VLSI Digital Design

# MODULE V
# TEST TECHNIQUES

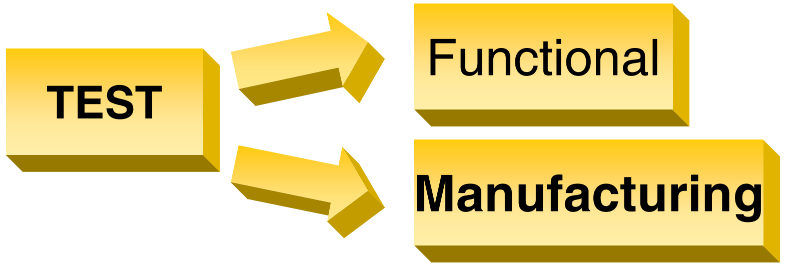
# 5.1 Introduction

## 5.1.1. Test motivation

***"Rule of 10"***

| Detection level | IC fault cost |
|---|---|
| Wafer | 0,01$ ~ 0,1$ |
| Package | 0,1$ ~ 1$ |
| PCB | 1$ ~ 10$ |
| System | 10$ ~ 100$ |
| Field | 100$ ~ 1000$ |

*Detect faults*
*as soon as possible!*



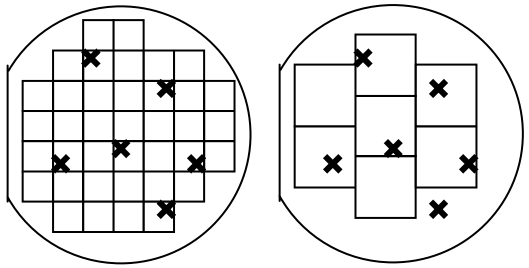
## 5.1.2. Functional test

•Behavior verification
(Function-level specifications verification)

Example: Microprocessor instruction execution

## 5.1.3. Manufacturing test

IC manufacture ➔ Inherent defaults



•IC gates and device verification

## 5.1 Introduction

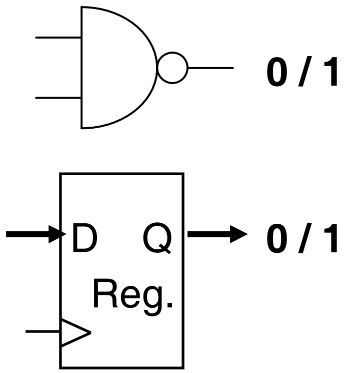
Definitions:

*Defect*: Manufacturing error that produces out-of-range deviations in some device operation.
*Fault*: Incorrect IC operation due to defects.

**A defect does not imply any fault necessarily**

Verification in manufacturing test:

- All gates correctly switch to both logic levels
- All registers correctly store both states



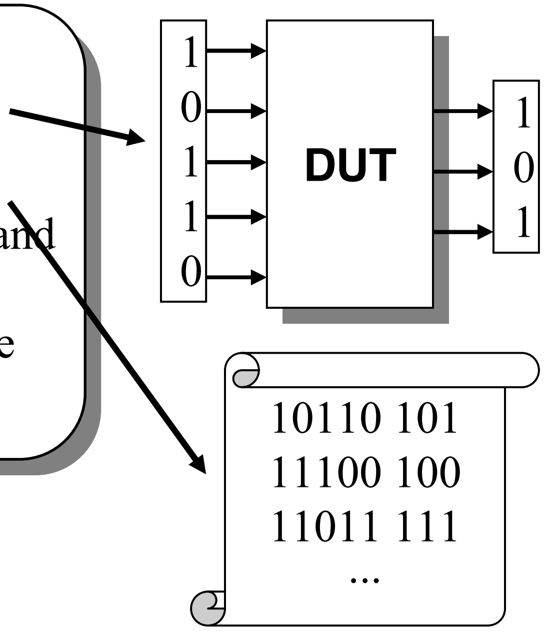
Other verifications:

- I/O level test (noise margin)
- Speed test
- Quiescent current test ($I_{DDQ}$), in complementary circuits.

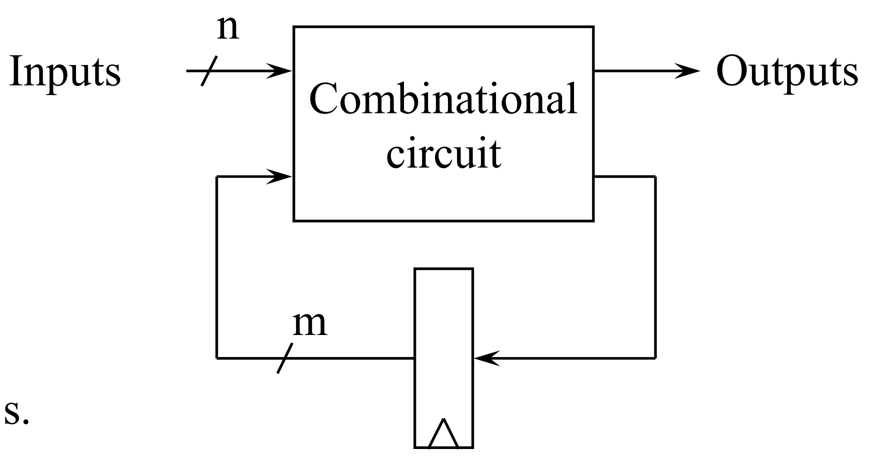
Wafer- and package-level tests are possible.

## 5.1 Introduction

***Test pattern***: Applied input and expected output signals combination.
***Test program***: Set of applied vectors.
***Exhaustive test***: Verification of all possible system input and state combinations.
***Process yield***: Percent of circuits that correctly perform the test program.



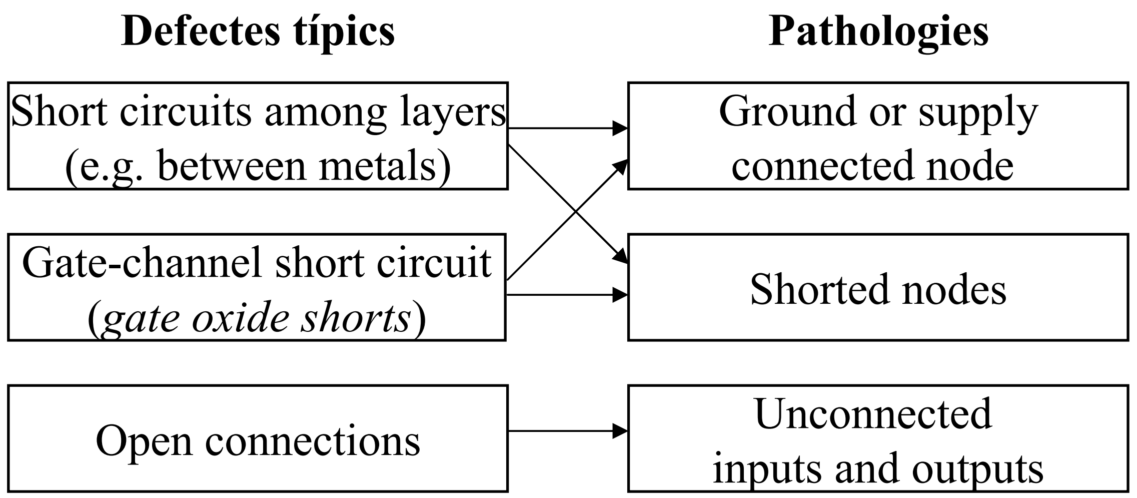
Cost example of exhaustive test.
Finite state machine:



It requires $2^{(n+m)}$ test patterns.
Not possible for most real IC's.

