DIGITAL TECHNICS I

Dr. Bálint Pődör

Óbuda University, Microelectronics and Technology Institute

6. LECTURE:
COMBINATIONAL CIRCUITS, HAZARDS

1st year BSc course 1st (Autumn) term 2018/2019

1

6. LECTURE

1. Some aspects of combinational circuits : symmetric functions and XOR logic.

2. Some aspects of combinational circuits : multiple output networks.

3. Hazards in combinational logic circuits.

2

SYMMETRIC BOOLEAN FUNCTIONS

If the variables of a function can be interchanged with each other (permuted) without changing the value of the function, the its called symmetric function.

Examples of symmetric function: XOR, XNOR, sum function of the full adder, $S_i = A_i \oplus B_i \oplus C_{i-1}$, etc.

E.g. for n=3 (A,B,C) if A and B can be interchanged with each other, but neither of them with C, the function is partially symmetric with respect of the pair of variables, A and B.

The symmetry therefore can be full or partial.

3

EXCLUSIVE OR LOGIC

The symmetric functions have special characteristics, like they form a ”chessboard” pattern on the Karnaugh map (at least partially), and they can be simplified by using XOR functions as functional elements.

Reduction of a function to XOR form is characterized by a Karnaugh map where the 1s are diagonally opposite to each other.

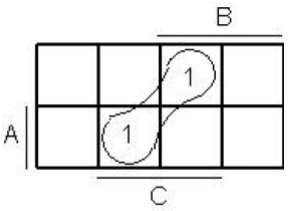
In the general context of minimization of Boolean functions XOR gates can, for certain problems, provide a more economic implementation than by using other logic gate s.

Two examples are the 1-bit full adder, and the binary-to-Gray code conversion.

4

UTILIZATION OF SYMMETRY: EXAMPLE

$F^3(A,B,C)=\sum(3,5)$



XOR, XNOR
Look for "checkerboard" squares
Depends whether XOR/XNOR gates available

$F^3=(A,B,C) = \bar{A}BC + A\bar{B}C = C \,\&\, (A\bar{B} + \bar{A}B) = C \,\&\, (A \oplus B)$

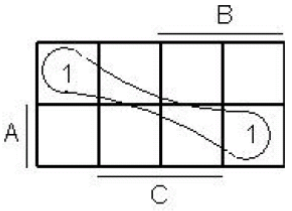
AND-OR (NAND-NAND) implementation: 8 pins 3 gates

Implementation using XOR: 4 pins 2 gates

Symmetry: partial, with respect to A and B

5

UTILIZATION OF SYMMETRY: EXAMPLE



$F^3(A,B,C) = \bar{A}\bar{B}\bar{C} + AB\bar{C} = \bar{C} \,\&\, (AB + \bar{A}\bar{B})$ $= C\,(\bar{A} \oplus B)$

AND-OR (NAND-NAND) implementation: 8 pins 3 gates

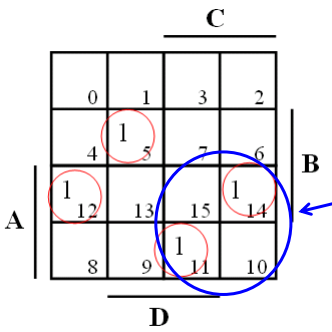
Implementation using XOR: 4 pins 2 gates

Symmetry: partial, with respect to A and B

6

UTILIZATION OF SYMMETRY: EXAMPLE

$$F(A,B,C,D) = B\overline{C}(A \oplus D) + AC(B \oplus D)$$



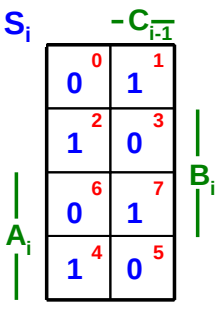
Chessboard pattern

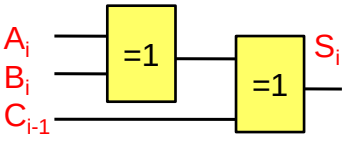
7

OUTLOOK: S_i (SUM) FUNCTION OF THE 1-BIT FULL ADDER

„chessboard pattern”
symmetric function

i	(4) A_i	(2) B_i	(1) C_{i-1}	S_i
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1





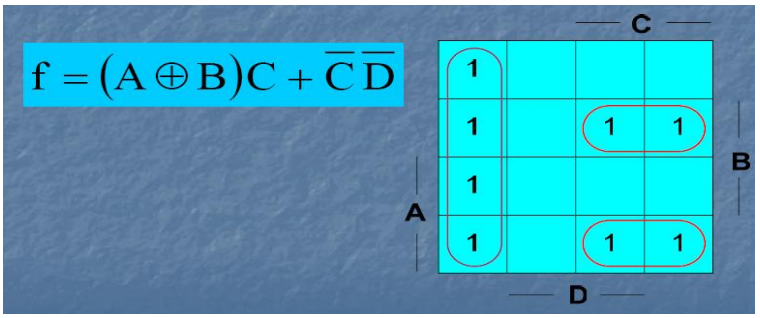
A_i, B_i operands i-th position
C_{i-1} carry-in from (i-1)-th position

Sum function $\quad S_i = A_i \oplus B_i \oplus C_{i-1}$

8

EXAMPLE: PARTIALLY SYMMETRIC FUNCTIONS

AND-OR: $F = A\overline{B}C + \overline{A}BC + \overline{C}\,\overline{D}$



Chessboard pattern (right half of the map)

Pin counts:

