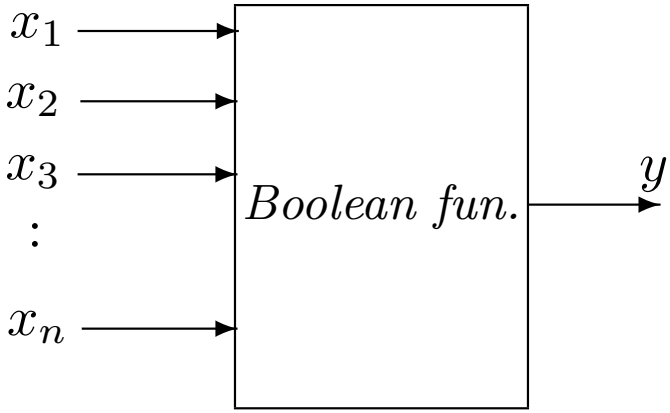
Digital systems and basics of electronics

Adam Szmigielski
aszmigie@pjwstk.edu.pl
materials: $ftp(public) : //aszmigie/SYC/ENG$

Minimization of Boolean functions - lecture 7

Boolean function



- *Boolean n argument function* is an transformation $f : B^n \to B$, where $B = \{0, 1\}$ is an set of function values,
- *Boolean function* is an model of *combinational circuit.*

Description of Boolean functions - truth tables

- One variable function (eg. negation $f(a) = \neg a$)

a	f(a)
0	1
1	0

- Two variable function (eg. conjunction $f(a, b) = a \wedge b$)

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

Disjunctive Normal Form (DNF)

a	b	$f(a,b)$
0	0	0
0	1	0
1	0	0
1	1	1

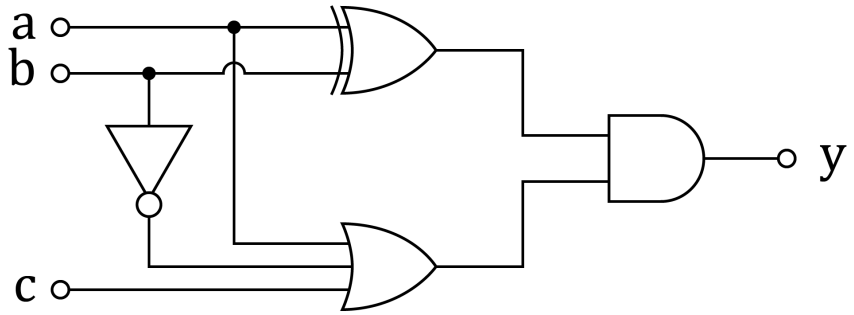
- Disjunctive Normal Form (DNF) is a disjunction, where a clause is a conjunctive of literals. : function f is a sum of products $f = \ldots(\ldots \wedge \ldots \wedge \ldots) \vee (\ldots \wedge \ldots \wedge \ldots) \vee (\ldots \wedge \ldots \wedge \ldots)\ldots$,
- Expression in brocket (product) corresponds to one one ("1")
- In our case: $f = (a \wedge b)$,
- Decimal description: $f(a,b) = \sum\{3\}$.

Conjunctive Normal Form (CNF)

a	b	$f(a,b)$
0	0	0
0	1	0
1	0	0
1	1	1

- Conjunctive normal form (CNF) is a conjunction of clauses, where a clause is a disjunction of literals.: function is a product of sums $f = \ldots(\ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots)\ldots$,
- Expression in brocket (sum) corresponds to one one ("0"),
- In our case: $f = (a \vee b) \wedge (a \vee \overline{b}) \wedge (\overline{a} \vee b)$.
- Decimal description: $f(a,b) = \prod\{0,1,2\}$