# 5.2 Manufacturing test principles

## 5.2.1. Fault models


Defectes típics
Pathologies
Short circuits among layers (e.g. between metals)
Ground or supply connected node
Gate-channel short circuit (*gate oxide shorts*)
Shorted nodes
Open connections
Unconnected inputs and outputs

**a) S*tuck-at* faults**.

Historically, the most popular model.

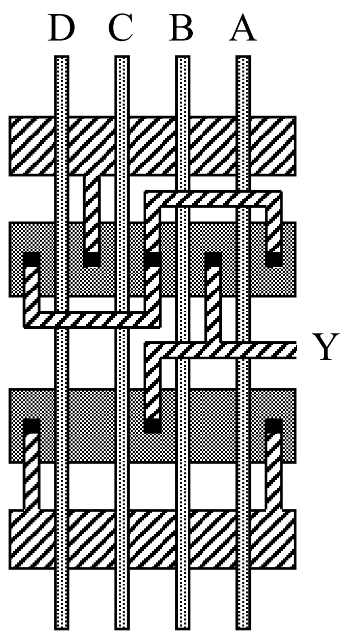
- *Stuck-at-0*: Input value tied to ‘0’, independently of the logic level applied from the circuit that controls its input.
- *Stuck-at-1*: Input tied to ‘1’, independently of the logic level applied from the circuit that controls its input.

Physical causes:

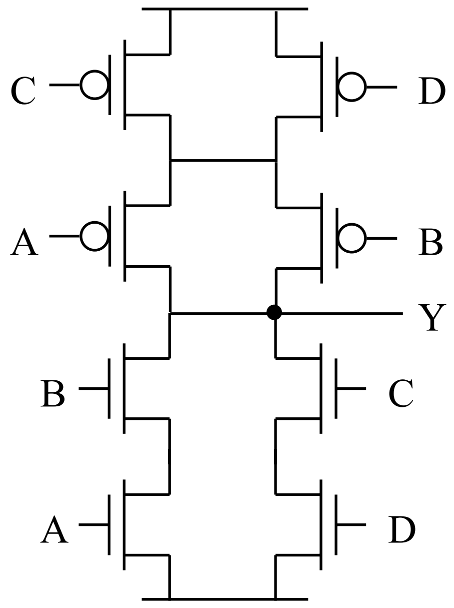
- Metal shorts (with $V_{SS}$ or $V_{DD}$)
- Gate-channel shorts (NMOS gate with $V_{SS}$ or PMOS gate with $V_{DD}$).

**b) B*ridge* or open circuit faults**.

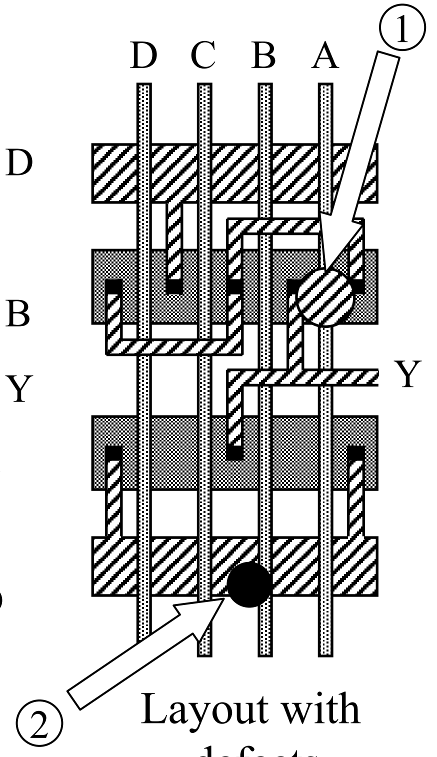
More general than stuck-at.