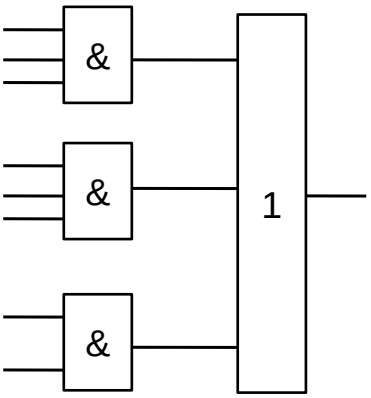
AND/OR implementation (2 level): 11

AND/OR/XOR implementation (3 level): 8

9

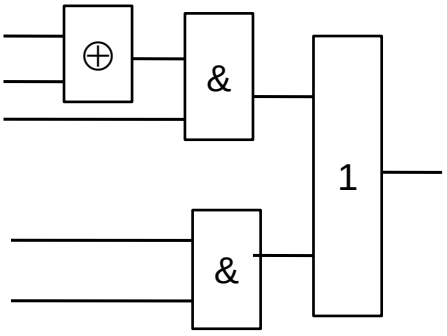
POSSIBLE IMPLEMENTATIONS

AND-OR



$F = A\overline{B}C + \overline{A}BC + \overline{C}\,\overline{D}$

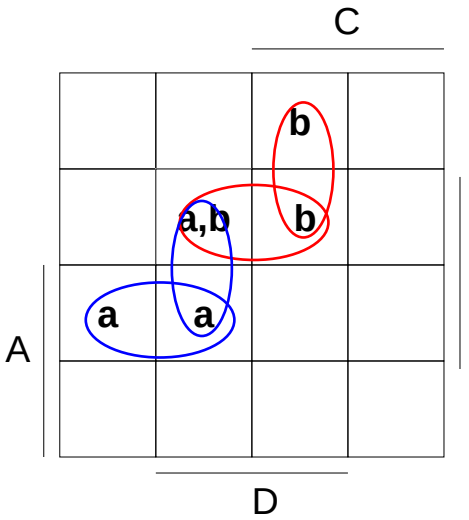
AND-OR-XOR



$F = (A \oplus B)C + \overline{C}\,\overline{D}$

10

MINIMIZATION AND IMPLEMENTATION OF MULTIPLE OUTPUT NETWORKS



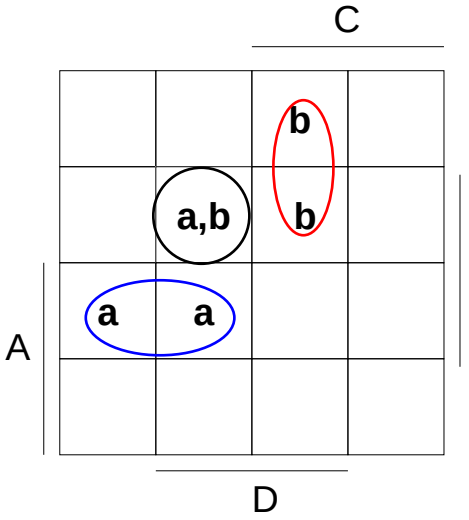
$Fa = \Sigma^4(5,12,13)$

$Fb = \Sigma^4(3,5,7)$

”Elementary” implementation: four 3-input AND gates and two 2-input OR gates

Cost function (pin count): 4x3 + 2x2 = 16

MINIMIZATION AND IMPLEMENTATION OF MULTIPLE OUTPUT NETWORKS

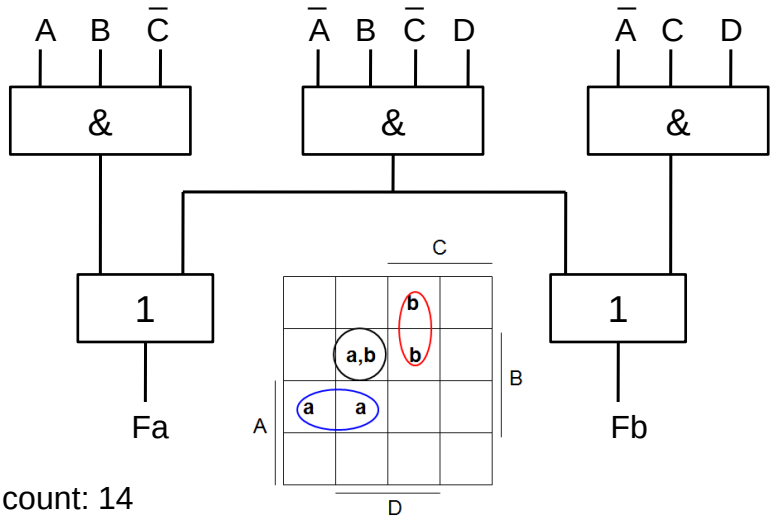


$Fa = \Sigma^4(5,12,13)$

$Fb = \Sigma^4(3,5,7)$

Utilization of common implicants and prime implicants

MINIMIZATION AND IMPLEMENTATION OF MULTIPLE OUTPUT NETWORKS



Pin count: 14

EXAMPLE: MINIMIZATION OF THREE OUTPUT FUNCTION

Determine the simplest conceptual two-level AND-OR logic diagram of the three output logic network:

$Fa = \Sigma^4(0,1,5,6,7,13)$

$Fb = \Sigma^4(0,1,5,10\text{-}15)$

$Fc = \Sigma^4(0,1,8\text{-}11,14,15)$

The common prime implicants of Fa and Fb are the prime implicants of the product function Fab = FaFb, etc.

COMMON (PRIME) IMPLICANTS

Product functions (pairs):

Fa = Σ^4(0,1,5,6,7,13)

Fb = Σ^4(0,1,5,10-15)

Fc = Σ^4(0,1,8-11,14,15)

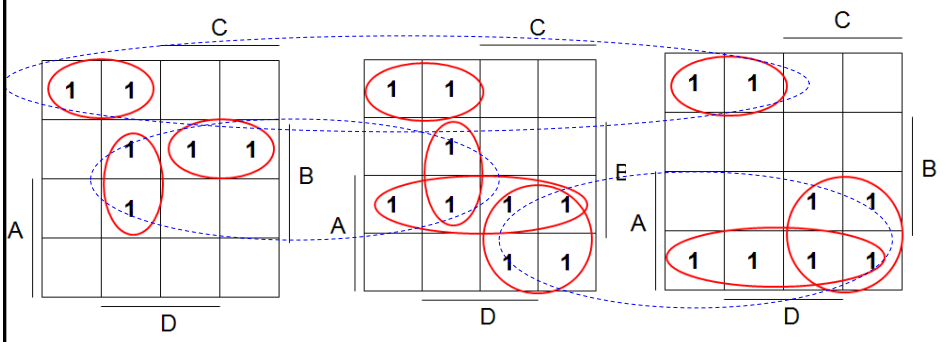
Fab = FaFb = Σ^4(0,1,5,13) = m(0,1) + m(5,13)

Fbc = FbFc = Σ^4(0,1,10,11,14,15) = (0,1) + m(10,11,14,15)

Fca = FcFa = Σ^4(0,1) = m(0,1)

15

RESULT OF MINIMIZATION



Principle: the common prime implicants occurring in more outputs are implemented only once.