Diagram of logic circuit



1. Logic circuit describes structure of Boolean function,
2. Information flow: From INPUT to OUTPUT , eg. $y = f(a, b, c)$,
3. Dot means electrical connection,
4. In our case diagram realizes function:

$$y = f(a, b, c) = (a \oplus b) \cdot (a + \overline{b} + c)$$

Realization based on truth table

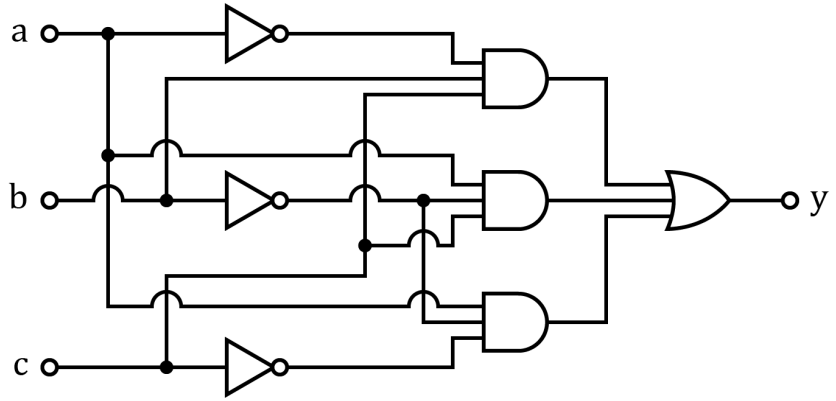
a	b	c	$y = f(a, b, c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

- DNF - (We are looking for "1" on output.):

$$y = f(a, b, c) = \overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c$$

Realization of Boolean function with logic gates

a	b	c	$y = f(a, b, c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0



- $y = f(a, b, c) = \overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c$
- Is it possible to reduce the number of logic gates ?

Translation of Boolean function

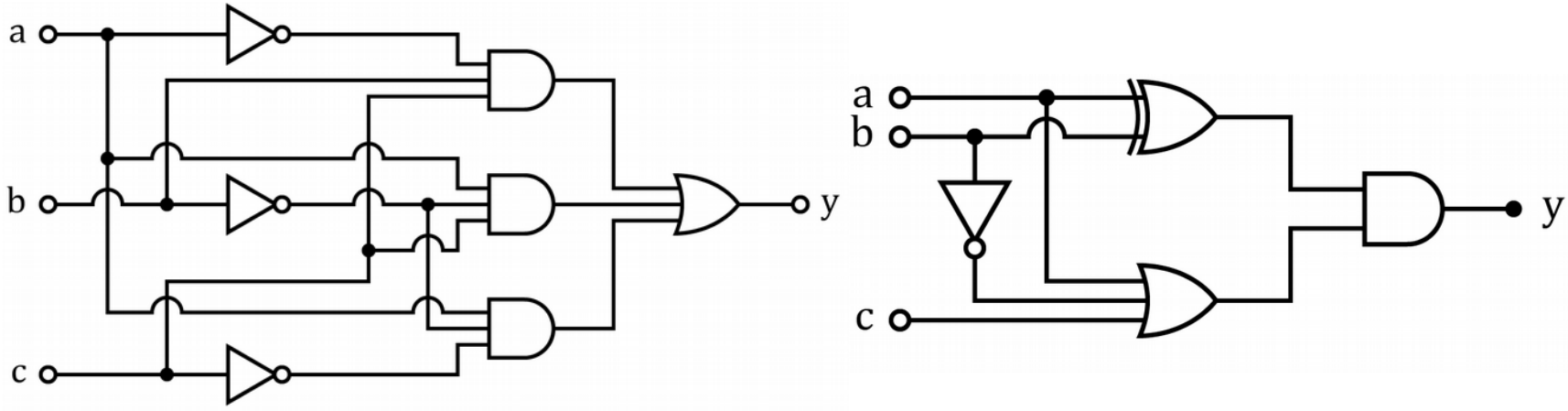
1. $y = f(a, b, c) = \overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c$
2. $\overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c = \overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c + a\overline{b}c$
3. $\overline{a}bc + a\overline{b}\overline{c} + a\overline{b}\overline{c} + a\overline{b}c = \overline{a}bc + a\overline{b}(\overline{c} + c) + a\overline{b}c = \overline{a}bc + a\overline{b} + a\overline{b}c$
4. $\overline{a}bc + a\overline{b} + a\overline{b}c = \overline{a}bc + aa\overline{b} + a\overline{b}\overline{b} + a\overline{b}c$
5. $\overline{a}bc + aa\overline{b} + a\overline{b}\overline{b} + a\overline{b}c = \overline{a}bc + aa\overline{b} + a\overline{b}\overline{b} + a\overline{b}c + a\overline{a}b + ab\overline{b}$
6. $\overline{a}bc + aa\overline{b} + a\overline{b}\overline{b} + a\overline{b}c + a\overline{a}b + ab\overline{b} = a\overline{b}(a + \overline{b} + c) + \overline{a}b(a + \overline{b} + c)$
7. $a\overline{b}(a + \overline{b} + c) + \overline{a}b(a + \overline{b} + c) = (a\overline{b} + \overline{a}b)(a + \overline{b} + c)$

