Testability of VLSI

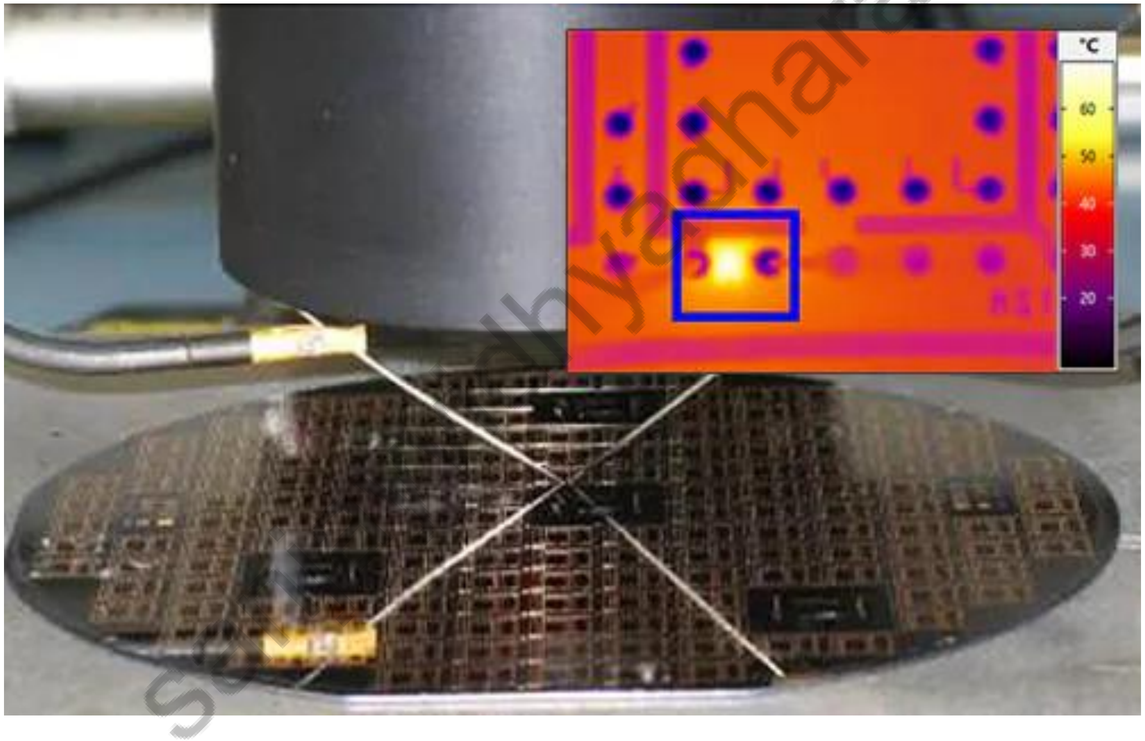
Lecture 6B:
Introduction to Automatic Test Pattern Generation

By Dr. Sanjay Vidhyadharan

Testing

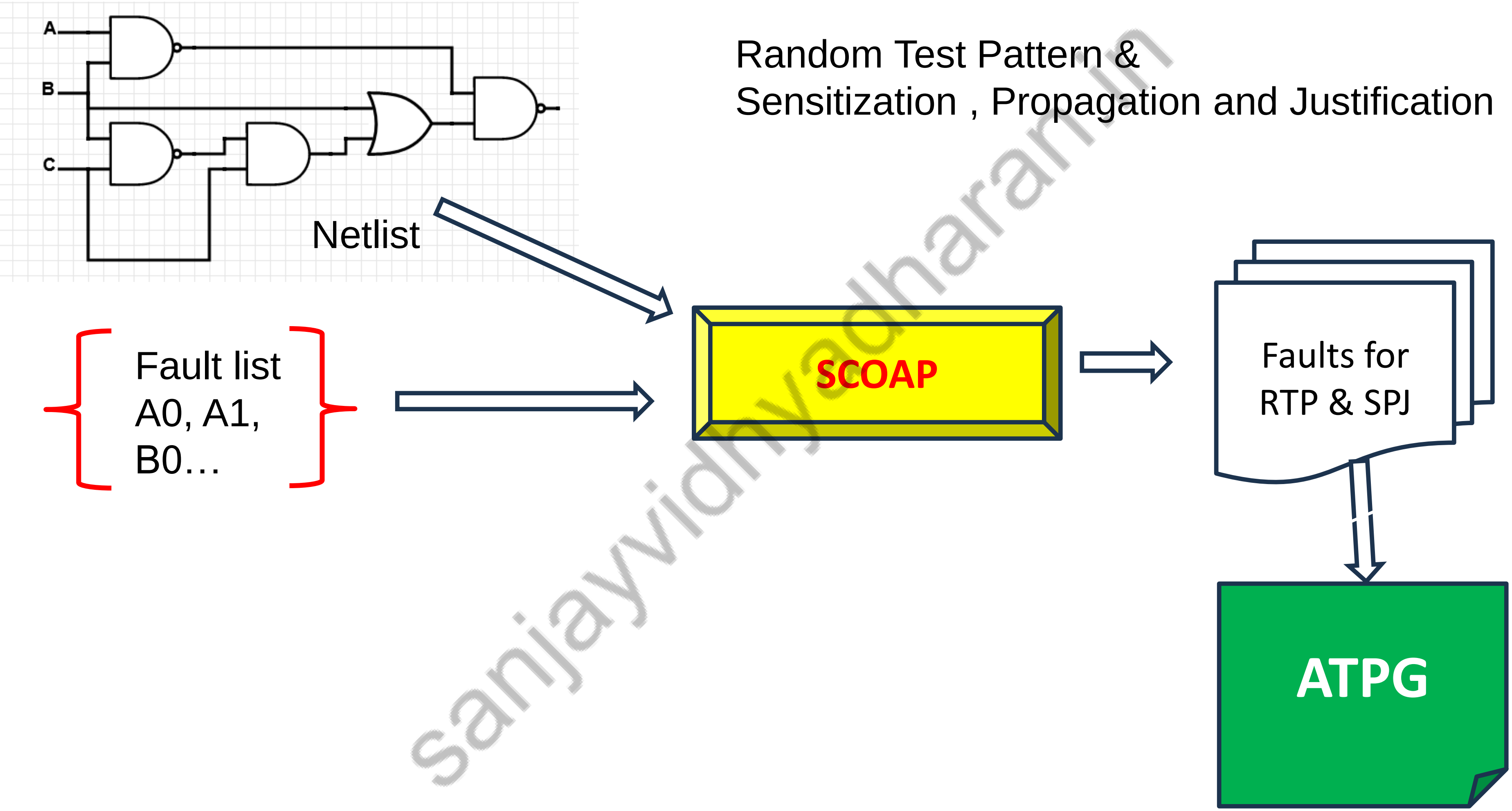
Thermal Imaging.