Fa,Fb,Fc:	/A /B /C	m(0,1)
Fa,Fb:	B /C D	m(5,13)
Fa,Fb:	A C	m(10,11,14,15)

16

RESULT OF MINIMIZATION

a,b,c	a,b,c		
	a,b	a	a
b	a,b	b,c	b,c
c	c	b,c	b,c

C

B

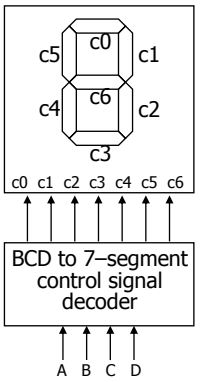
A

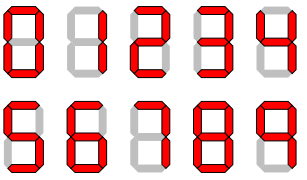
D

17

ANOTHER DEMO EXAMPLE: BCD TO 7-SEGMENT DISPLAY CONTROLLER

- Understanding the problem
  - Input is a 4 bit BCD digit (A, B, C, D)
  - Output is the control signals for the display (7 outputs C0 – C6)
- Block diagram



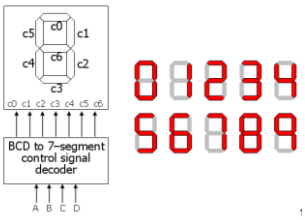


18

FORMALIZE THE PROBLEM

- Truth table
  - Show don't cares
- Choose implementation target
  - If ROM, we are done
  - Don't cares imply PAL/PLA may be attractive
- Follow implementation procedure
  - Minimization using K-maps

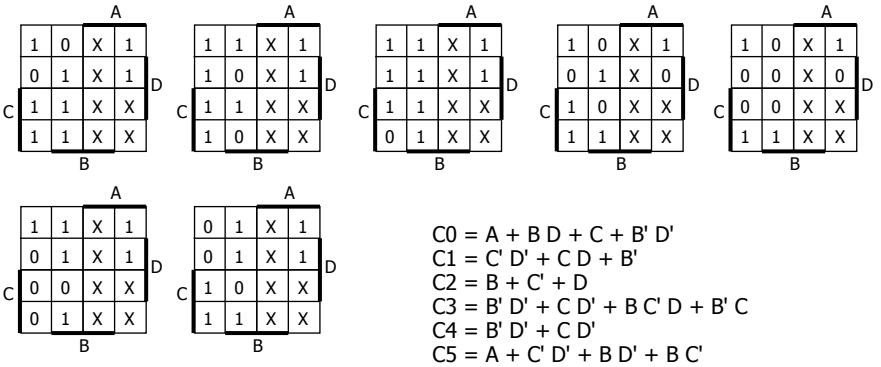
A	B	C	D	C0	C1	C2	C3	C4	C5	C6
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	–	–	–	–	–	–	–	–
1	1	–	–	–	–	–	–	–	–	–



19

IMPLEMENTATION AS MINOMIZED SUM-OF-PRODUCTS (SOP)

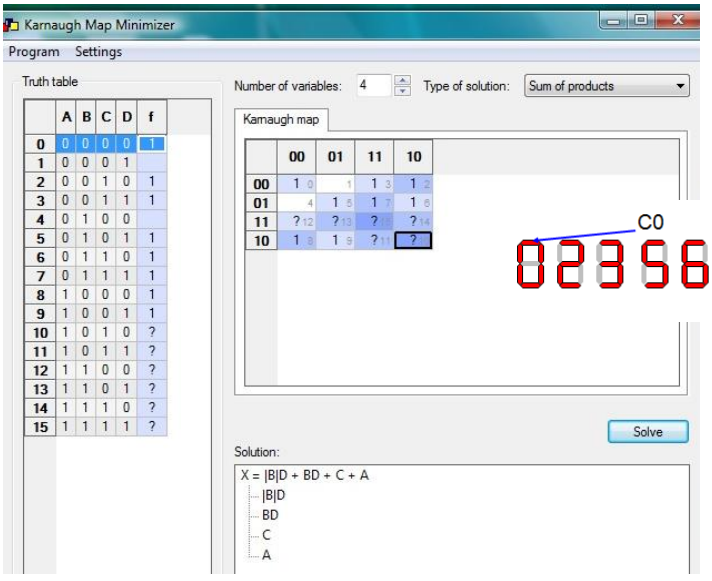
- 15 unique product terms when minimized individually



C0 = A + B D + C + B' D'
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B C'
C6 = A + C D' + B C' + B' C

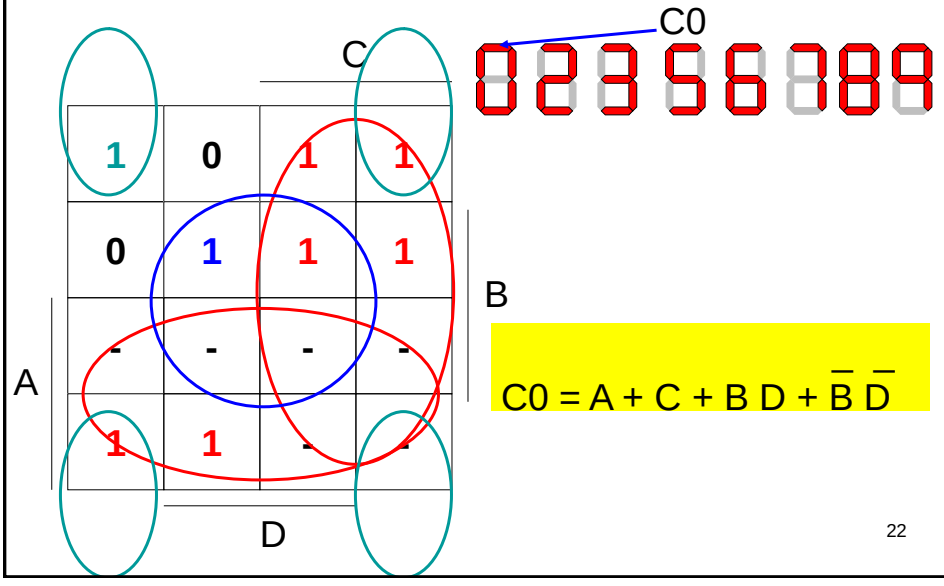
20

EXAMPLE: MINIMIZATION OF C0



21

EXAMPLE: MINIMIZATION OF C0

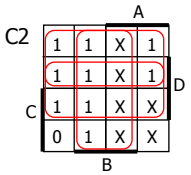


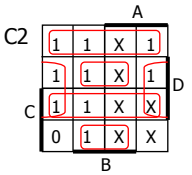
$$C0 = A + C + B\,D + \overline{B}\,\overline{D}$$