Equivalence of Boolean functions

- If Boolean functions can be equivalent each other

  $\overline{a}bc + a\overline{b}\overline{c} + a\overline{b}c \Leftrightarrow (a\overline{b} + \overline{a}b)(a + \overline{b} + c)$

- then theirs realizations are also equivalent.



Optimization of Boolean function task

Reduction of combinational circuits cost can be done in several ways:

- By reducing the number of logic gates,
- By reducing the number of gate's inputs,
- By reducing the variety of gates,
- By reducing the designing time.

Reduction of variety of logic gates

What is the smallest number of logic gate types ?

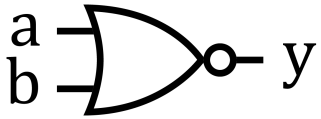
Classical logic (investigates the operation: conjunction $\wedge$, alternative $\vee$, implication $\Rightarrow$ and negation $\neg$) is redundant. It means it is possible to define operator using the rest operators. The smallest systems need:

- Implicative-negative - using negation and implication,
- Conjunctive-negative - using conjunction and negation,
- Alternative-negative - using alternative and negation.

Universal gates - NOR and NAND

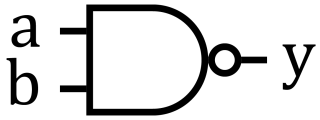
- Using NOR (negated alternative) gates any Boolean function can be realized,

a	b	NOR(a,b)
0	0	1
0	1	0
1	0	0
1	1	0



- Using NAND (negated conjunction) gates any Boolean function can be realized,

a	b	NAND(a,b)
0	0	1
0	1	1
1	0	1
1	1	0



Graya code

000
001
011
010
110
111
101
100

The reflected binary code, also known as Gray code after Frank Gray, is a binary numeral system where two successive values differ in only one bit. Gray code is also periodical, first and last elements also meet the one bit difference condition.

