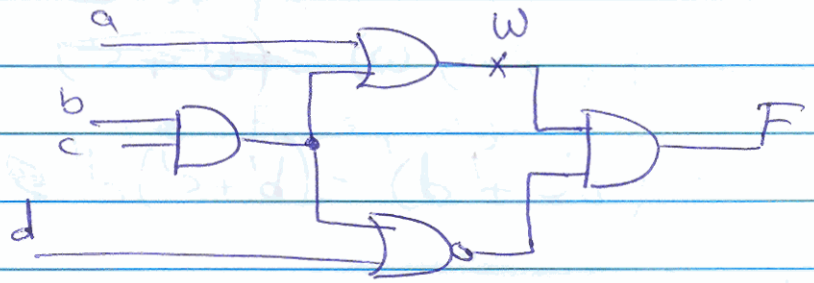
⊛ BOOLEAN DIFFERENCE

Ex.



1) find the set of test inputs to test the fault a sa1. & a sa0

$$F(a,b,c,d) = [(a+bc)][(d+bc)']$$

$$\frac{dF}{da} = f(a=1) \oplus f(a=0)$$

$$= [(1+bc)][(d+bc)'] \oplus [(0+bc)][(d+bc)']$$

$$= d'.(b'+c') \oplus (bc)[d'.(b'+c')]$$

$$= d'b' + d'c' \oplus (bc)[d'b' + d'c']$$

$$= d'b' + d'c' \oplus 0$$

$$= b'd' + c'd'$$

⇒ F is sensitive to a when bcd = 000 or 010 or 100

⊛ the test vectors to test a sa1 are

$$a'.(b'd' + c'd')$$

⇒ abcd = 0000 or 0010 or 0100

⊛ the set of test vectors to test a sa0 are

abcd = 1000 or 1010 or 1100

* To test the stucks at point $\omega \Rightarrow$

$F = f(\text{inputs}, \omega) = (b'+c')$

$= (bc+d)' = (b'+c')d'$ @ ω

$\frac{dF}{d\omega} = f(\omega=1) \oplus f(\omega=0)$

$= b'd' + c'd' \oplus 0 = b'd' + c'd'$

$\Rightarrow$ F is sensitive to ω when

$bcd = 000$ or 010 or 100

$\omega = a + bc$

$\Rightarrow$ to test ω sa0

$(a+bc) \cdot (b'd' + c'd') = 1$

$= ab'd' + ac'd' + 0 + 0 = ab'd' + ac'd' = a(b'd' + c'd'$

$\Rightarrow abcd = 1000$ or 1010 or 1100

to tes ω sa1

$\Rightarrow \omega' = (a+bc)' = a' \cdot (b'+c') = a'b' + a'c'$

$\Rightarrow (a'b' + a'c')(b'd' + c'd') = 1$

$\Rightarrow a'b'd' + a'b'c'd' + a'c'd' = 1$

$a'b'd'(1+c') + a'c'd' = a'b'd' + a'c'd'$

$= a'd'(b'+c')$

$\Rightarrow abcd = 0000$ or 0010 or 0100

④ To test the stucks at pairs F

$F = F$

$\frac{dF}{dF} = 1 \oplus 0 = 1$ (i.e F is always sensitive to itself.)

To test F sa0 ⇒

$F = (a+bc)(d+bc)' = 1$

$= (a+bc)(d'(b'+c')) =$

$= (a+bc)(b'd'+c'd') = ab'd' + ac'd'$

⇒ abcd = 1000 or 1010 or 1100

To test F sa1 ⇒

$F' = (ab'd' + ac'd')'$

$= (a'+b+d).(a'+c+d)$

complement of F $= a' + a'c + a'd + a'b + bc + bd + cd + d$

$= a'(1+c+d+b) + d(1+c+b) + bc$

$= a' + d + bc$

⇒ all vectors except the vectors that test for sa0

	a	b	c	d
✓	0	0	0	0
✓	0	0	0	1
✓	0	0	1	0
✓	0	0	1	1
✓	0	1	0	0
✓	0	1	0	1
✓	0	1	1	0
✓	0	1	1	1
	1	0	0	0
✓	1	0	0	1
	1	0	1	0
✓	1	0	1	1
	1	1	0	0
✓	1	1	0	1
✓	1	1	1	0
✓	1	1	1	1

