^m$

In general, a BF can return an m-tuple of bits:

$$f : \{0,1\}^n \rightarrow \{0,1\}^m$$

Ex.:

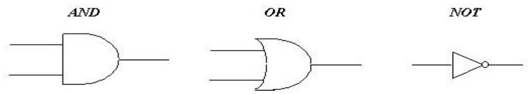
x y	f
0 0	0 0 0
0 1	1 0 0
1 0	0 1 1
1 1	1 0 0

From now on, we shall consider such a function as an m-tupla of functions with codomain $\{0,1\}$.

Ex.:

x y	f_1	f_2	f_3
0 0	0	0	0
0 1	1	0	0
1 0	0	1	1
1 1	1	0	0

Basic logic gates



AND

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

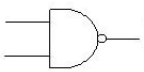
NOT

x	y
0	1
1	0

Other logic gates

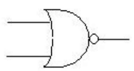
NAND

x	y	z
0	0	1
0	1	1
1	0	1
1	1	0



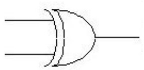
NOR

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0



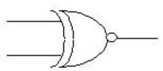
XOR

x	y	z
0	0	0
0	1	1
1	0	1
1	1	0



XNOR

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1

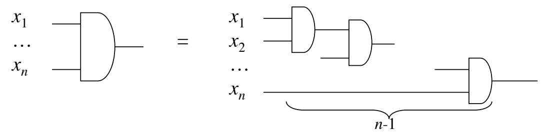


$OBS.: x \oplus y = \overline{x} \cdot y + x \cdot \overline{y}$

$OBS.: \overline{x \oplus y} = \overline{x} \cdot \overline{y} + x \cdot y$

Multiple inputs Gates

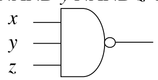
$x_1 \cdot \ldots \cdot x_n = (\ldots(x_1 \cdot x_2) \cdot \ldots \cdot x_n)$ (Associativity)



The same happes for OR, that implements an associative operator (+).

What happens for NAND, NOR, XOR and XNOR?

- for XOR and XNOR, the situation is similar (they're associative)
- for NAND and NOR, the situation changes: since they're NOT associative, writing x NAND y NAND z is meaningless. Hence, with



we denote a specific 3 input gate, not realizable with two 2-inputs NAND gates one after the other.

Multiple inputs Gates (cont.)

XOR is associative:

$$(x\oplus y)\oplus z = \overline{(x\oplus y)}\cdot z + (x\oplus y)\cdot \overline{z} = (x\cdot y + \overline{x}\cdot \overline{y})\cdot z + (x\cdot \overline{y} + \overline{x}\cdot y)\cdot \overline{z}$$

$$= x\cdot y\cdot z + \overline{x}\cdot \overline{y}\cdot z + x\cdot \overline{y}\cdot \overline{z} + \overline{x}\cdot y\cdot \overline{z} =$$

$$= \overline{x}\cdot(y\cdot \overline{z} + \overline{y}\cdot z) + x\cdot(y\cdot z + \overline{y}\cdot \overline{z}) = \overline{x}\cdot(y\oplus z) + x\cdot\overline{(y\oplus z)} = x\oplus(y\oplus z)$$

Similarly, you can prove that XNOR is associative.

By contrast, NOR and NAND are not!

Ex. (NAND): $\overline{x\cdot\overline{(y\cdot z)}} = \overline{x} + y\cdot z$ $\qquad$ $\overline{\overline{(x\cdot y)}\cdot z} = x\cdot y + \overline{z}$

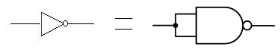
These two BEs are not equivalent:
consider the assignment $x = y = 0$ and $z = 1$, that makes 1 the first BE and 0 the second one.

13

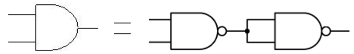
Universality of NAND gates

SAPIENZA
Università di Roma
Dipartimento di Informatica

By idempotency, $x = x\cdot x \Rightarrow \overline{x} = \overline{x\cdot x} = NAND(x,x)$

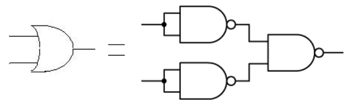


By involution, $x\cdot y = \overline{\overline{x\cdot y}} = \overline{NAND(x,y)} = NAND(NAND(x,y),NAND(x,y))$



By involution and De Morgan,

$$x + y = \overline{\overline{x+y}} = \overline{\overline{x}\cdot\overline{y}} = NAND(\overline{x},\overline{y}) = NAND(NAND(x,x),NAND(y,y))$$