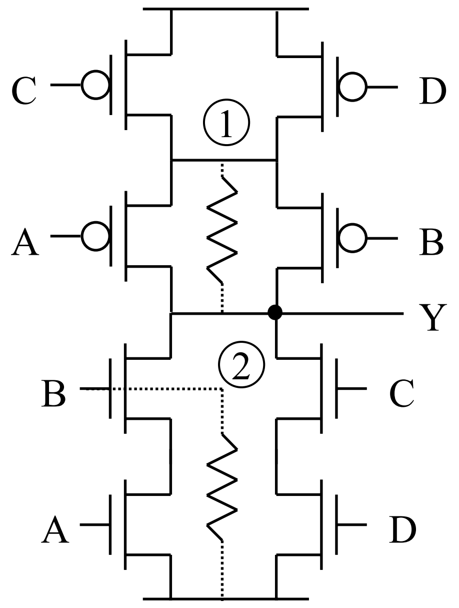
Correct layout



Correct circuit



Layout with defects

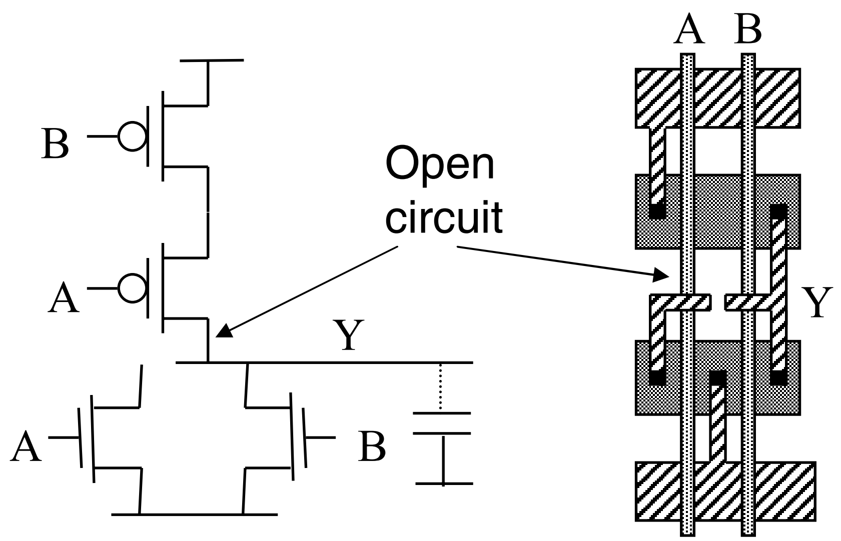


Circuit with defects

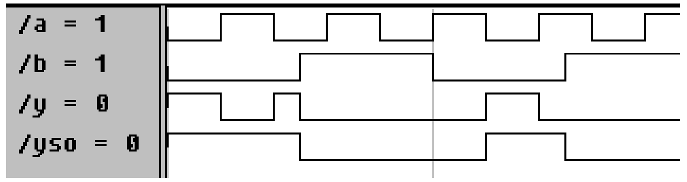
## 5.2 Manufacturing test principles

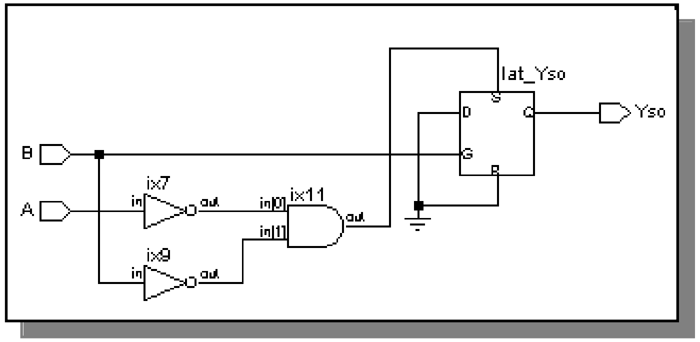
To cope with short or open faults transistor level models are required (or, at least, switch-level).

Example: Sequential fault.



```
correct: process (A, B)
begin
        Y <= A nor B;
end process;
stuck_open: process (A, B)
begin
        if A='0' and B = '0' then Yso <= '1';
        elsif B = '1' then Yso <= '0';
        end if;
end process;
```





$Y = (\overline{A + B}) + A\cdot\overline{B}\cdot Y^{-}$ ➔ depends on previous state

## 5.2.2. Fault coverage

*Fault coverage*: Fault percent that a test program can detect.

*Fault coverage* calculation:
1 Simulate
- The correct circuit
  A single-fault circuit (e.g., stuck-at-0 at a single node).
- Fault is detect if some difference appears in an output node during the test.

2 Repeat the process for all possible circuit fault.

High CPU time ➔ *stuck-at* fault model.
Problem: The *stuck-at* model does not guarantee every fault detection

## 5.2.3. Controlability and Observability

*Controlability* of a circuit node: Degree of simplicity/difficulty to fix its logic value to 0 or 1 applying logic values to inputs

*Observability* of a node: Degree of simplicity/difficulty to observe its logic value (propagating it to an output node).

There exist design techniques that make nodes easily controlables and observables.

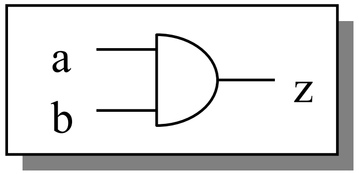
A possible definition foe **SCOAP** controlability and observability, is shown in the following.

## 5.2.4. SCOAP algorithm. Method of controlability and observability assignment.

- Static (combinational). Definitions:
  *CC0(n)*. Node n combinational controlability to 0.
  *CC1(n)*. Node n combinational controlability to 1.
  *CO(n)*. Node n combinational observability.

- Dynamic (sequential). Definitions:
  *SC0(n)*. Node n sequential controlability to 0.
  *SC1(n)*. Node n sequential controlability to 1.
  *SO(n)*. Node n sequential observability.

Controlability and observability calculation example: AND gate



$$\begin{aligned}
CC1(z) &\equiv CC1(a) + CC1(b) + 1 \\
CC0(z) &\equiv \min[CC0(a),CC0(b)] + 1 \\
CO(a) &\equiv CO(z) + CC1(b) + 1
\end{aligned}$$

1 is added to static definitions because the AND gate represents a combinational logic level.

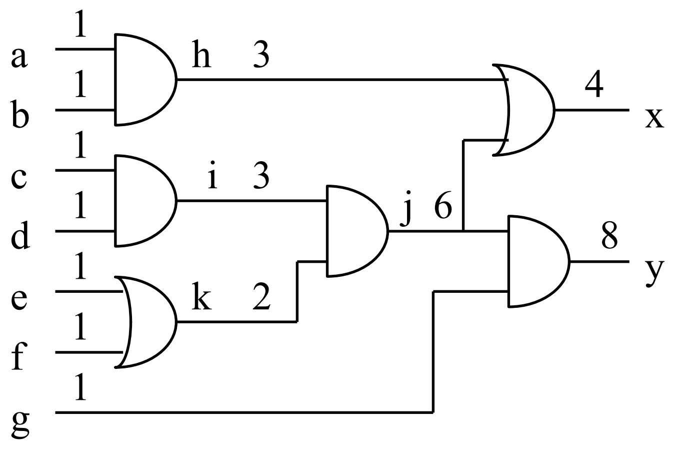
$$\begin{aligned}
SC1(z) &\equiv SC1(a) + SC1(b) \\
SC0(z) &\equiv \min[SC0(a),SC0(b)] \\
SO(a) &\equiv SO(z) + SC1(B)
\end{aligned}$$

Circuit calculation process:

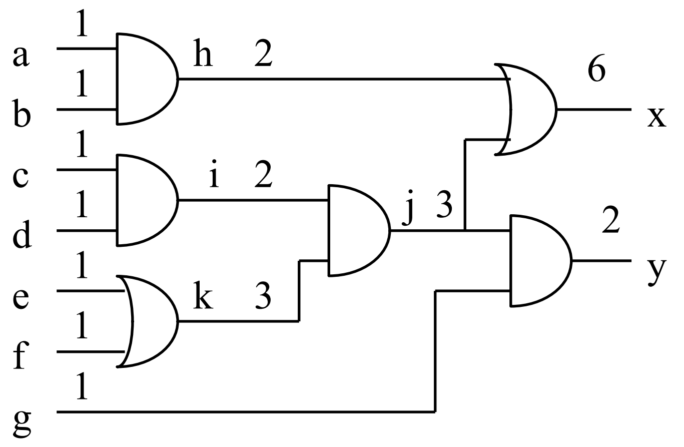
1. Controlabilities are calculated starting from the primary inputs. (CC0 ≡ 1, CC1 ≡ 1).
2. Observabilities are calculated starting form the primary outputs CO ≡ 0.

Example.