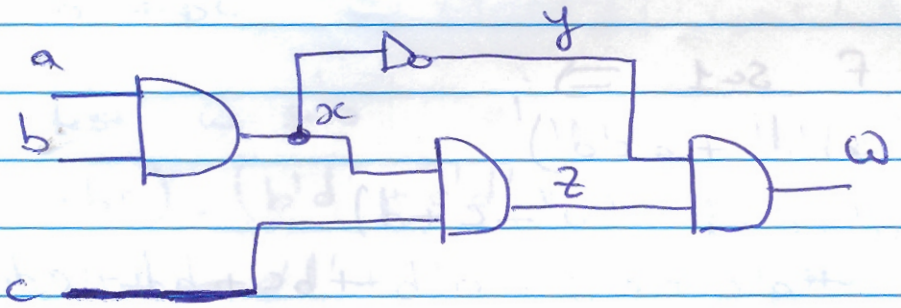
⊛ Untestable faults

- untestable faults are due to ~~hard~~ redundant (~~hardware~~) (unnecessary) hardware.

- The ~~faults~~ untestable faults can be categorized into two parts

(i) completely untestable faults: This is due to completely redundant hardware such that the primary output will not be sensitive for a specific node.

Ex:



to test stucks at point x

(*) x is s-a-0

1) $x = D$ $(a=1, b=1)$

$\Rightarrow y = \bar{D}$

now if we put $c=1 \Rightarrow z = D$

$\Rightarrow \omega = D \cdot \bar{D} = 0$

now if we put $c=0 \Rightarrow z=0$

$\Rightarrow \omega = 0 \cdot \bar{D} = 0$

That means ω is always 0. ~~(not depending)~~

(*)

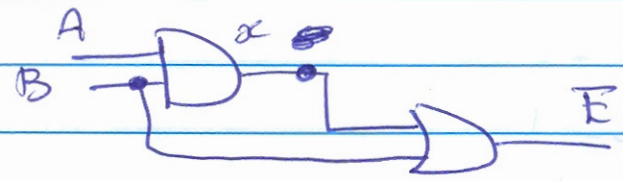
⊛ using Boolean Difference

$\omega = (x \cdot c) \cdot \bar{x} = 0$

$\Rightarrow \frac{d\omega}{dx} = 0 \Rightarrow \omega$ is not sensitive to x.

④ untestable faults due to some redundancy in hardware.

Ex: $E = AB + \bar{B}$



x sa0 ⇒

$x = D \Rightarrow A = B = 1$

$\Rightarrow E = 1$ (not depending in the real value on node x)

⇒ x sa1 ⇒ $x = \bar{D}$

⇒ AB = 00 or 01 or 10

to propagate node x to E ⇒ B = 0

⇒ test vectors AB = 00 or 10

(i.e node x is testable for sa1 but it is not testable for sa0).

That means F is sensitive to node x.