22

IMPLEMENTATION AS MINIMIZED SOP (CONT'D)

- Can do better
  - 9 unique product terms (instead of 15)
  - Share terms among outputs
  - Each output not necessarily in minimized form



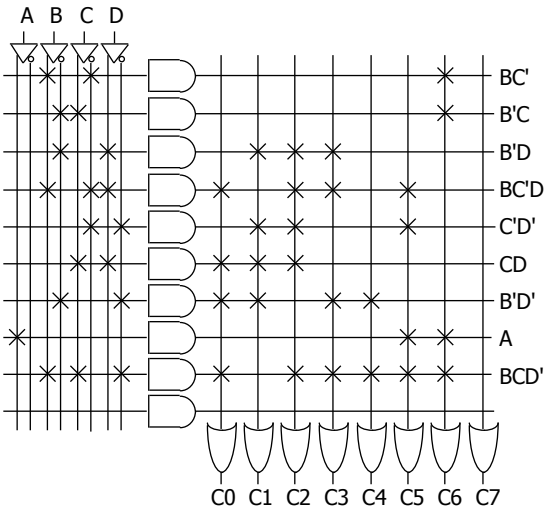


C0 = A + B D + C + B' D'
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B C'
C6 = A + C D' + B C' + B' C

C0 = B C' D + C D + B' D' + B C D' + A
C1 = B' D + C' D' + C D + B' D'
C2 = B' D + B C' D + C' D' + C D + B C D'
C3 = B C' D + B' D + B' D' + B C D'
C4 = B' D' + B C D'
C5 = B C' D + C' D' + A + B C D'
C6 = B' C + B C' + B C D' + A

23

PLA IMPLEMENTATION



24

CAUSES OF SIGNAL PROPAGATION DELAYS

1. Real gate: in response to a change at the input the output changes in a short but finite (nonzero) time.
The time necessary to reach the new value of the output: propagation delay.

2. Interconnections: Finite velocity of propagation of electromagnetic waves, delays caused by stray capacitances and inductances.

25

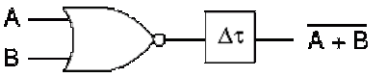
SIGNAL PROPAGATION DELAY

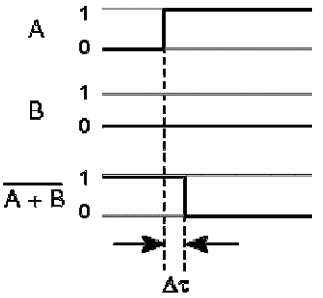
Ideal case: the gates generate the output signals simultaneously with the appearance of input signals, and no time is necessary for signal propagation through the interconnections.

Real case: the gates generate the output signals only with time delay, and the propagation velocity through the interconnections is finite, resulting in an additional delay.

26

A NOR GATE WITH A LUMPED DELAY





Timing behavior

Lumped element model:
real gate = ideal gate +"lumped delay element"

HAZARDS

The signal propagation delays can cause transitory erroneous operation of the logic networks. Such erroneous phenomena, occurring governed by chance, are called hazards.

The cause of hazards is the timing delay of different components in the logic circuit. The resulting glitches in the circuit may or may not induce additional problems - other than increased issues due to switching noise.

It is good design practice to design circuits to minimize these hazards.

28

HAZARDS

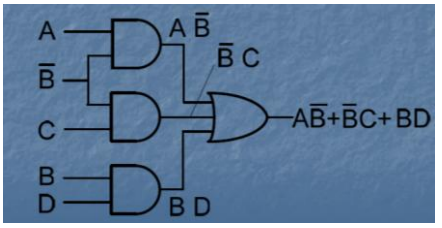
A hazard or glitch in digital logic is a fault in the logic system due to a change at the input. A static hazard is when the output of a logic circuit momentarily changes when its final value is the same as its value before the hazard (when the output is "trying" to remain the same, it jumps once, then settles down). A dynamic hazard (or oscillation hazard) is where a logic circuit will momentarily change back to its original value while changing to a new value.

The cause of hazards is the timing delay of different components in the circuit. The resulting glitches in the circuit may or may not induce additional problems - other than increased issues due to switching noise. It is good design practice to design circuits to minimize these hazards.

29

ANALYSIS OF STATIC HAZARDS

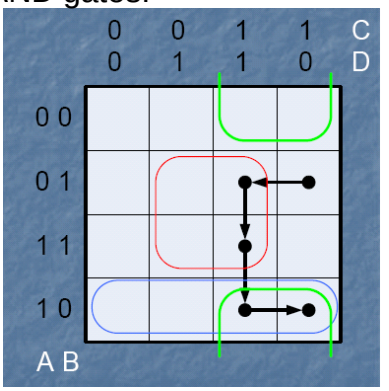
The change sequence of the input states and variables can be followed using the K map.



A	B	C	D
0	1	1	0
0	1	1	1
1	1	1	1
1	0	1	1
1	0	1	0

Control handover between AND gates: static hazards

Loops in the K map corresponding to the AND gates.



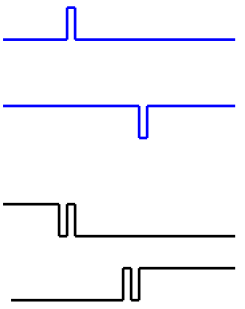
HAZARD PHENOMENA

HAZARD
The pulse „0” or „1” on the output is caused not by the logic conditions. The delays may depend on various and unexpected circumstances (e.g. self-heating of the circuit, etc.), therefore cannot be always controlled.