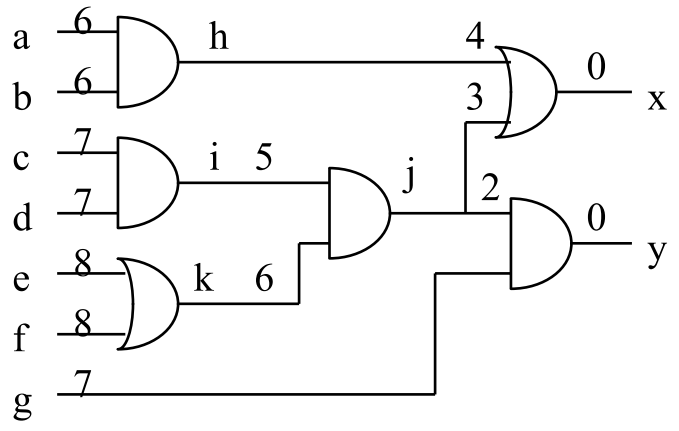
Combinational controlability to 1 (CC1):



Combinational controlability to 0 (CC0):



Combinational observability (CO):



## 5.2 Manufacturing test principles

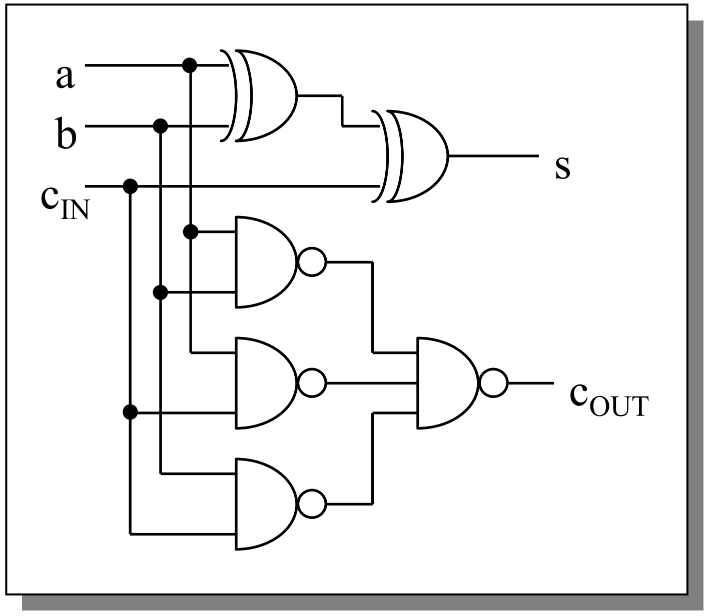
Observations:

- For multiple fanout output, the smaller observability measure is taken.
- High controlability (observability) values indicate difficulty to control (observe).
- Good testability is related with good controlability and observability, but not directly. Exceptions: Redundant circuits.

**Exercise**

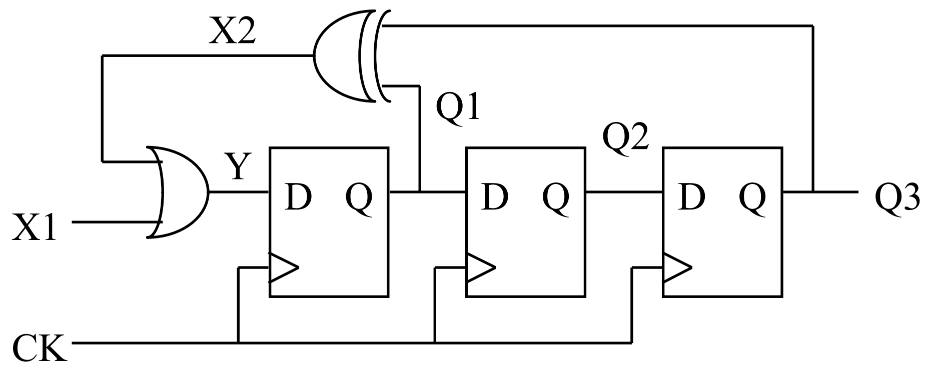
a) Calculate CC0, CC1 i CO for an XOR gate and for 2- 3-input NAND gates.
b) Calculate CC0, CC1 i CO of a full adder.
c) Obtain for an n-bit carry propagate adder the maximum value of
CC0, CC1 i CO as a function of n.

Justify the answers.



## 5.2 Manufacturing test principles

### SCOAP calculation example for a sequential circuit



| Node | CC0 | CC1 | CO | SC0 | SC1 | SO |
|---|---|---|---|---|---|---|
| CK | 1 | 1 | 29 | 0 | 0 | 10 |
| X1 | 1 | 1 | 20 | 0 | 0 | 7 |
| X2 | 13 | 26 | 8 | 4 | 8 | 3 |
| Y | 15 | 2 | 6 | 4 | 0 | 3 |
| Q1 | 17 | 4 | 4 | 5 | 1 | 2 |
| Q2 | 19 | 6 | 2 | 6 | 2 | 1 |
| Q3 | 21 | 8 | 0 | 7 | 3 | 0 |

## 5.2.5. Automated Test Pattern Generation (ATPG)

**D algorithm**

Consists in propagating logic level differences between the correct and faulty circuit

5 logic values : 0, 1, D, /D, X

- D: 1 at correct system, 0 at faulty system
- /D: 0 at correct system, 1 at faulty system
- X: Undefined

Examples:

AND function

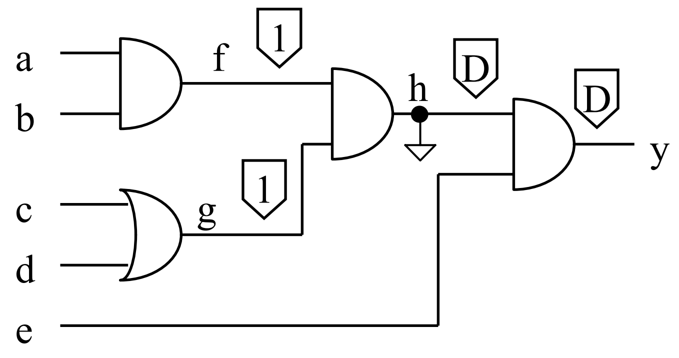
| A\B | 0 | 1 | X | D | /D |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | X | D | /D |
| X | 0 | X | X | X | X |
| D | 0 | D | X | D | 0 |
| /D | 0 | /D | X | 0 | /D |

OR function

| A\B | 0 | 1 | X | D | /D |
|---|---|---|---|---|---|
| 0 | 0 | 1 | X | D | /D |
| 1 | 1 | 1 | 1 | 1 | 1 |
| X | X | 1 | X | X | X |
| D | D | 1 | X | D | 1 |
| /D | /D | 1 | X | 1 | /D |

## 5.2 Manufacturing test principles

Example: S-A-0 (Stuck-at-0) detection at node *h*



*h* has to be set to 1 (to generate D):
[a,b,c,d] = [1,1,1,0],[1,1,0,1] o [1,1,1,1]
and propagate D to the y output:
[e] = [1]

The test pattern is thus
[a,b,c,d,e] = [1,1,1,0,1],
[1,1,0,1,1] or
[1,1,1,1,1].