$F = x + B$

$\frac{dE}{dx} = (1+B) \oplus (0+B) = 1 \oplus B = \bar{B}$

⇒ E is sensitive to x when $\bar{B} = 1 \Rightarrow \boxed{B = 0}$

to test x sa0

$\Rightarrow (x)\frac{dE}{dx} = 1 \Rightarrow x\bar{B} = 1 \Rightarrow$

$(AB)\bar{B} = 0 \neq 1 \Rightarrow$ no test for sa0

to test x sa1 $\quad (x = AB)$

$\Rightarrow (x)' \frac{dE}{dx} = 1 \quad \Rightarrow x' \bar{B} = 1$

$\Rightarrow (A' + B')\bar{B} = 1$

$\Rightarrow A'\bar{B} + \bar{B} = 1$

$\Rightarrow (A' + 1)\bar{B} = 1$

$\Rightarrow \bar{B} = 1$

$\Rightarrow B = 0$

$\Rightarrow$ test vectors are

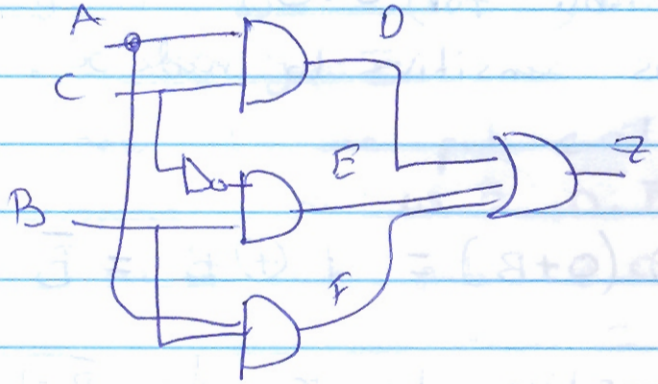
$AB = 00$ or 10

Ex.

$Z = A.C + B.\bar{C}$

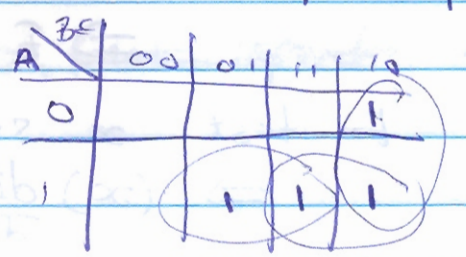
To avoid hazards $\Rightarrow$

$Z = A.C + B.\bar{C} + A.B$



F sa0 is untestable ??

A	B	C	$\bar{A}C$	$B\bar{C}$	
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
1	1	1	1	0	1



$AC + B\bar{C} + AB$

⊛ Design For Testability

- The methods given in the previous lectures are good to test ~~simple~~ (not complex) combinational circuits. but for complex combinational circuits it will be cost and time consuming.

- Also, testing of sequential circuit will be much more difficult because the current state of the circuit should be taken into account as well as the inputs.

- Design for testability (DFT) refers to those design techniques that make test generation and test application cost-effictive.

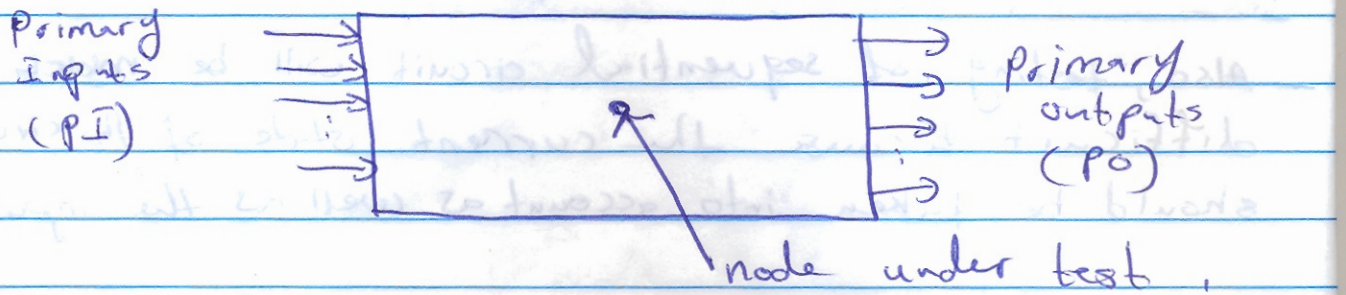
⊛ - Design For Testability at Chip level include
1- Ad hoc DFT.
2- Structured DFT
3- Built In Self Test (BIST)

- Design For Testability at board level ~~instead~~ by using boundary scan.

⊛ Controllability and Observability

- controllability: The ability to control the logic value of an internal node from primary inputs. (The ease with ~~Observab~~ which a node can be set to a value of $D/\bar{D}$).