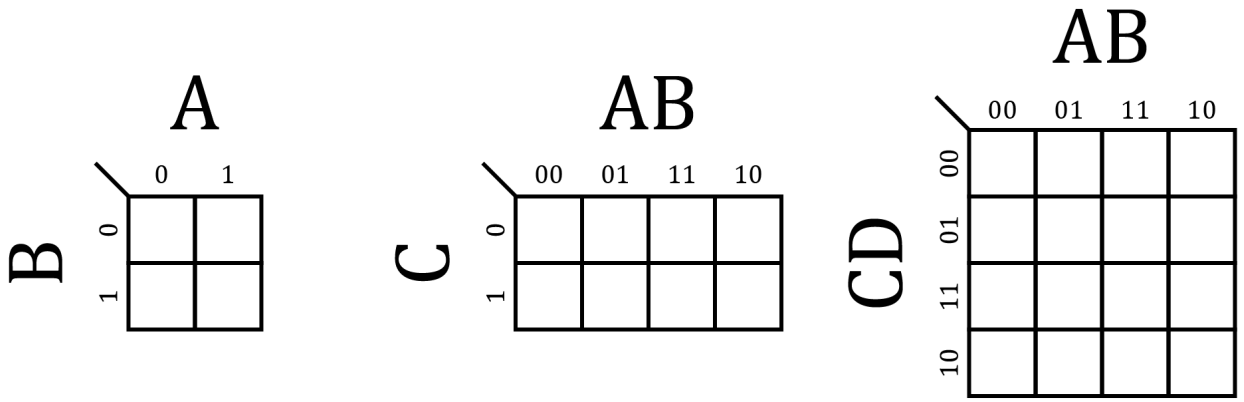
"Grouping rule" and Gray code

- "Grouping rule": $a \cdot f(x_1, x_2, \ldots x_n) + \overline{a} \cdot f(x_1, x_2, \ldots x_n) = (a + \overline{a}) \cdot f(x_1, x_2, \ldots x_n) = f(x_1, x_2, \ldots x_n)$

000	$\overline{a}\overline{b}\overline{c}$
001	$\overline{a}\overline{b}c$
011	$\overline{a}bc$
010	$\overline{a}b\overline{c}$
110	$ab\overline{c}$
111	abc
101	$a\overline{b}c$
100	$a\overline{b}\overline{c}$

- Two successive values can be replaced with one, skipping the bit, which changed, eg expression $\overline{a}bc + \overline{a}b\overline{c}$ is equivalent $\overline{a}b$.

Karnaugh map



- Karnaugh map is a tool to facilitate the simplification of Boolean algebra integrated circuit expressions.
- Karnaugh map is filled due to truth table,
- Variables (in rows and columns) are ordered due to Gray code.

Karnaugh map

a	b	c	$y = f(a, b, c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

and

abc	abc	
000	$\overline{a}\overline{b}\overline{c}$	0
001	$\overline{a}\overline{b}c$	0
011	$\overline{a}bc$	1
010	$\overline{a}b\overline{c}$	0
110	$ab\overline{c}$	0
111	abc	0
101	$a\overline{b}c$	1
100	$a\overline{b}\overline{c}$	1

.

Another realization of Karnaugh map

abc	
$\overline{a}\overline{b}\overline{c}$	0
$\overline{a}\overline{b}c$	0
$\overline{a}bc$	1
$\overline{a}b\overline{c}$	0
$ab\overline{c}$	0
abc	0
$a\overline{b}c$	1
$a\overline{b}\overline{c}$	1

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

$ab\backslash c$	0	1
00	0	0
01	0	1
11	0	0
10	1	1

Karnaugh map - grouping “1”

abc	
$\overline{a}\overline{b}\overline{c}$	0
$\overline{a}\overline{b}c$	0
$\overline{a}bc$	1
$\overline{a}b\overline{c}$	0
$ab\overline{c}$	0
abc	0
$a\overline{b}c$	1
$a\overline{b}\overline{c}$	1

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

$ab\backslash c$	0	1
00	0	0
01	0	1
11	0	0
10	1	1

- We group ones “1” vertically or horizontally in quantity of power of 2 obtaining DNF format.
- We erase that bits which changed,
- Minimal DNF: $y = a\overline{b} + \overline{a}bc$.

Karnaugh map - grouping “0”

abc	
$\overline{a}\overline{b}\overline{c}$	0
$\overline{a}\overline{b}c$	0
$\overline{a}bc$	1
$\overline{a}b\overline{c}$	0
$ab\overline{c}$	0
abc	0
$a\overline{b}c$	1
$a\overline{b}\overline{c}$	1

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

$ab\backslash c$	0	1
00	0	0
01	0	1
11	0	0
10	1	1

- We group zeros “0” vertically or horizontally in quantity of power of 2 obtaining CNF format.
- We erase that bits which changed, and the **rest bits must be negated**,
- Minimal CNF: $y = (a + b) \cdot (\overline{b} + c) \cdot (\overline{a} + \overline{b})$.