For all internal nodes S-A-0 and S-A-1:

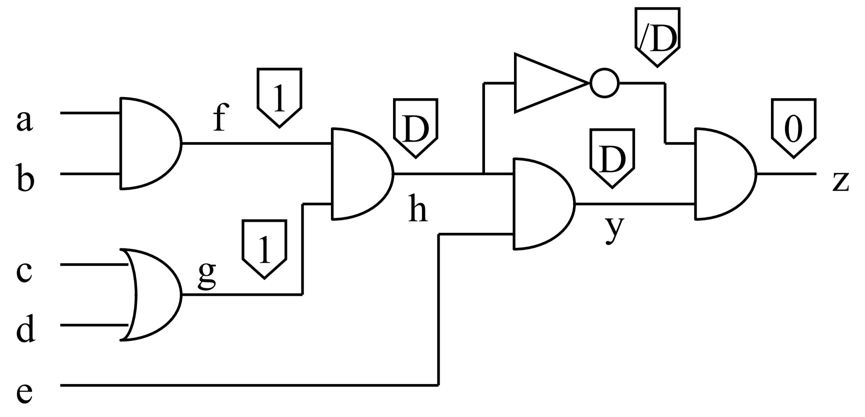
| **Node** | **Test** | **Pattern(s)** |
|---|---|---|
| f | S-A-0 | [1,1,X,1,1], [1,1,1,X,1] |
| f | S-A-1 | [0,X,X,1,1], [0,X,1,X,1], [X,0,X,1,1], [X,0,1,X,1] |
| g | S-A-0 | [1,1,1,X,1], [1,1,X,1,1] |
| g | S-A-1 | [1,1,0,0,1] |
| h | S-A-0 | [1,1,1,X,1], [1,1,X,1,1] |
| h | S-A-1 | [0,X,X,X,1], [X,0,X,X,1], [X,X,0,0,1] |

The pattern set

{[1,1,0,0,1], [1,1,0,1,1], [0,0,1,0,1]}

permits detecting all possible stuck-at faults in nodes *f*, *g* and *h*.

Testability problem in redundant circuits :



Test pattern generation steps:

- Internal node **D value propagation** to a primary output (**sensitized path**). As a function of previously-calculated controlability figures, the gate path propagation is decided.
- **Primary input values determination** to uncover the fault and make it observable. This is done by **backtracing** from the faulty node and the sensitized path to the primary inputs using the controlability indexes of each node.

## 5.3. Design for testability

### 5.3.1. Design For Testability (DFT)

DFT techniques:

- Ad-hoc techniques
- *Scan-path* type
- Self test and *built-in* test

### 5.3.2. Ad-hoc test

Collection of ideas to improve controlability and observability. It comprises:

- Large sequential circuit partition
- Test point insertion
- Multiplexer insertion
- Reset state

**Typical examples**:

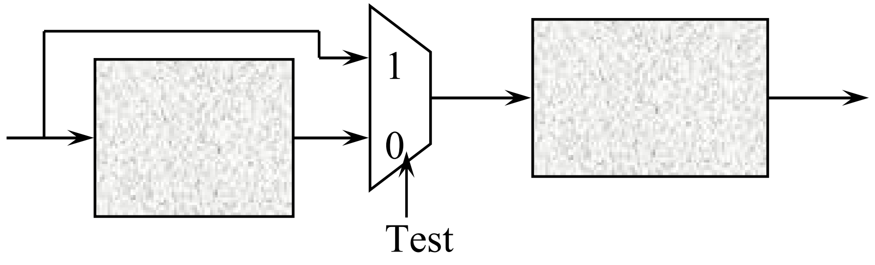
Large counters.

- Parallel load allows controlling easily the internal state.
- Intermediate point observation. Permits reducing the test length.

Bus-based systems.

- To allow applying inputs an reading outputs from all the elements connected to the bus.

Subsystem partition with multiplexers:



Reset need:
• Test
• System initialization to a known state

Ad-hoc techniques are fine for reduced complexity designs.

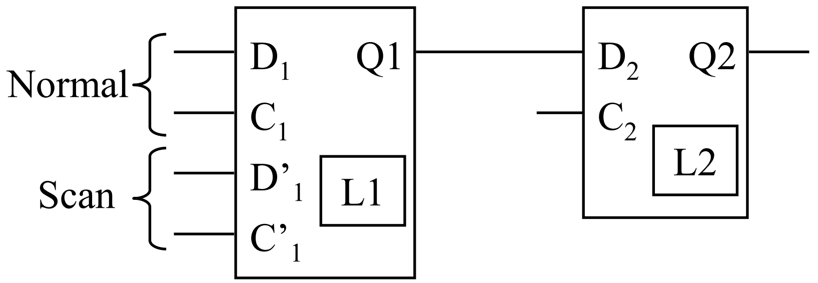
Complex designs demand **systematic approaches**.

## 5.3.3. Scan-based test

Consists in substituting the circuit flip-flops by dual input and input-clock **flip-flops**.

This permits building a ***scan chain*** independent of the system normal operation..

Basic block :