Powering up the chip and application of few test pattern and using high resolution IR camera to capture hot and cold areas of chip.



https://www.infratec.eu/thermography/thermography-on-the-trail-of-the-fault/

ATPG



Random Test Pattern &
Sensitization , Propagation and Justification

Netlist

Fault list
A0, A1,
B0…

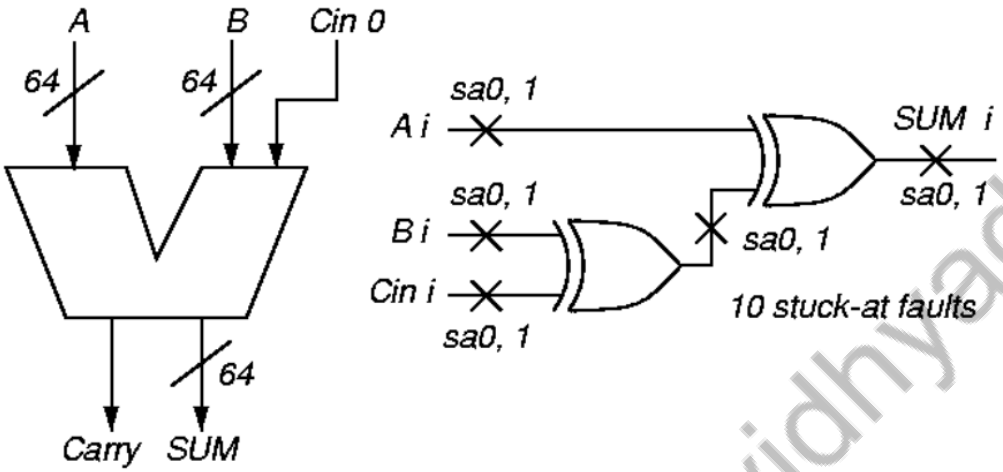
SCOAP

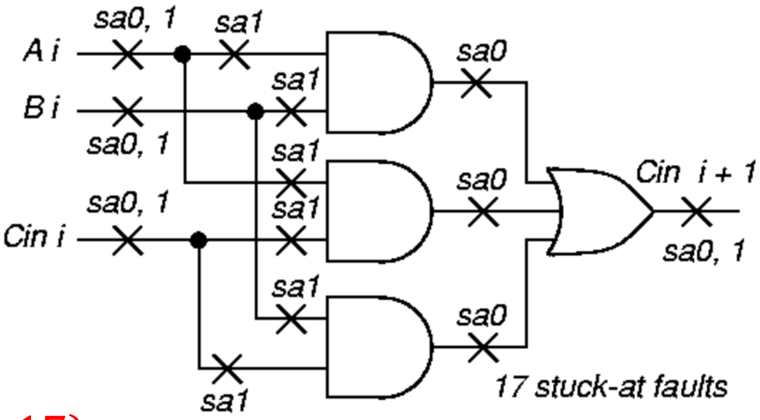
Faults for
RTP & SPJ

ATPG

ATPG

Functional Versus Structural Testing





$Functional\ Test\ \ Vectots\ 2^{129} = 6.8\ *\ 10^{38}$

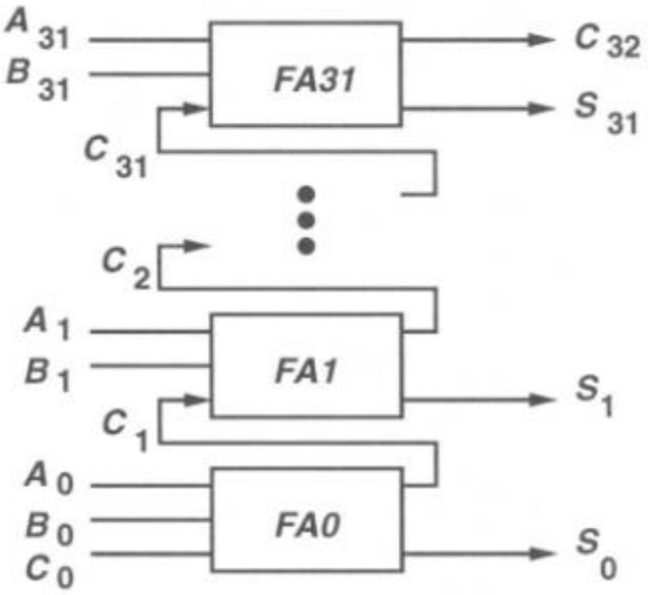
$Required\ time\ with\ \ Clock\ of\ 1\ GHz \approx 22\ years$

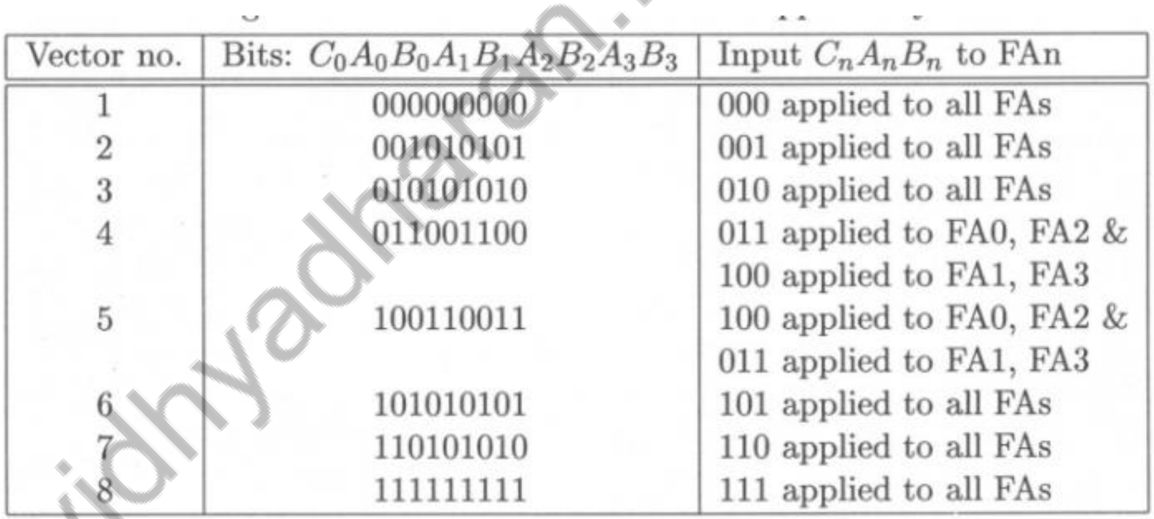
$Max\ Structural\ Test\ vectors\ required\ \ 64 * (10 + 17)$

$Actaul\ Structural\ Test\ vectors\ required\ \ could\ be\ smuch\ lesseras\ multiple\ s - a$
$faults\ gets\ detected\ with\ single\ vector$

ATPG

Logic design of a 32-bit ripple-carry adder.





Vector no.	Bits: $C_0A_0B_0A_1B_1A_2B_2A_3B_3$	Input $C_nA_nB_n$ to FAn
1	000000000	000 applied to all FAs
2	001010101	001 applied to all FAs
3	010101010	010 applied to all FAs
4	011001100	011 applied to FA0, FA2 & 100 applied to FA1, FA3
5	100110011	100 applied to FA0, FA2 & 011 applied to FA1, FA3
6	101010101	101 applied to all FAs
7	110101010	110 applied to all FAs
8	111111111	111 applied to all FAs

The first seven vectors cover all stuck-at faults. One may, therefore, use only the first seven vectors in the manufacturing test.

Possible only for Modular Structures.