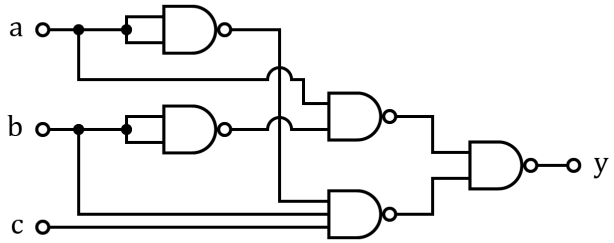
Equivalence of DNF and CNF

- Both, CNF and DNF are equivalence,

$$a\overline{b} + \overline{a}bc \Leftrightarrow (a+b)\cdot(\overline{b}+c)\cdot(\overline{a}+\overline{b})$$

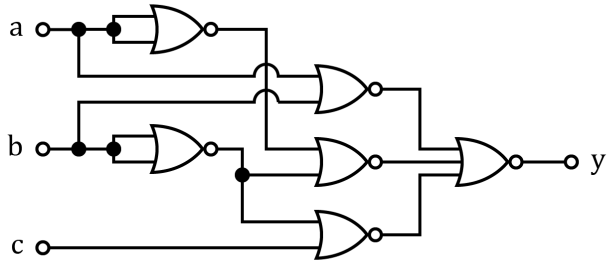
- motivation:
- $(a+b)\cdot(\overline{b}+c)\cdot(\overline{a}+\overline{b}) \Leftrightarrow (a\overline{b}+ac+b\overline{b}+bc)\cdot(\overline{a}+\overline{b})$
- $(a\overline{b}+ac+b\overline{b}+bc)\cdot(\overline{a}+\overline{b}) \Leftrightarrow a\overline{a}\overline{b}+a\overline{a}c+\overline{a}bc+a\overline{b}\overline{b}+a\overline{b}c+b\overline{b}c$
- $a\overline{a}\overline{b}+a\overline{a}c+\overline{a}bc+a\overline{b}\overline{b}+a\overline{b}c+b\overline{b}c \Leftrightarrow \overline{a}bc+a\overline{b}+a\overline{b}c$
- $\overline{a}bc+a\overline{b}+a\overline{b}c \Leftrightarrow a\overline{b}+\overline{a}bc$

Realization of DNF using NAND gates



- Function in DNF format $y = a\overline{b} + \overline{a}bc$ double negate.
- $y = \overline{\overline{a\overline{b} + \overline{a}bc}}$. For internal negation use de Morgan law.
- $y = \overline{\overline{a\overline{b}} \cdot \overline{\overline{a}bc}}$

Realization of CNF using NOR gates



- Function in CNF format $y = (a + b) \cdot (\overline{b} + c) \cdot (\overline{a} + \overline{b})$ double negate.
- $y = \overline{\overline{(a + b) \cdot (\overline{b} + c) \cdot (\overline{a} + \overline{b})}}$. For internal negation use de Morgan law.
- $y = \overline{\overline{a + b} + \overline{\overline{b} + c} + \overline{\overline{a} + \overline{b}}}$

Tasks for laboratory

1. For individual student following function is given $y = \sum(\ldots\ldots\ldots\ldots)$. Using NAND gates realize given function.
2. Using only NOR gates realize control circuit for mobile robot. Robot should move without any collision. Robot has 3 distance sensor located in front of, left and right side. When obstacle is detected sensor gives logical "1", otherwise gives "0". The moving platform uses differential drive. Engine is on when logical "1ćontrol signal is applied. When robot could not continue movement should stop.