- observability: The ability to observe the logic value of an internal node at a primary outputs. (The ease with which the value of a node can be steered to a primary output).

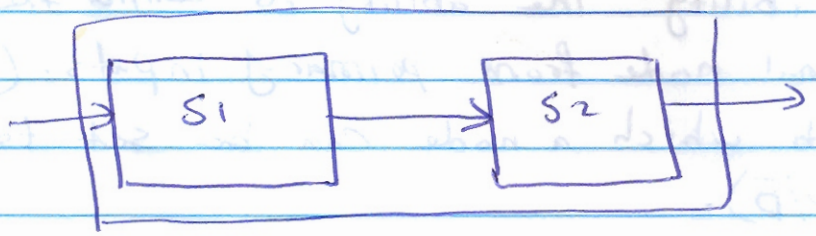


* In general,
* Our main purpose of DFT is to increase the controllability and observability of internal nodes.

* Ad hoc testability
Ad-hoc makes only minor changes to the design approach, but offer treatments for common cases that can cause testability problems.

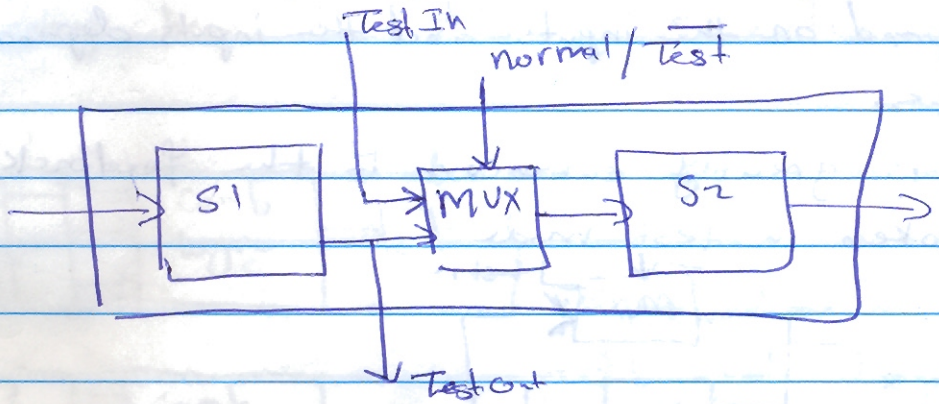
① partitioning of systems to subsystems
The ~~probe~~ testing will be ~~more~~ much easier if the system is partitioned into self contained sub-systems. The subsystems can then be tested in isolation.

Ex.



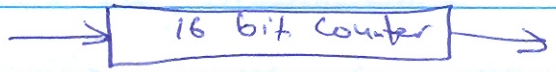
In this example we can control the inputs of

S1 and observe the output of S2.
(we can't control the input of S2, we can't observe the output of S1).
But we would like to be able to directly observe the outputs of S1 and directly control the input of S2 ⇒ this can be achieved using a MUX

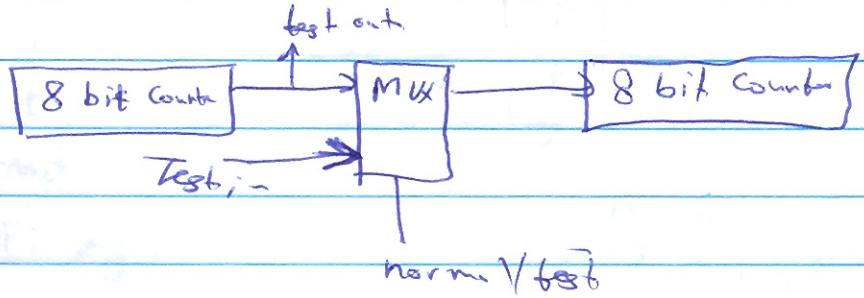


- when normal/Test = 1 ⇒ normal mode ⇒ S_1 is connected to S_2.
- when normal/Test = 0 ⇒ Test in is connected with S2.