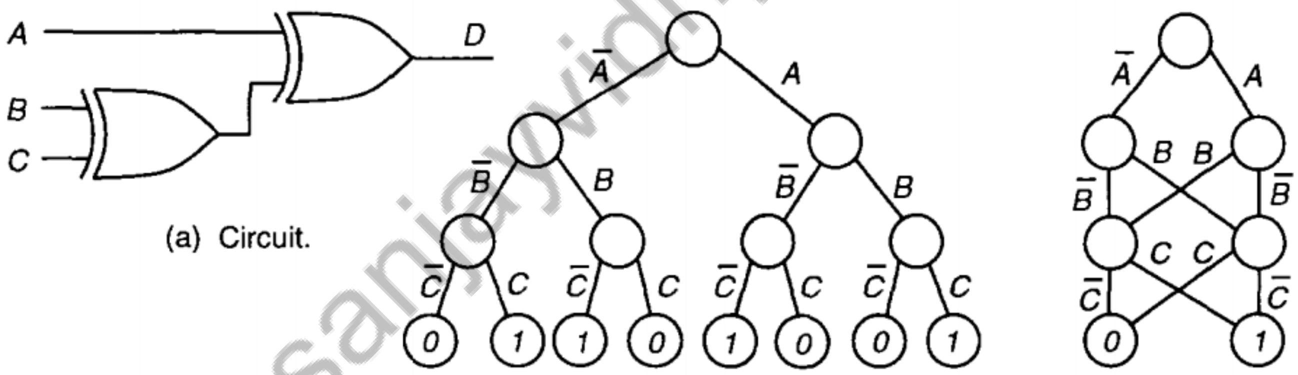
Types of ATPG Algorithm

1. Exhaustive.

In this approach, for an *n*-input circuit, we generate all input patterns.

The circuit is partitioned into cones of logic, each with 15 or fewer inputs. We can then perform exhaustive test-pattern generation for each cone.

However, those faults that require multiple cones to be activated in a synergistic way during testing may not be tested.



(a) Circuit.

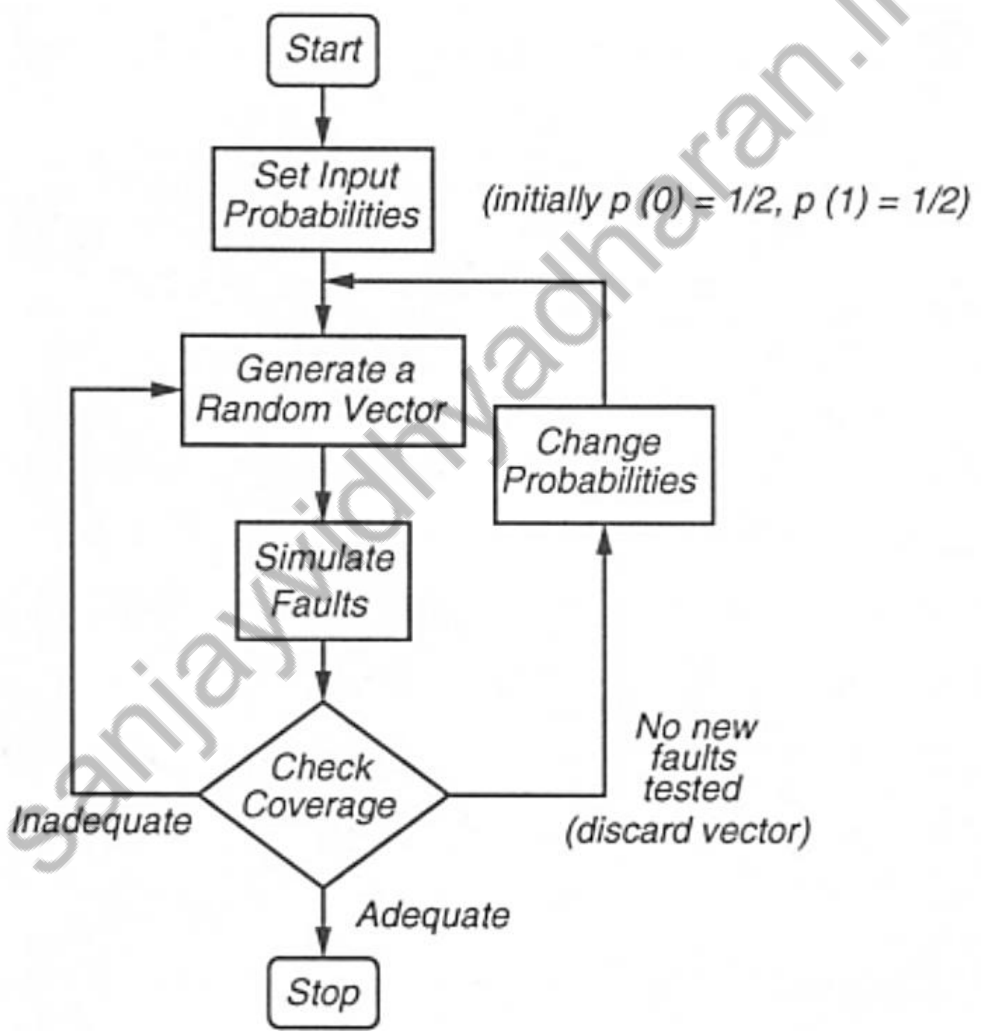
(b) Binary decision tree.

(c) Binary decision diagram.

Vast changes in compute time, depending on the order in which circuit PIs are expanded in the BDD

Types of ATPG Algorithm

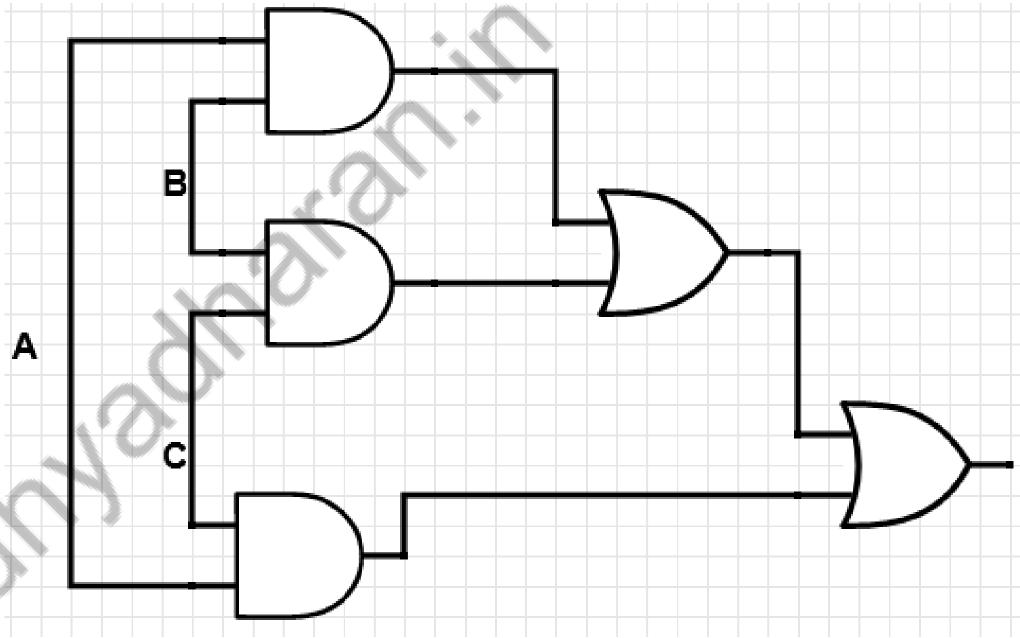
2. Random – Used With Algorithmic Methods