HAZARD TYPES
Static hazard
„0”-type hazard
„1”-type hazard

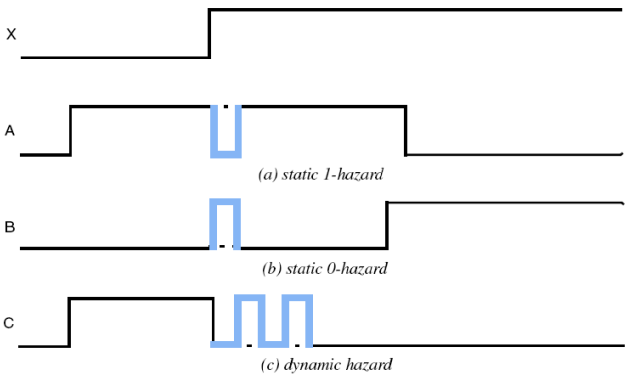
Dynamic hazard

Functional hazard (not shown)



31

STATIC HAZARDS



There are two types of static hazards: a low-going glitch (or static one hazard) is where the high output transitions to a low and back high (1-0-1) and a high-going glitch (or static zero hazard) is where the low output transitions to a high and back low (0-1-0).

32

STATIC HAZARDS

Static 0 hazards occur in product-of sums implementations, but do not occur in sum-of-products implementations. Static 1 hazards occur in sum-of-products implementations, but do not occur in product-of sums implementations.

Adding logic redundancy using a Karnaugh map is the easiest way to eliminate static hazards.

Static hazards can be eliminated using a sum-of-product s implementation containing every prime implicant.

33

STATIC HAZARD IN COMBINATIONAL CIRCUITS

$F(ABC) = AB + \overline{A}C$

$ABC = 111 \rightarrow A\overline{B}C = 011$

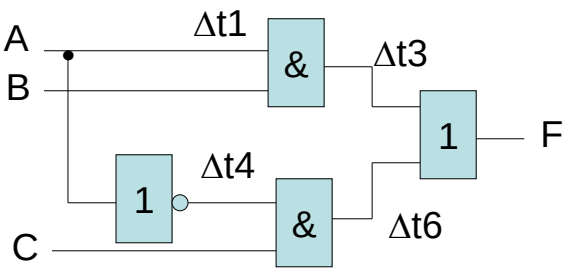
If

$\Delta t1+\Delta t3 < \Delta t4+\Delta t6$

on the output

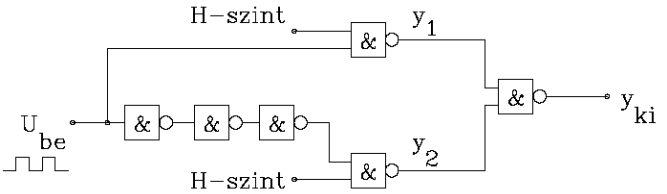
$1 \rightarrow 0 \rightarrow 1$

change will occur!



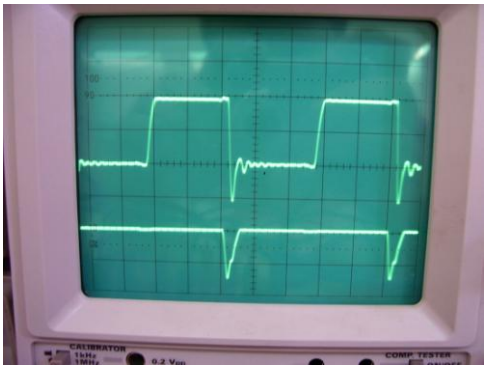
34

STATIC HAZARD



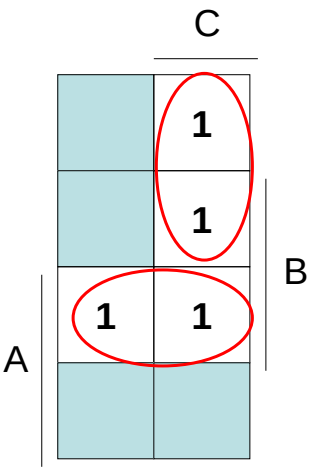
The brief pulse or glitch in the output is caused by the propagation delay difference of the signals through the gates. Measured width at 50% level: app. 40 nsec, series 74 one gate average delay app. 13 nsec.
(Student's measurement.)

1 cm = 100 nsec



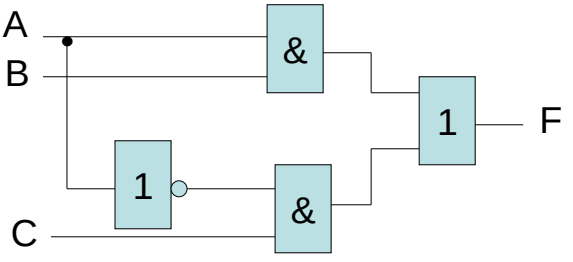
STATIC HAZARD IN COMBINATIONAL CIRCUITS

$F(ABC) = AB + \overline{A}C$



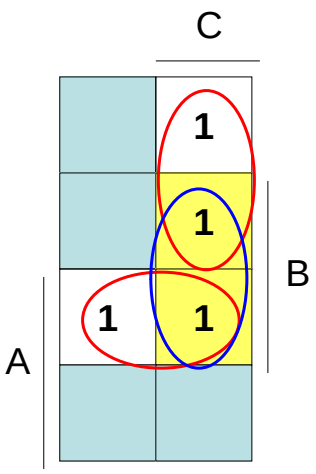
critical transition:

$ABC = 111 \rightarrow A\overline{B}C = 011$



36

ELIMINATION OF STATIC HAZARD (1)



Elimination of static hazard: it is necessary to prevent the influence of critical transition, for this the BC prime implicant should also be covered.