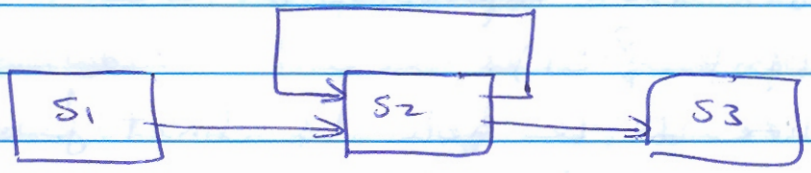
(2) Breaking up long chains of sequential logic
Ex: 16 bit counter ⇒ in order to test this Counter, we would have to count through every possible combination
⇒ This would require $2^{16} = 65536$ clock cycles.



- if we put a MUX and break the counter chain in half ⇒ This arrangement would take only
$2 \times 2^8 = 512$ clock cycles

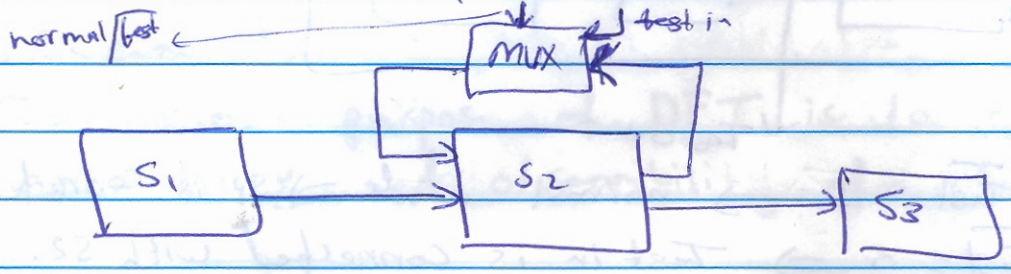


③ Feedback loops



→ The feedbacks create testability problems because the outputs depend on the inputs and the inputs depend on the outputs.

- Testability is greatly enhanced if the feedback loop can be broken in test mode



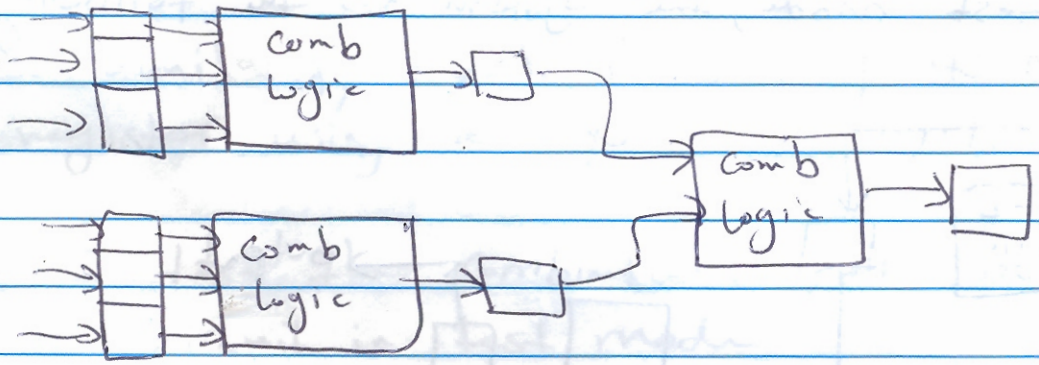
④ Initializing of Sequential Logic

- In order to test a circuit, it must be initialised into a known state. This can be acheived by using reset or preset pins.

⊛ Structured Design For Test

- A structured test methodology makes fairly substantial modifications to the design in order to build in testability.

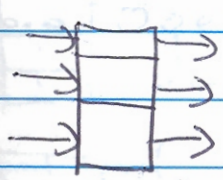
- In general, Digital designs composed from combinational logic and memory elements (flip-flops and registers), and they have the following structure



- structured DFT methods work on the basis of making the internal registers easily accessible through the scan path approach.