Types of ATPG Algorithm

3. Deterministic ATPG

Symbolic – Boolean Difference



$$F = AB + BC + CA$$

$$F = \bar{A}(BC) + A(B + C + BC)\ Shanon$$

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

$Fg = BC \oplus (B + C)$ =1

$Test\ vector(b, c) = (1{,}0), (0{,}1)$ S-a-0 at A

$$Boolean\ difference, or\ Boolean\ partial\ derivative\ \frac{\delta F}{\delta A} = \frac{AB + BC + CA}{\delta A}$$

$$\frac{\delta F}{\delta A} = \overline{BC}\,\frac{(AB+CA)}{\delta A} = \overline{(BC)}(B + C) = (\bar{B} + \bar{C})(B + C) = \bar{B}C + B\bar{C}$$

We can get the test vectors by mere boolean algebra

Types of ATPG Algorithm

3. Deterministic ATPG

Symbolic – Boolean Difference

For few variables

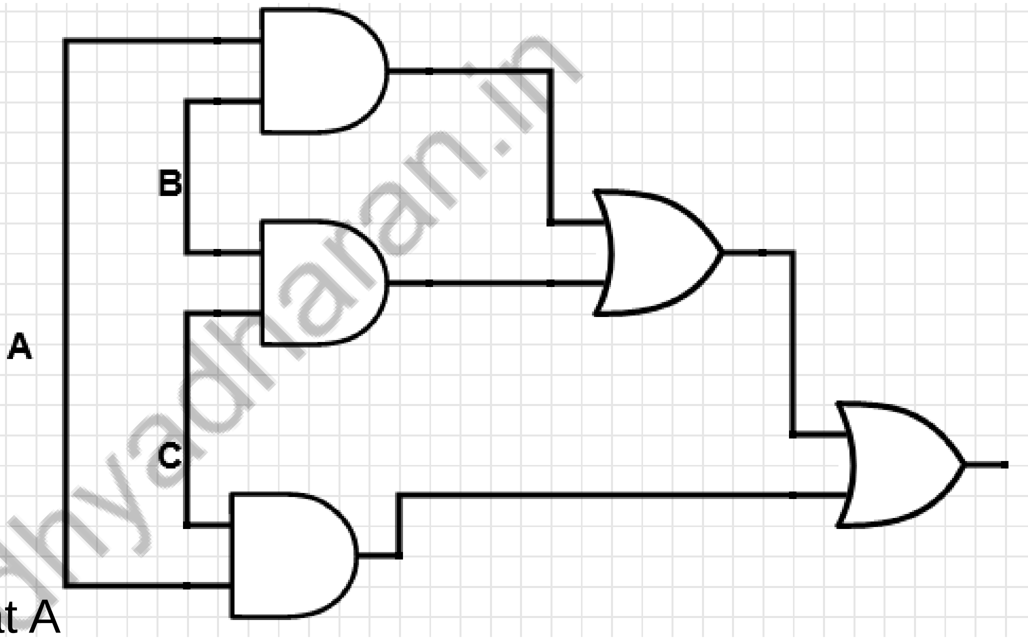
$F = AB + BC + CA$

$F = \bar{A}(BC) + A(B + C + BC)\ Shanon$

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

$Fg = BC \oplus (B + C)$ =1

$Test\ vector(b, c) = (1,0), (0,1)$ S-a-0 at A



		BC			
		00	01	11	10
A	0	0	1	1	1
	1	0	0	1	0

$F = \bar{A}B + BC + C\bar{A}$

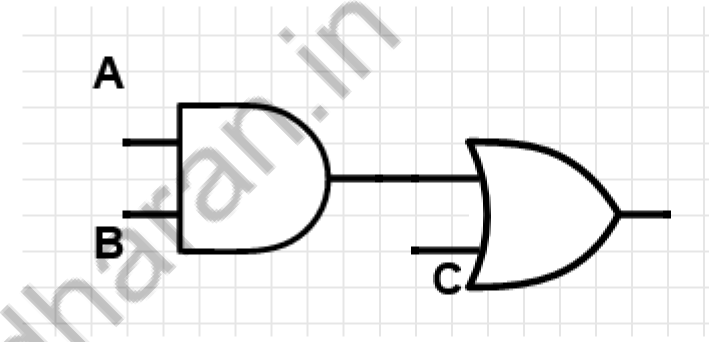
		BC			
		00	01	11	10
A	0	0	0	1	0
	1	0	1	1	1

$F = AB + BC + CA$

		BC			
		00	01	11	10
A	0	0	1	0	1
	1	0	1	0	1

Types of ATPG Algorithm

3. Deterministic ATPG



$F = AB + C$

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

$Fg = C \oplus (B + C)$ =1

$Test\ vector(b, c) = (1{,}0)$ S-a-0 at A

$$\frac{\delta F}{\delta A} = \frac{\delta(AB + C)}{\delta A} = \bar{C}B$$

		BC			
		00	01	11	10
A	0	0	1	1	1
	1	0	1	1	0

$F = \bar{A}B + C$

		BC			
		00	01	11	10
A	0	0	1	1	0
	1	0	1	1	1

$F = AB + C$

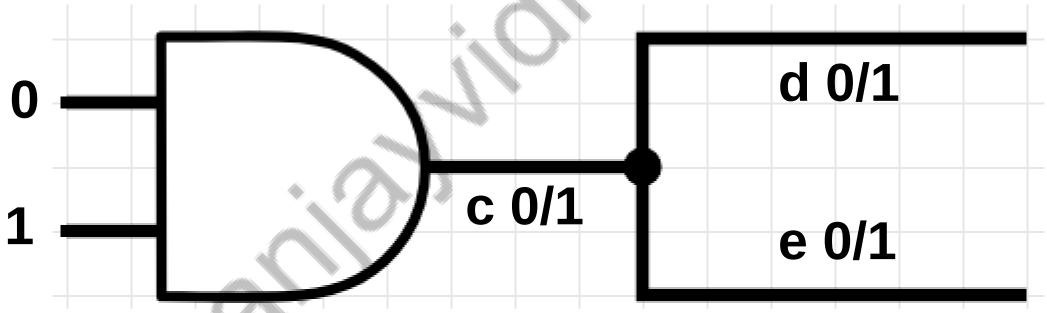
		BC			
		00	01	11	10
A	0	0	0	0	1
	1	0	0	0	1

ATPG Algebra

The ATPG *algebra* is a higher-order Boolean set notation with the purpose of representing both the “good” and the “failing” circuit (or machine) values simultaneously.

This has the advantage of requiring only *one* pass of ATPG to determine signal values for both machines.