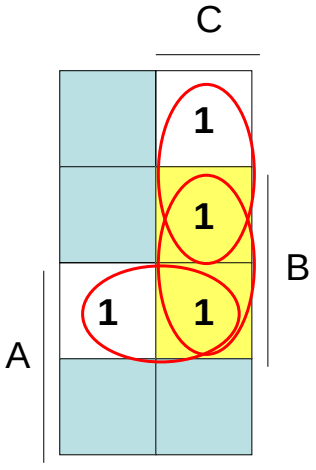
$F(ABC) = AB + \overline{A}C + BC$

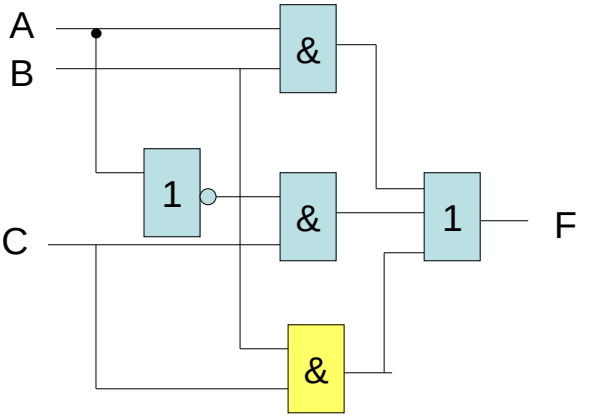
Critical transition:

$ABC = 111 \rightarrow A\overline{B}C = 011$

37

ELIMINATION OF STATIC HAZARD (2)





38

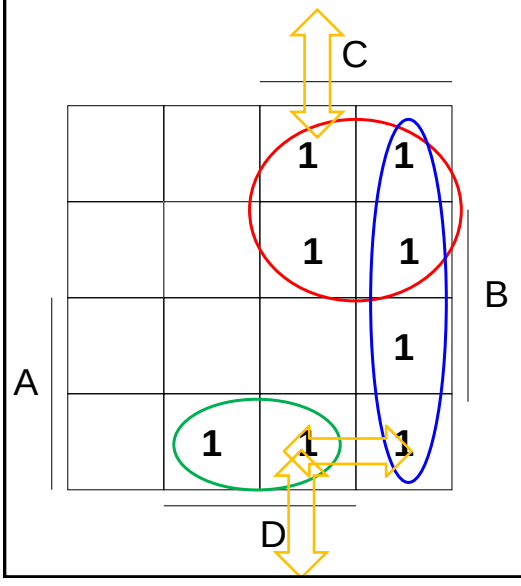
SYNTHESIS OF NETWORKS FREE OF STATIC HAZARDS

The two-level AND-OR network is free of static hazard only if for any/all pairs of adjacent input combinations generating a value of 1 on the output the is an AND gate the output of which is 1 for both adjacent input combination.

In other words: for any two adjacent minterms there is at least one prime implicant in the circuit covering both minterms.

39

STATIC HAZARDS

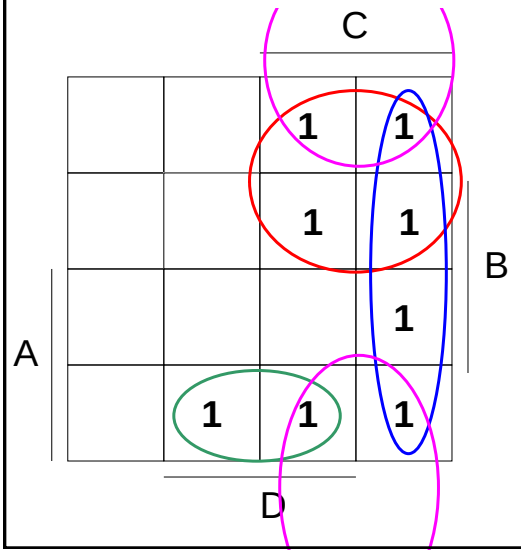


K maps are useful for detecting and eliminating race hazards.

An additional term $\overline{B}C$ would eliminate the potential race hazards, bridging between the green and blue output state or blue and red output states.

40

ELIMINATION OF RACE HAZARDS



An additional term $\bar{B}C$ eliminates the potential race hazards, bridging between the green and blue output state or blue and red output states.

The term is redundant in terms of the logic state of the system, but such redundant terms are often needed to assure race-free dynamic performance.

41

SYNTHESIS OF NETWORKS FREE OF STATIC HAZARDS GUIDELINES

Guidelines for synthesis:

The simplest static hazard free disjunctive form (SOP) can be obtained by adding the possible minimum number and simplest prime implicants to the simplest disjunctive form (SOP) to fulfill the necessary covering conditions.

42

STATIC HAZARD IN PRODUCT-OF-SUM NETWORKS

The above analysis can also be applied to the two-level OR-AND networks..

The only modification is that the maxterms should be considered, an in case of necessity the redundant OR gates ensuring the covering of adjacent loops should be included into the network.

43

EXAMPLE: HAZARD ELIMINATION IN TWO-LEVEL NETWORKS

Construct the static hazard free conceptual AND-OR logic diagram of the function given in its simplest SOP form

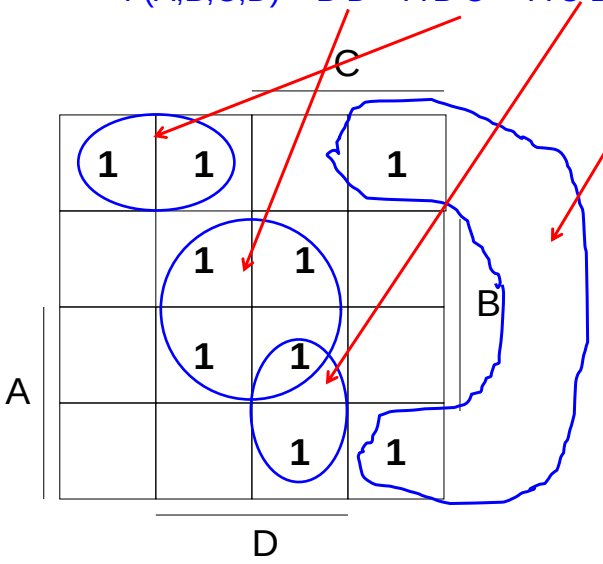
$$F(A,B,C,D) = B\,D + \overline{A}\,\overline{B}\,\overline{C} + A\,C\,D + \overline{B}\,C\,\overline{D}$$

Discuss the NAND gate based implementation, and as an alternative the PLA based implementation too.
(74LS00 4x2 input, 74LS20 2x4 input, 74LS30 1x8 input NAND gates).