⊛ Scan path testing:

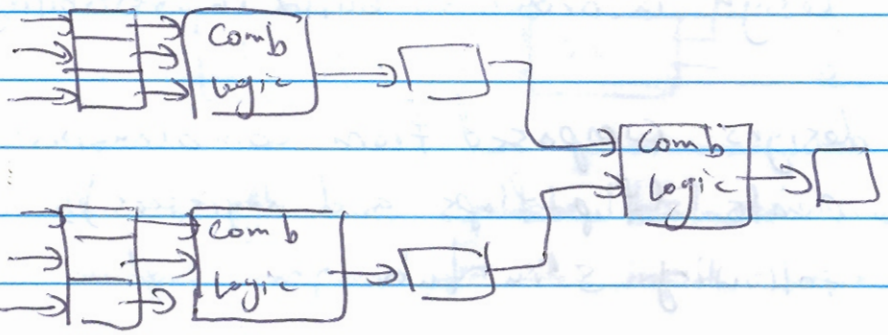
- Scan register is a device with two modes. In its normal mode, it behaves as a normal register, and in its test mode it behaves as shift register.



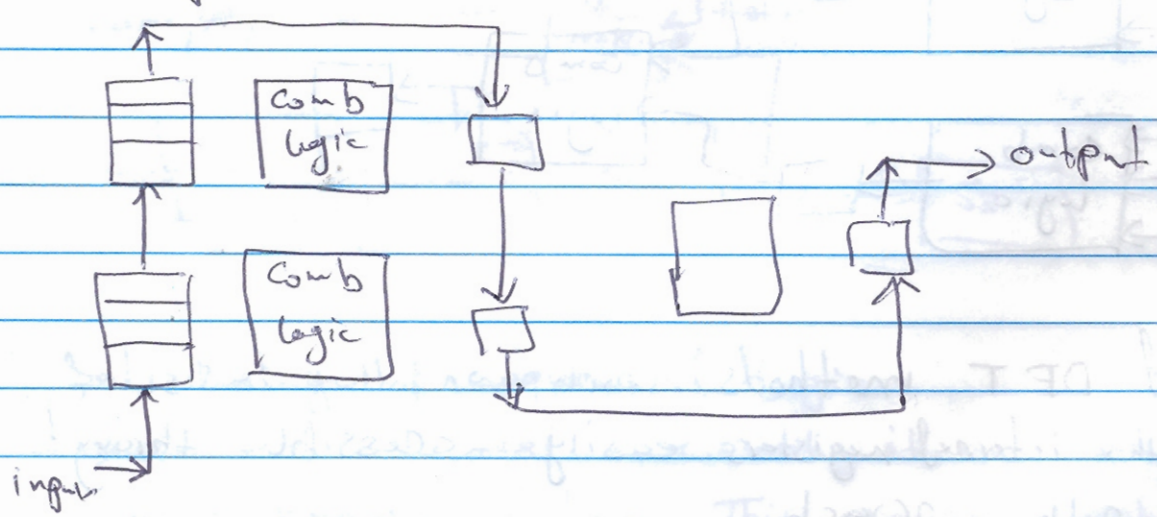
normal mode
normal DFF register

Test mode
like shift register

⇒ so in normal mode the system has the following appearance



and in test mode, the system has the following shape

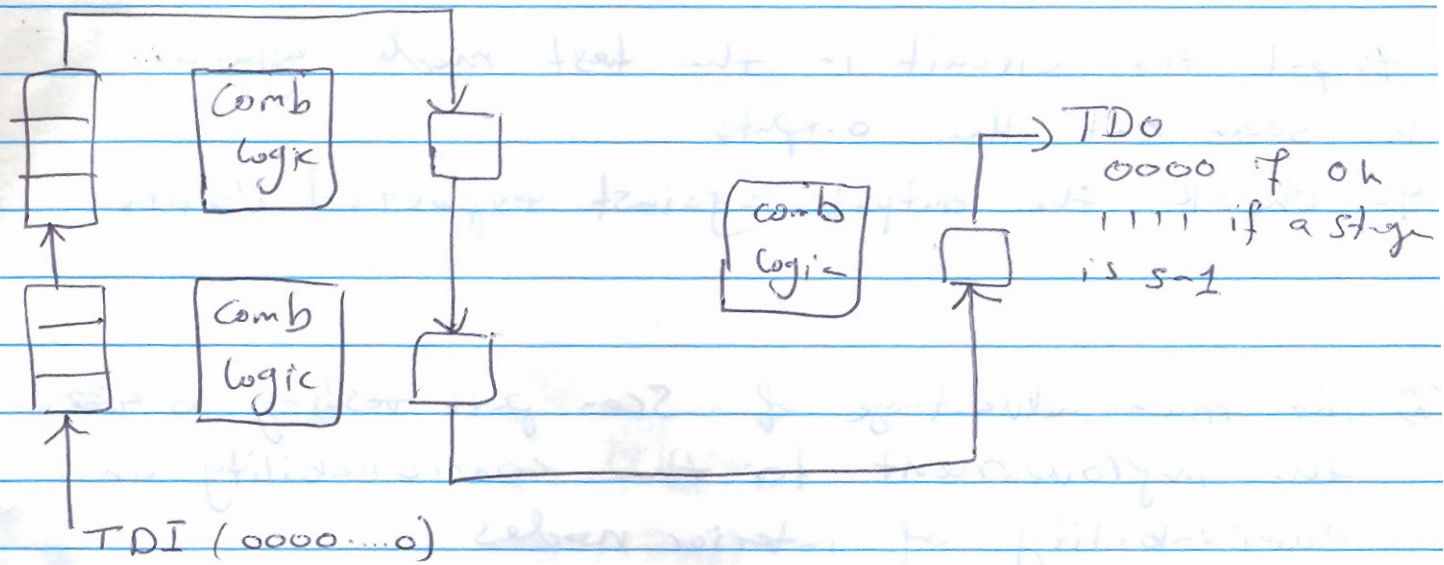


⊛ Procedure for testing circuits using scan path testing.

(1) put the circuit into test mode.

(2) Test the shift register for sa1 and sa0 faults.

2a- scan in a sequence 0000000000 and check the output for any ones

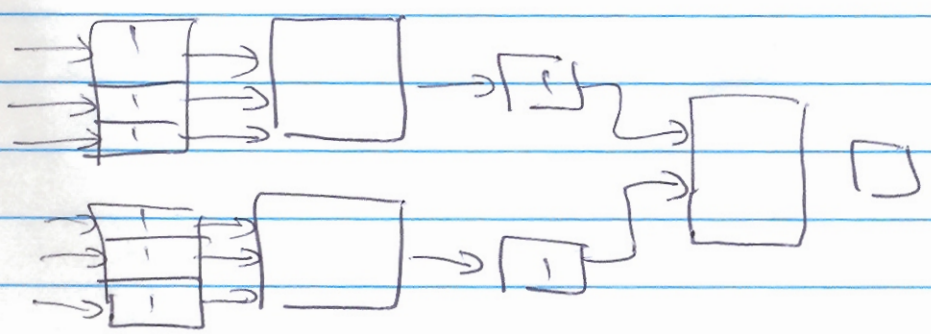


2b- similarly, check for sa0 faults in the shift register using a sequence of (11111111)

3- To test the combinational logic blocks, put the circuit in test mode

4- (let us assume for sake of illustration that for each comb logic block, if the inputs is 111, then the output is also 1)

4- Send using TDI input the sequence 111111,
5- put the circuit in normal mode
6- Execute one clock cycle.



7- put the circuit in the test mode again
8- scan out the outputs
9- check the outputs against expected value.

(X) The main advantage of scan path testing is the improvement to the controllability and observability of interior nodes.

(X) The main disadvantage is that we need to increase the silicon area required for a design by about 20% in order to replace a normal register by a scan register.