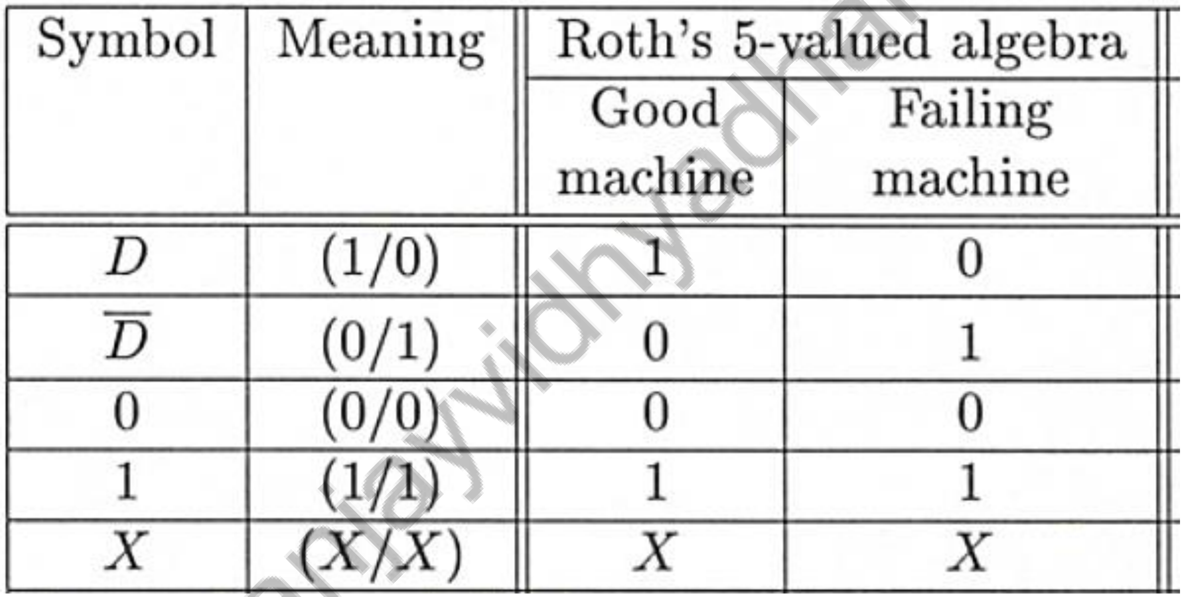
Since a test vector requires that a difference be maintained between the two machines, it is computationally fastest to represent both machines in the algebra, rather than maintaining them separately.



0/1 & 1/0 are not Boolean variables

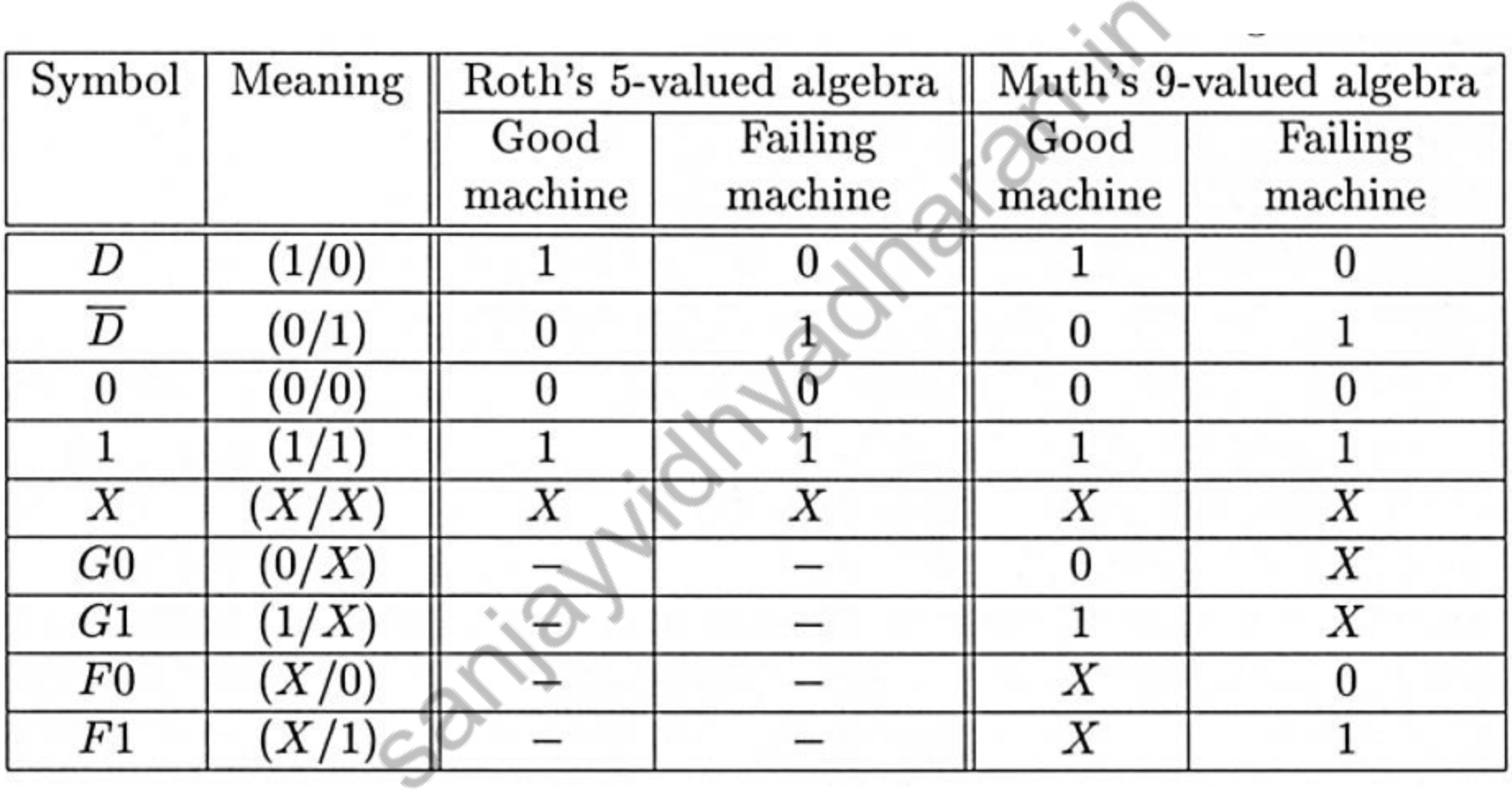
ATPG Algebra

Roth showed how multiple-path sensitization, required to test certain combinational circuits, could be done with his five-valued algebra given in Table below.



Symbol	Meaning	Roth's 5-valued algebra	
		Good machine	Failing machine
D	(1/0)	1	0
$\overline{D}$	(0/1)	0	1
0	(0/0)	0	0
1	(1/1)	1	1
X	(X/X)	X	X