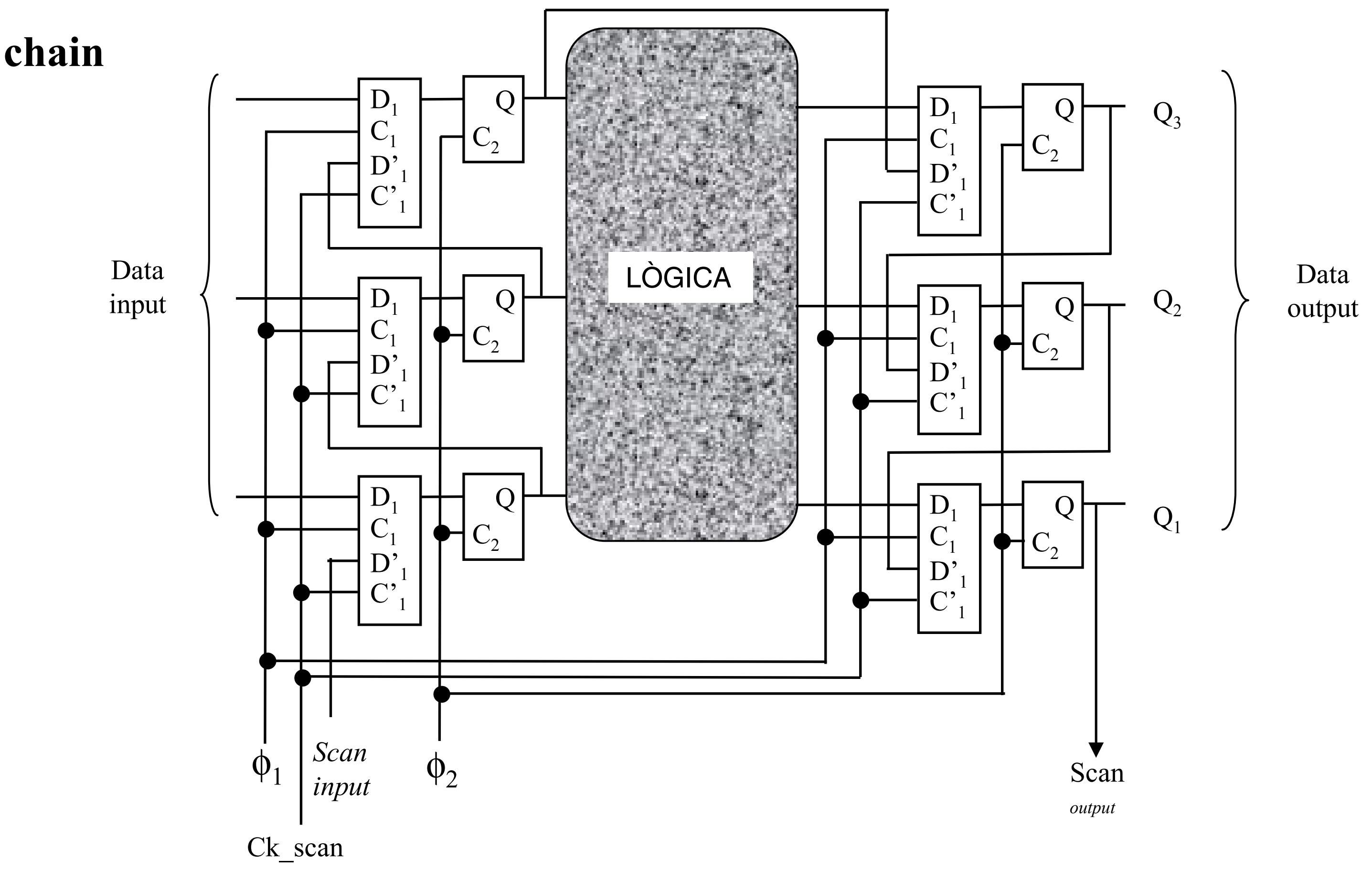
L1: *Master*. Double input port.
- $D_1$: Data input
- $C_1$: Data clock (phase 1)
- $D'_1$ : Scan input
- $C'_1$: Scan clock (phase 1)

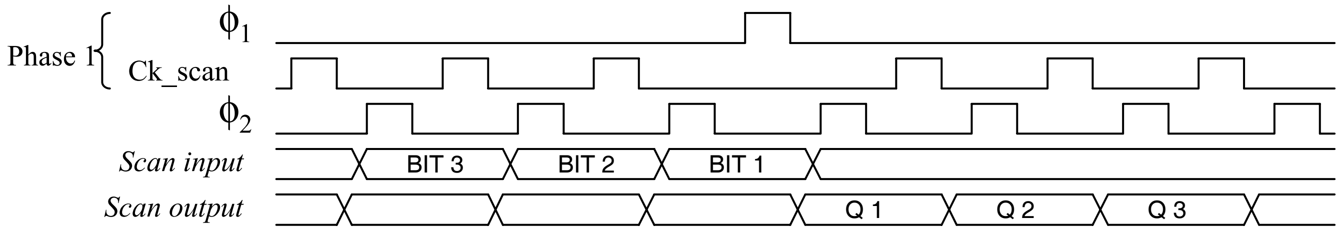
$C_1$ i $C'_1$ cannot be simultaneously active.

L2: *Slave*
- $D_2$: Data input
- $C_2$: Data/scan clock (phase 2)

## 5.3. Design for testability

### Scan chain





# 5.4. Self test

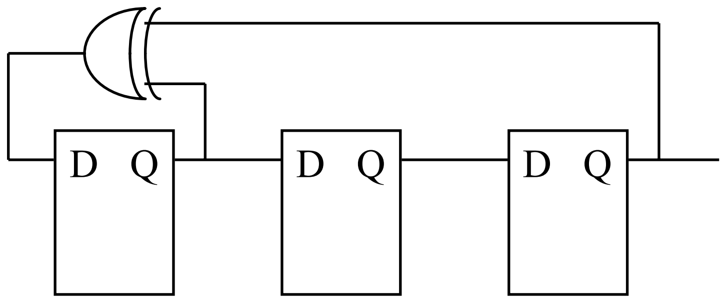
Test-specialized circuits are included into the IC for self verification. It requires:

- Test pattern generation
- Correct output verification (signature analysis)

## 5.4.1. Pseudo-random sequence generation

Pseudo-random sequence generator: LFSR (Linear Feedback Shift Register).

3 bit LFSR example:



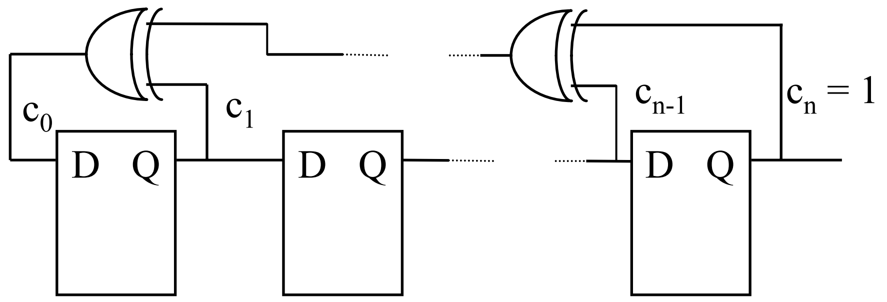
- Low-cost test pattern generation method
- Also used for signature analysis.
- *For* n flip-flops*, $2^n$ - 1* vectors (states) are generated before the state repeats.