44

KARNAUGH MAP: MINIMAL COVER

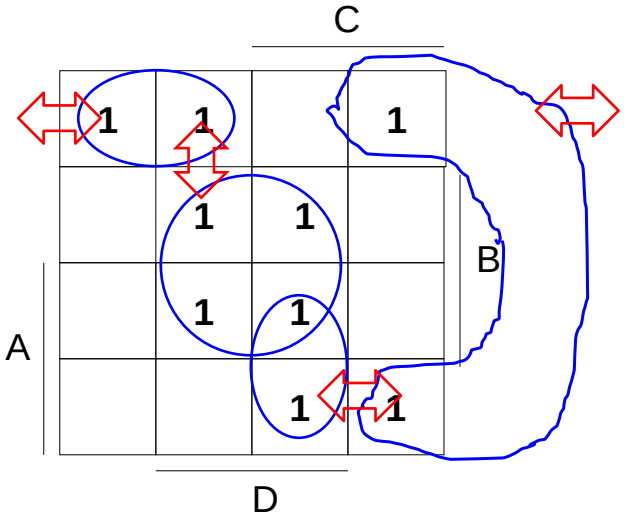
$F(A,B,C,D) = B\,D + \bar{A}\,\bar{B}\,\bar{C} + A\,C\,D + \bar{B}\,C\,\bar{D}$



45

POTENTIAL STATIC HAZARDS

$F(A,B,C,D) = B\,D + \bar{A}\,\bar{B}\,\bar{C} + A\,C\,D + \bar{B}\,C\,\bar{D}$



Hazards can occur for three transitions:

0 0 0 0 ↔ 0 0 1 0

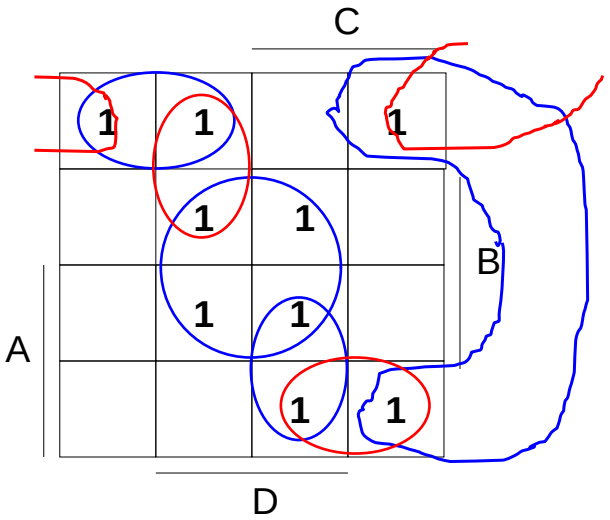
0 0 0 1 ↔ 0 1 0 1

1 0 1 0 ↔ 1 0 1 1

46

ELIMINATION OF HAZARDS

$$F(A,B,C,D) = B\,D + \bar{A}\,\bar{B}\,\bar{C} + A\,C\,D + \bar{B}\,C\,\bar{D}$$



Three additional loops are necessary to eliminate the hazards

47

Complete form: f(d,c,b,a) = sum m(0,1,2,5,7,10,11,13,15)

Minimal form: f(d,c,b,a) = ac + b'c'd' + a'bc' + bc'd

Representation of logic function: Sum of Products Algorithm: Quine-McCluskey

Welcome | K-Map | Boolean n-Cube | Quine-McCluskey

Size 1 primes			Size 2 primes		Size 4 primes	
Number of 1s	Minterm	0-cube	Minterm	1-cube	Minterm	2-cube
0	m0	0000	m(0,1) m(0,2)	000-* 00-0*		
1	m1 m2	0001 0010	m(1,5) m(2,10)	0-01* -010*		
2	m5 m10	0101 1010	m(5,7) m(5,13) m(10,11)	01-1 -101 101-*	m(5,7,13,15)	-1-1*
3	m7 m11 m13	0111 1011 1101	m(7,15) m(11,15) m(13,15)	-111 1-11* 11-1		
4	m15	1111				

Prime Implicants Table

	0	1	2	5	7	10	11	13	15
m(0,1)	X	X							
m(0,2)	X		X						
m(1,5)		X		X					
m(2,10)			X			X			
m(5,7,13,15)				X	X			X	X
m(10,11)						X	X		
m(11,15)							X		X

48

QUINE-MCCLUSKEY SUMMARY

Prime Implicants Table

	0	1	2	5	7	10	11	13	15
m(0,1)	X	X							
m(0,2)	X		X						
m(1,5)		X		X					
m(2,10)			X			X			
m(5,7,13,15)				X	X			X	X
m(10,11)						X	X		
m(11,15)							X		X

Minimal cover: 4 prime implicants, remaining 3 non-essential prime implicants, however they are needed for hazard elimination.

49

SUMMARY

$$F(A,B,C,D) = B\,D + \bar{A}\,\bar{B}\,\bar{C} + A\,C\,D + \bar{B}\,C\,\bar{D}$$

To eliminate the static hazards in the minimal network realized by its four essential prime implicants, it should be be complemented by three redundant prime implicants:

$$\bar{A}\,\bar{C}\,D,\ \ A\,\bar{B}\,C,\ \ \bar{A}\,\bar{B}\,\bar{D}$$

The hazard free network contains one 2-input AND gate, six 3-input AND gate and one 7-input OR gate.

50

IMPLEMENTATION WITH NAND GATES

Implementation with NAND gates: (AND-OR ⇒ NAND-NAND)

Minimal cover:

74LS00 (4x2 input)	1/4
74LS20 (2x4 input)	1 1/2
74LS30 (1x8 input)	1

Hazard free network:

74LS00 (4x2 input)	1/4
74LS20 (2x4 input)	3
74LS30 (1x8 input)	1

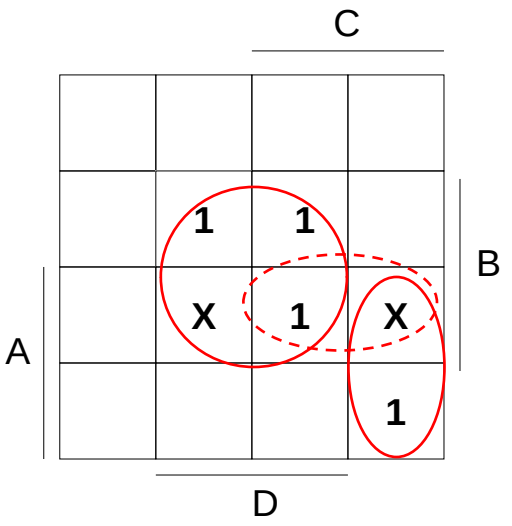
51

STATIC HAZARDS IN INCOMPLETELY SPECIFIED NETWORKS

If the logic function to be realized is incompletely specified, i.e. it contains don't care terms, then the method and approach of obtaining the simplest static hazard free two-level logic network is less systematic.