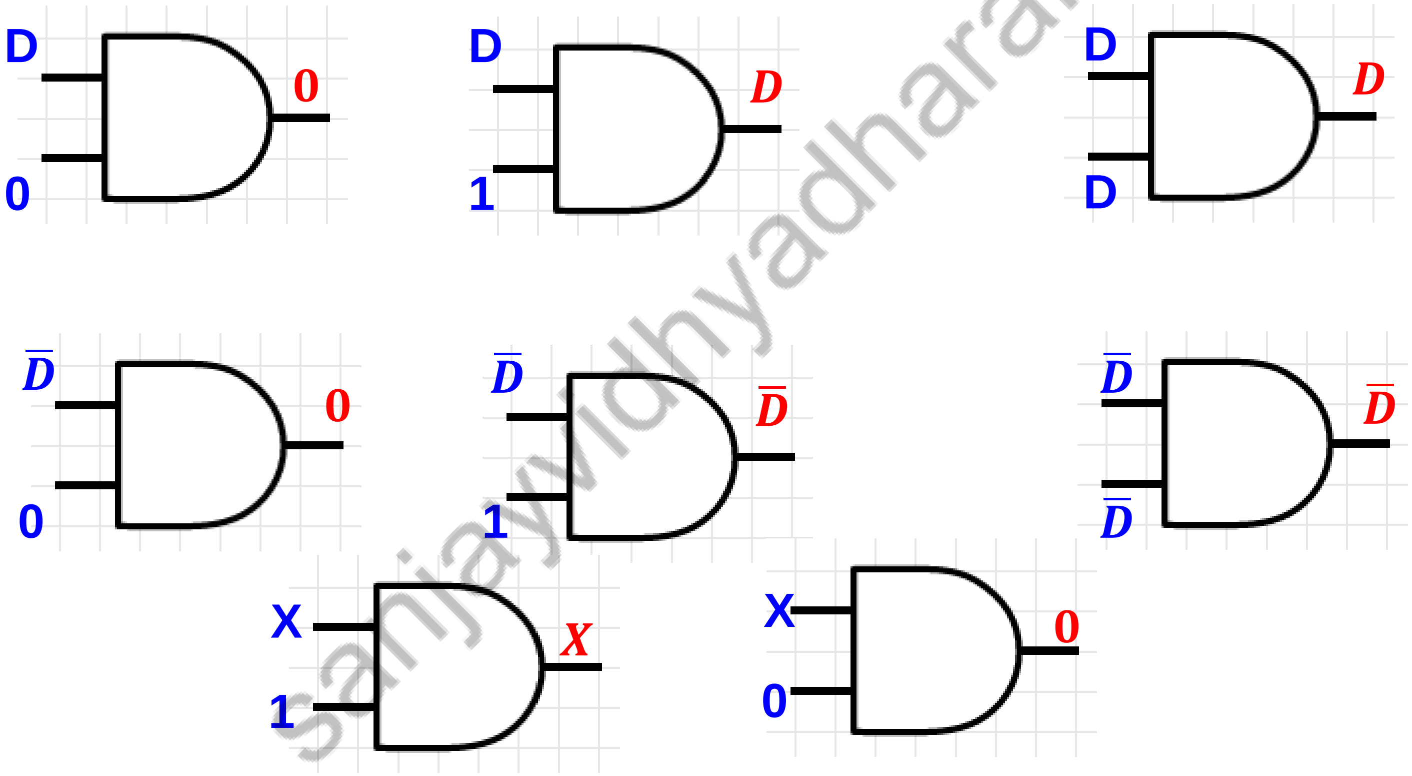
ATPG Algebra



Symbol	Meaning	Roth's 5-valued algebra		Muth's 9-valued algebra	
		Good machine	Failing machine	Good machine	Failing machine
D	$(1/0)$	1	0	1	0
$\overline{D}$	$(0/1)$	0	1	0	1
0	$(0/0)$	0	0	0	0
1	$(1/1)$	1	1	1	1
X	(X/X)	X	X	X	X
$G0$	$(0/X)$	–	–	0	X
$G1$	$(1/X)$	–	–	1	X
$F0$	$(X/0)$	–	–	X	0
$F1$	$(X/1)$	–	–	X	1

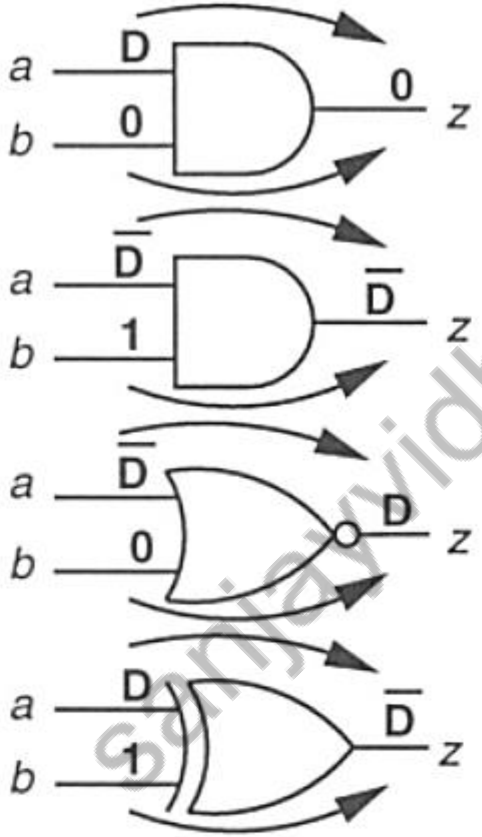
ATPG Algebra

D ***algebra Single Fault***

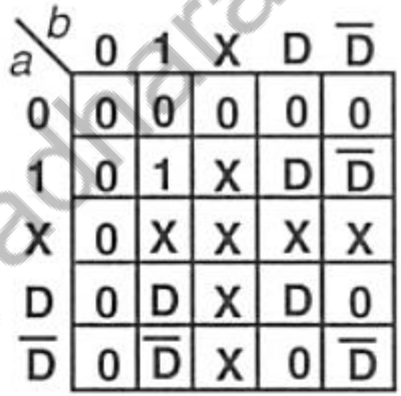


D Algebra

Forward implication



(a) Forward implications.



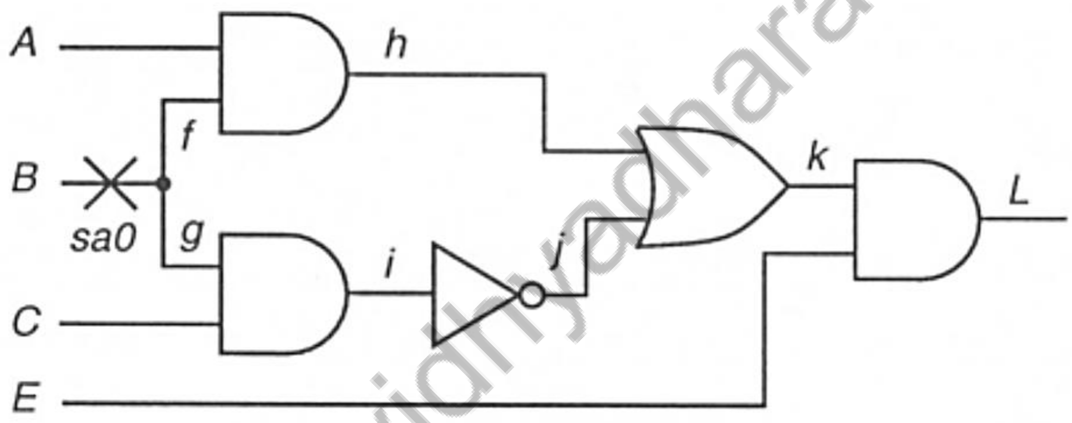
a \ b	0	1	X	D	$\overline{D}$
0	0	0	0	0	0
1	0	1	X	D	$\overline{D}$
X	0	X	X	X	X
D	0	D	X	D	0
$\overline{D}$	0	$\overline{D}$	X	0	$\overline{D}$

(b) AND gate implication table.