## 5.4. Self test

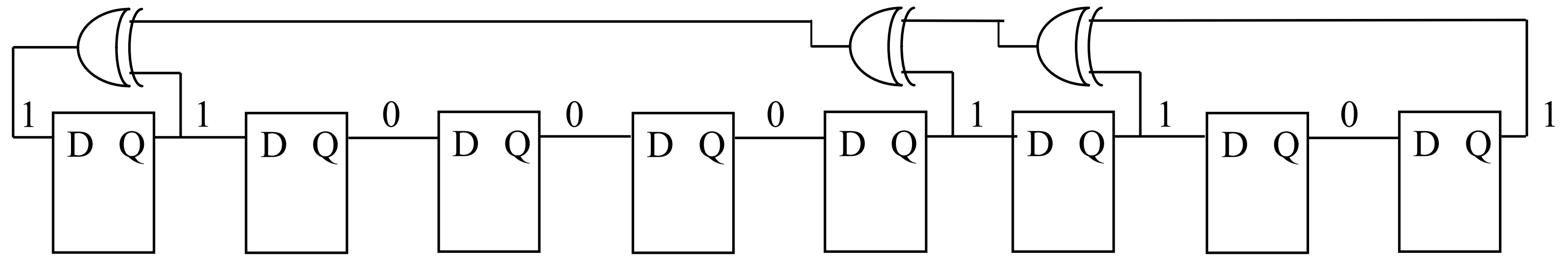
LFSR coefficient table:

| n | s | Binary |
|---|---|---|
| 1 | 0,1 | 11 |
| 2 | 0,1,2 | 111 |
| 3 | 0,1,3 | 1101 |
| 4 | 0,1,4 | 11001 |
| 5 | 0,2,5 | 101001 |
| 6 | 0,1,6 | 1100001 |
| 7 | 0,1,7 | 11000001 |
| 8 | 0,1,5,6,8 | 110001101 |
| 9 | 0,4,9 | 1000100001 |
| 10 | 0,3,10 | 10010000001 |
| 16 | 0,2,3,5,16 | 10110100000000001 |
| 32 | 0,1,27,28,32 | 1100000000000000--<br>0000000000110001 |

Circuit structure:

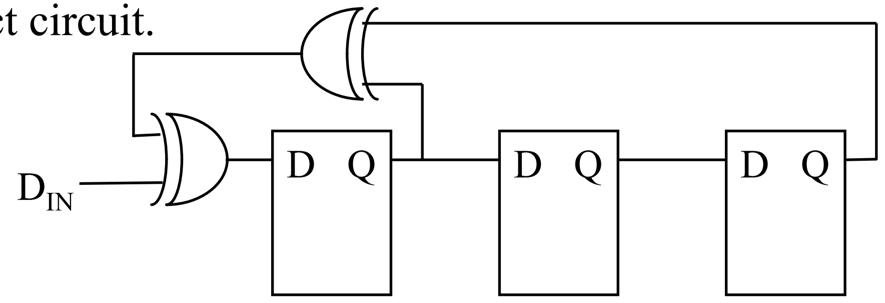


Example: 8-bit LFSR



## 5.4.2. Signature analysis

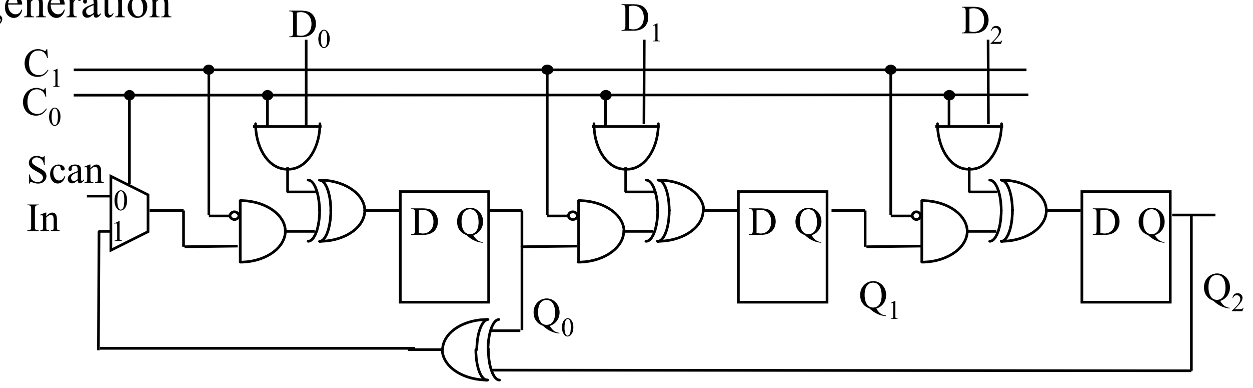
Cyclic sum of circuit under test (CUT) outputs.
For each test pattern (e.g., generated with an LFSR), outputs are XOR'ed with previous results.
**Syndrome**: Test sequence resulting number. Can be stored in ROM.
A faulty circuit circuit will generate, with very high probability, a syndrome value different than the correct circuit.
Example:



## 5.4.3. Built-In Logic Block Observation (BILBO)

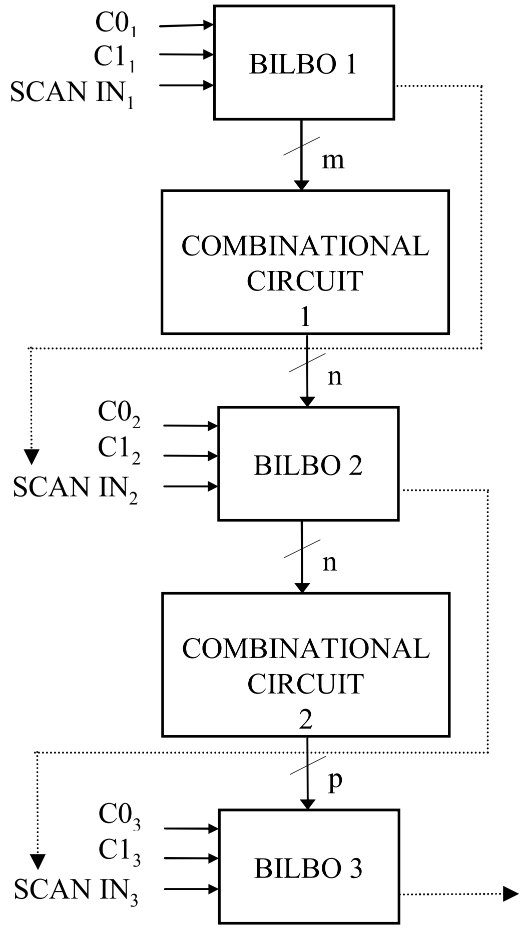
It combines:

- Pseudorandom sequence generation
- Signature analysis
- Shift register



| MODE | $C_0$ | $C_1$ | |
|---|---|---|---|
| A | 0 | 0 | Scan |
| B | 0 | 1 | Reset |
| C | 1 | 0 | LFSR or signature analysis |
| D | 1 | 1 | Parallel register |