52

HAZARD ELIMINATION ANALYSIS WITH DON'TCARE TERMS



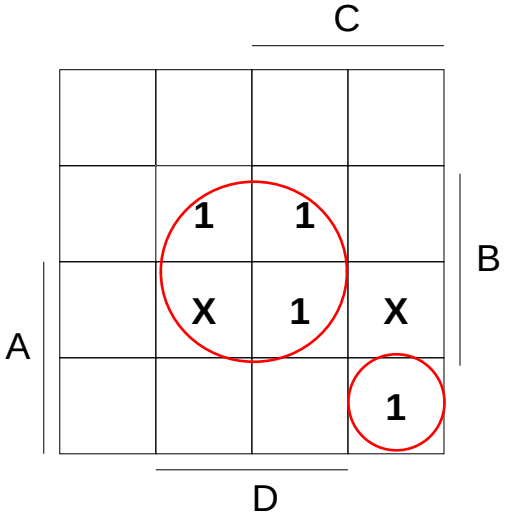
Cover minimization followed by hazard elimination using redundant prime implicant (all prime implicants included): m(5,7,13,15) + m(10,14) + m(14,15)

The prime implicant m(14,15) is really necessary?

This depends on the interpretation of the X terms!

53

HAZARD FREE MINIMAL COVER



m(5,7,13,15) + m(10)

3 gates/8 pins

Previous version:

4 gates/11 pins

Approach: mostly heuristic

54

STATIC HAZARDS IN INCOMPETELY SPECIFIED NETWORKS

The construction of minimal hazard free two-level network for the incompletely specified logic functions might depend on the interpretation of don't care terms.

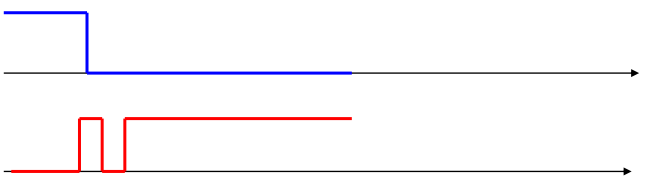
If the hazard elimination should be performed for the don't care terms too, then in the assignment of value for the don't care terms is not always and necessarily based on the consideration of obtaining the simplest prime implicants.

In such cases the simplification process might include heuristic (trial-and-error) steps too.

55

DYNAMIC HAZARD

Dynamic hazard occurs, when in the case of the change of a single input should result a single change of the output (e.g. input: $1 \Rightarrow 0$, output $0 \Rightarrow 1$) , but instead of it the stationary output is reached only with an additional (double) jump (e.g. $0 \Rightarrow 1 \Rightarrow 0 \Rightarrow 1$).



56

DYNAMIC HAZARD

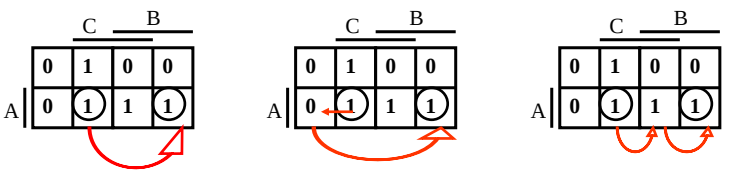
Dynamic hazards can only occur in three-level networks or above, if on any of the levels a static hazard is present.

Therefore eliminating the possible static hazards on each individual level, the dynamic hazard is also eliminated automatically.

57

FUNCTIONAL HAZARD

It can occur if two or more input variables change simultaneously.
E.g. for the transition 101⇒110 two different time sequence is possible, therefore on the output an unwanted 0 state can occur for a short time.



1⇒1 1⇒0⇒1 1⇒1⇒1

58

ELIMINATION OF FUNCTIONAL HAZARDS

The best method to eliminate the functional hazards to disallow the non adjacent input changes. This should be made in the unit generating the input combinations.

The other possibility is to use synchronizing and clock signal.

59

REVIEW QUESTIONS

1. Define the concept of symmetric and partially symmetric Boolean functions and give appropriate illustrative examples.

2. Describe the structure and properties of symmetric and partially symmetric functions on the Karnaugh map. Discuss their implementation using AND-OR-XOR logic.

3. Define and describe the concept, cause, and effects of hazards in a combinational circuit.

4. Define and explain the following concepts: *static hazard*, *dynamic hazard* and *functional hazard*.

4. Describe and discuss the method of elimination of static hazards in a combinational circuit.

5. Describe the main steps of the Quine-McCluskey algorithm for finding the prime implicants and to establish the minimal cover.

60

PROBLEMS AND EXERCISES

1. Implement the logic function shown below:
(a) using a two level minimal system (minimal cover),
(b) without static hazard.

$F(A,B,C,D,E) = \Sigma^5 (2,6,8,10,12,14,17,19,21,23,26,27,30,31)$

(HINT: the minimized SOP form contains five 4-cube (i.e. 3-variable) loops. Two additional product terms are necessary to eliminate the static race hazards.)

Supplementary exercise: Repeat the minimization using the Quine-McCluskey algorithm too.

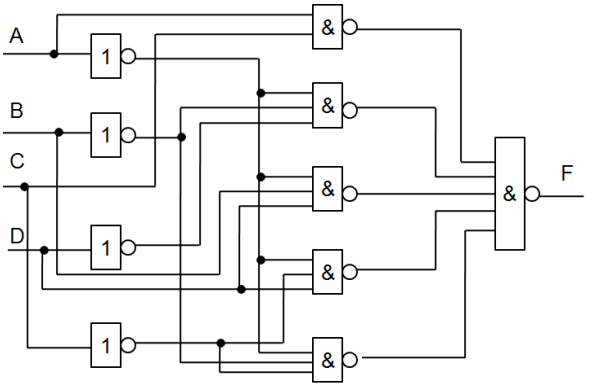
61

PROBLEMS AND EXERCISES

2. Analyze the combinational circuit shown below.
- Is it minimal?
- Is it free of static hazard?

If not, reengineer the circuit to obtain the minimal hazard-free form!

.



62

PROBLEMS AND EXERCISES

3. Using the Quine-McCluskey method find all prime implicants of the function below. Give also the minimal cover.

$F(A, B, C, D) = \Sigma^4(0, 4, 5, 6, 7, 9, 11, 13, 14)$

63

END OF LECTURE

64