Types of ATPG Algorithm

3. Deterministic ATPG

Sensitization , Propagation and Justification. Many Iterations may be required. Different Paths and Simultaneous Multiple paths may be required



1. Sensitization: Test Vector D to detect s-a-0 at B

2. Propagation: Select Path B-f-h-k-L

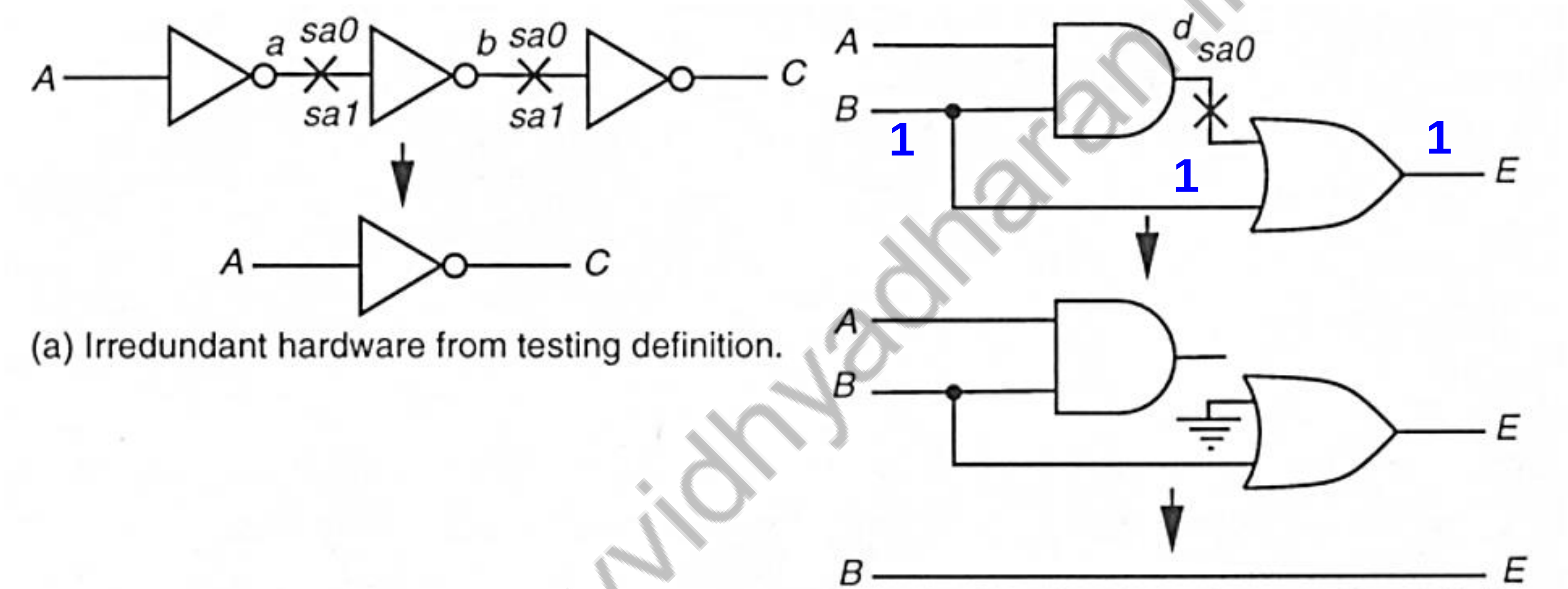
3. Justify: For Path B-f-h-k-L (A=1 , E=1, j=0, i=1(Conflict as i cannot be made 1)

4. Iteration Propagation: Select Path B-g-i-j-k-L

5. Justify (A=0 , E=1,C=1), i= **D**, *j=* $\overline{\mathbf{D}}$ *, k=* $\overline{\mathbf{D}}$ *L=* **D**

In one go Normal and Faulty o/p obtained ABCD (0111) L= 0 Good, L=1 B s-a-0

Redundancy Definition for Testing Purposes



(a) Irredundant hardware from testing definition.

(b) Redundant hardware from testing definition.

Combinational ATPG algorithms provide a major side benefit. *They can determine when the circuit has unnecessary, or redundant, hardware.*

In combinational circuits untestable faults indicate redundant hardware. In testing, one can remove redundant hardware and the circuit will still function exactly the same way as before.

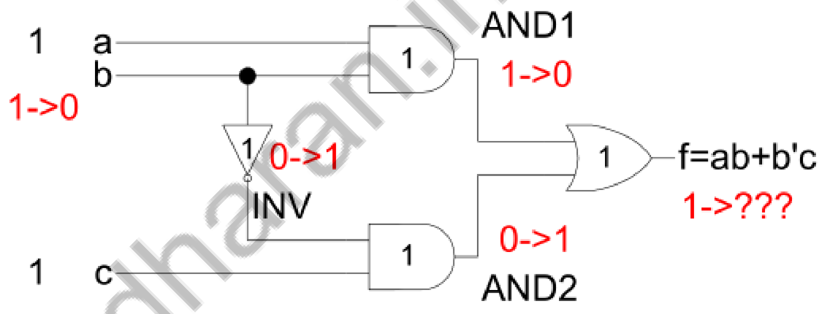
Static Glitch Example

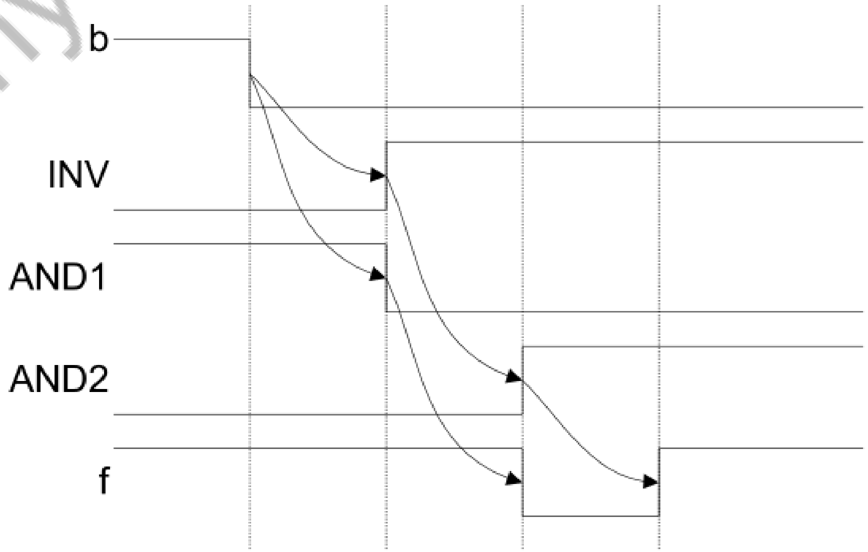
Consider the following circuit with delays where only one input (input b) changes…

Draw a timing diagram to see what happens at output with delays.

From the logic expression, we see that b changing should result in the output remaining at logic level 1…

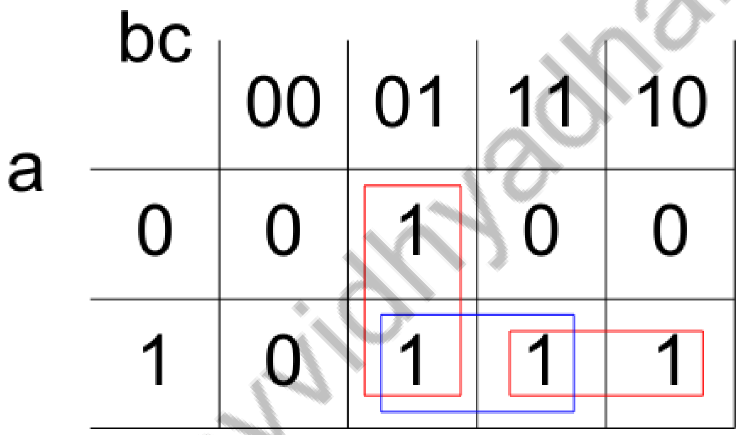
Due to delay, the output goes 1->0->1 and this is an output glitch; **we see a static-1 hazard.**





OUTPUT GLITCH!!!