14

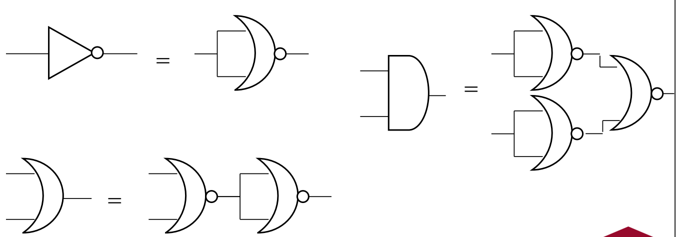
Universality of NOR gates

SAPIENZA
Università di Roma
Dipartimento di Informatica

By duality, we have that

$$\overline{x} = \overline{x+x} \qquad x + y = \overline{\overline{x+y}} \qquad x\cdot y = \overline{\overline{x}+\overline{y}}$$

and so



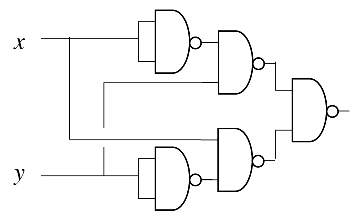
15

All-NAND circuits (example)

SAPIENZA
Università di Roma
Dipartimento di Informatica

Let us implement a XOR gate by only using NAND gates:

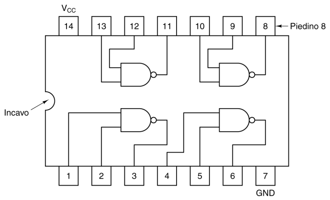
$$x\oplus y = x\cdot\overline{y} + \overline{x}\cdot y = \overline{\overline{(x\cdot\overline{y})}\cdot\overline{(\overline{x}\cdot y)}} = \overline{\overline{(x\cdot\overline{y\cdot y})}\cdot\overline{(\overline{x\cdot x}\cdot y)}}$$

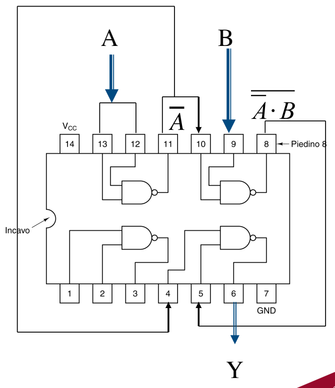


16

Practical Usage: integrated circuits

SAPIENZA
Università di Roma
Dipartimento di Informatica





$$Y = A + \bar{A}B$$

$$= \overline{\bar{A}\cdot\overline{(\bar{A}\cdot B)}} = \overline{\overline{A\cdot A}\cdot\overline{(\overline{A\cdot A}\cdot B)}}$$

From a SOP expression to an ALL-NAND one

SAPIENZA
Università di Roma
Dipartimento di Informatica

Given a SOP BE (a Sum Of Products of variables and negated variables), it is very easy to built an equivalent ALL-NAND BE (by assuming the use of multiple inputs NAND and NOR gates):

1. Apply De Morgan to the disjunction (outmost operator)
   - This turns the outmost OR into a negated AND, and
   - all the conjunctions among the variables into many NANDs
2. Finally, we have to replace the negated variables with self NANDs

Ex. (previous slide):

$$x\cdot\bar{y} + \bar{x}\cdot y = \overline{\overline{(x\cdot\bar{y})}\cdot\overline{(\bar{x}\cdot y)}} = \overline{\overline{(x\cdot\overline{y\cdot y})}\cdot\overline{(\overline{x\cdot x}\cdot y)}}$$

From a POS expression to an ALL-NOR one

SAPIENZA
Università di Roma
Dipartimento di Informatica

Dually, given a POS (product of sums of variables and negated variables), it is easy to build an equivalent ALL-NOR BE (again, having multiple inputs NOR gates):

1. Apply De Morgan to the conjunction (outmost operator)
   - This turns the AND into a negated OR, and
   - the disjunctions among variables into NORs
2. Finally, we have to replace the negated variables with self NORs

ES.:

$$(x+\bar{y}+z)\cdot(\bar{x}+y) = \overline{\overline{(x+\bar{y}+z)}+\overline{(\bar{x}+y)}} = \overline{\overline{(x+\overline{y+y}+z)}+\overline{(\overline{x+x}+y)}}$$

Remark: here we're using a 3-inputs NOR!!