Static Glitch Elimination



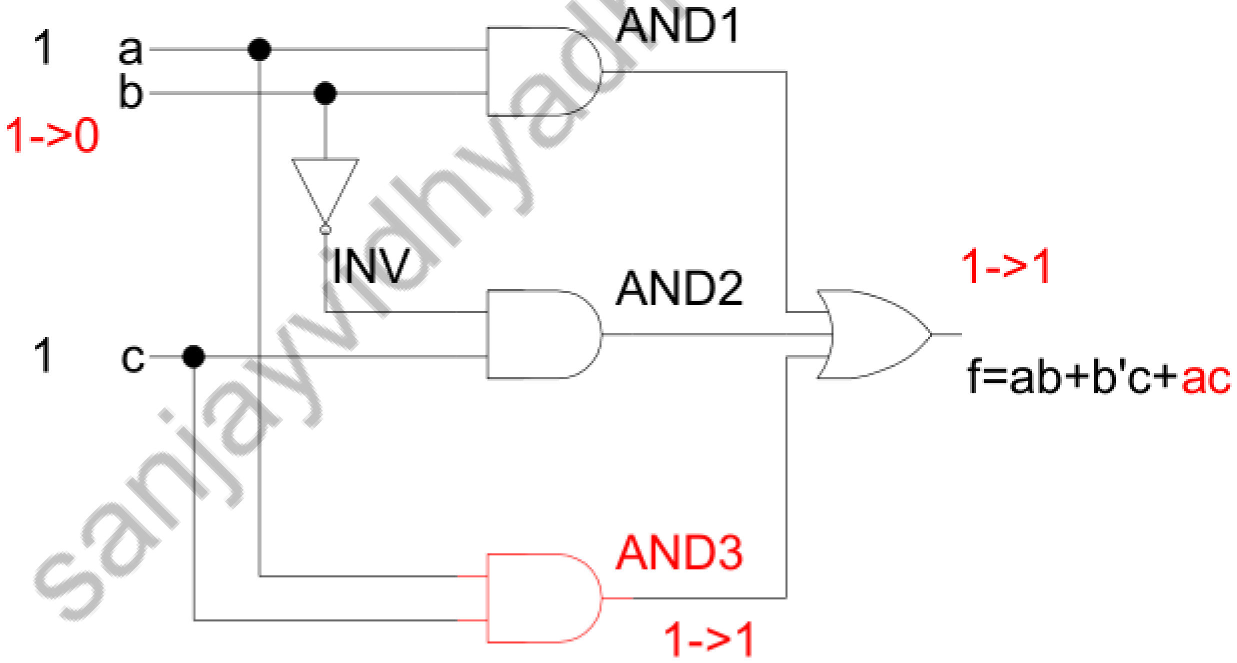
a \ bc	00	01	11	10
0	0	1	0	0
1	0	1	1	1

$f=ab+b'c + ac$

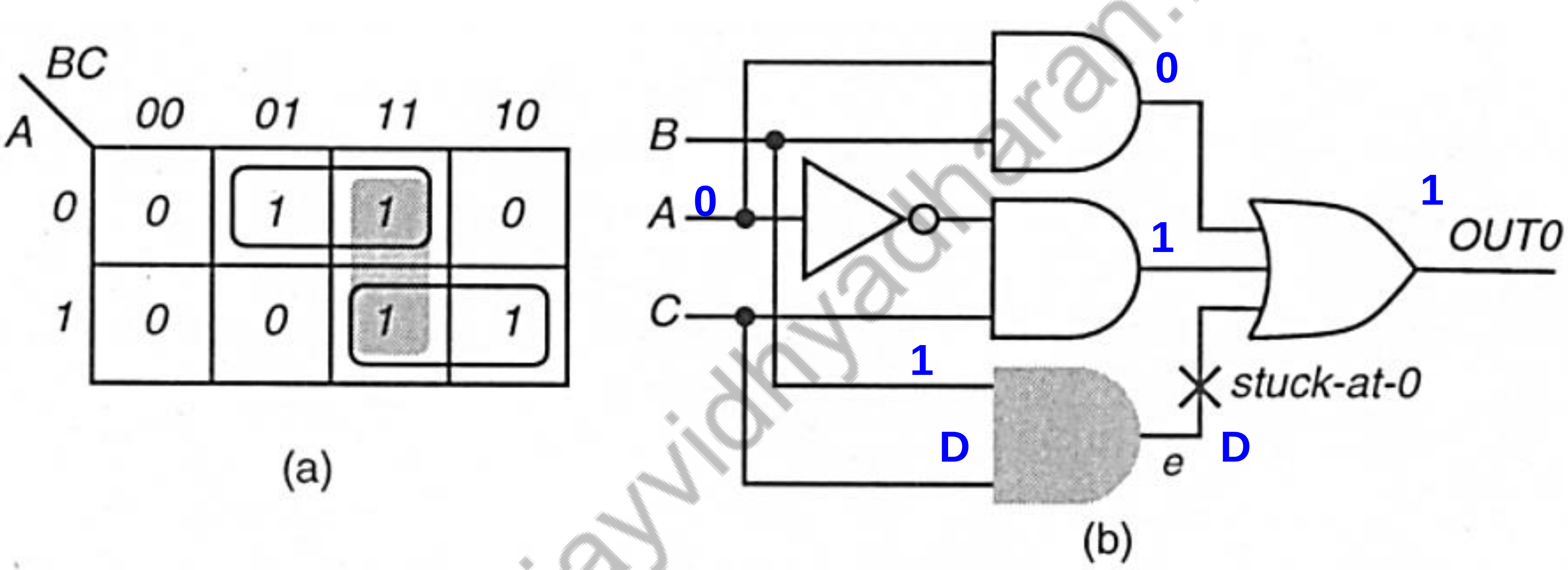
The extra product term does not include the changing input variable, and therefore serves to prevent possible momentary output glitches due to this variable.

Static Glitch Elimination

The redundant product term is not influenced by the changing input.



Redundancy in Testing



A \ BC	00	01	11	10
0	0	1	1	0
1	0	0	1	1

(a)

(b)

References

1. “Essentials of Electronic Testing, for Digital, Memory and Mixed-Signal VLSI Circuits”, Michael L. Bushnell and Vishwani D. Agrawal, – Kluwer Academic Publishers (2000).

2. Video lectures by Professor James Chien-Mo Li
Lab. of Dependable Systems Graduate Institute of Electronics Engineering
National Taiwan University
https://www.youtube.com/watch?v=yfcoKOUV5DM&list=PLvd8d-SyI7hjk_Ci0zpTqImAtpEjdK5JF&index=1

3. NPTEL Lectures
https://www.youtube.com/watch?v=M8VEEaYwlQ&list=PLbMVogVj5nJTClnafWQ9FK2nt3cGG8kCF&index=31

Thankyou