## Example: Registers replaced by BILBO in a pipeline.



# 5.5. System-level test

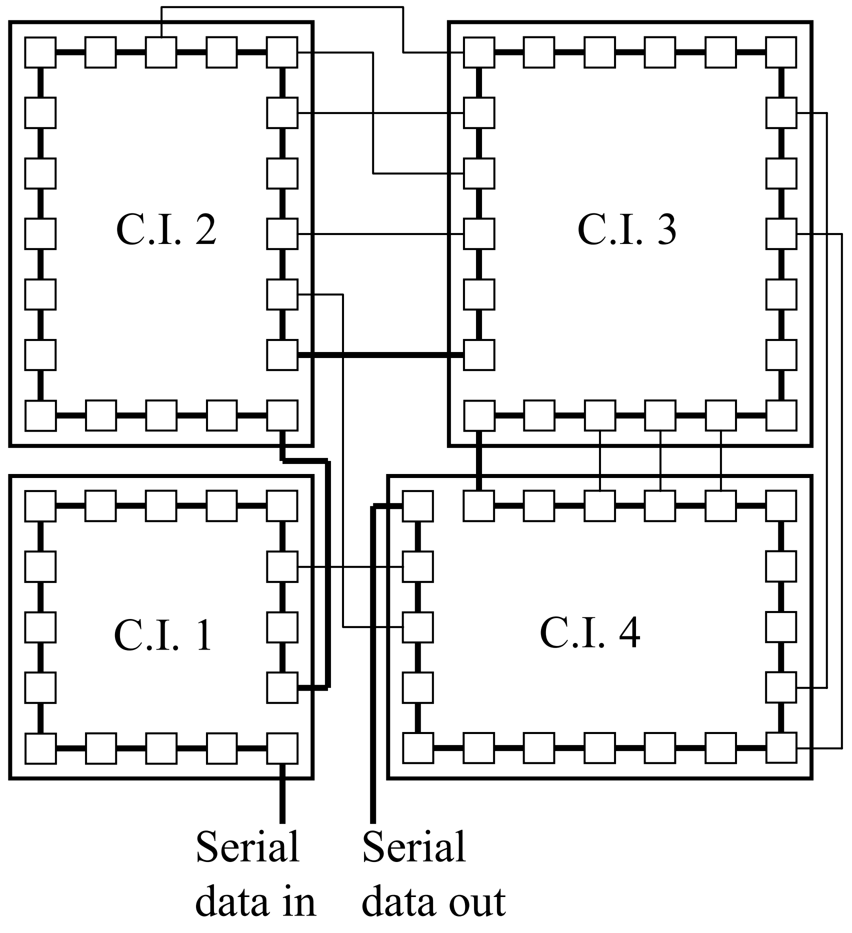
## 5.5.1. Boundary Scan

Unified methodology based on *scan* to verify ICs at PCB level.

Standard IEEE 1149 (JTAG): It defines a Boundary Scan architecture.

I/Os can be interconnected to form shift registers. This allows:

- Connectivity tests among components
- I/Os capturing and loading
- Self-test data distribution and result extraction.

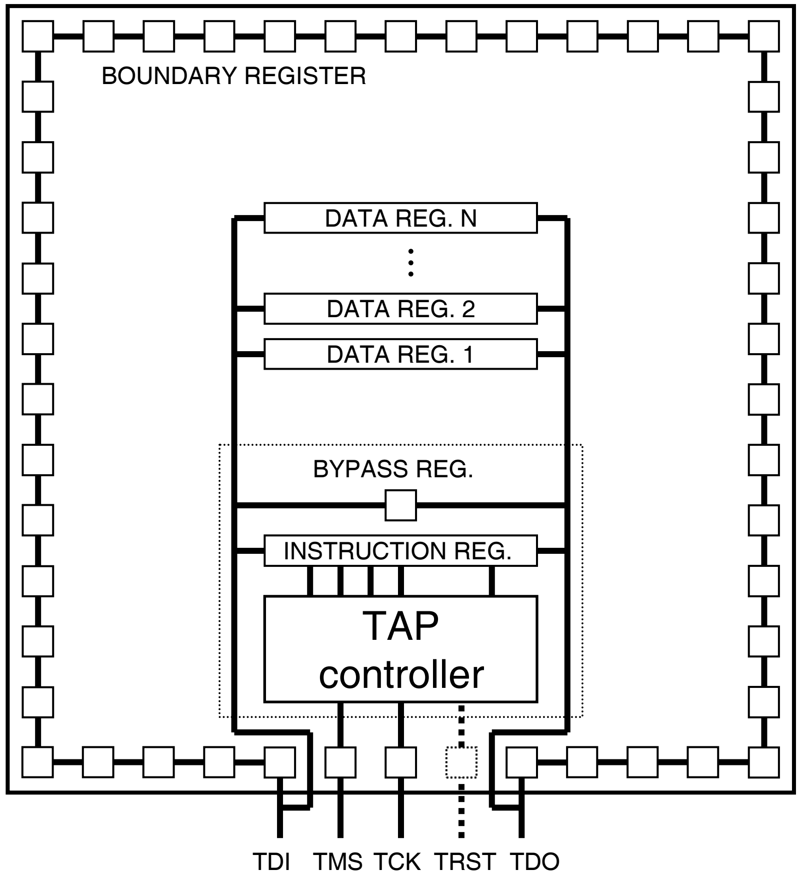


## 5.5.2. Boundary Scan Architecture

Consists of:

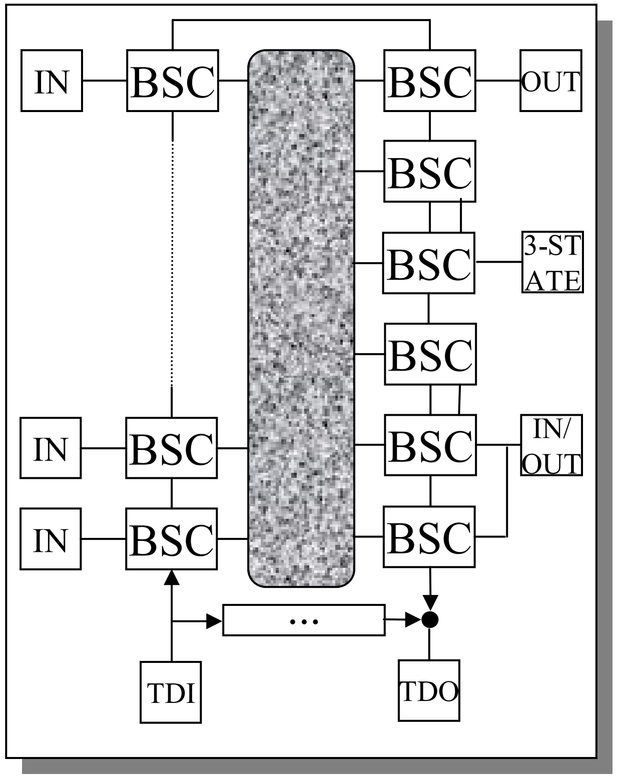
- Boundary register
- Data register
- Test Access Port (TAP)
  - Instruction register
  - Bypass register
  - TAP controller
- I/O pins: TDI, TDO, TMS, TCK
  - Optionally TRST

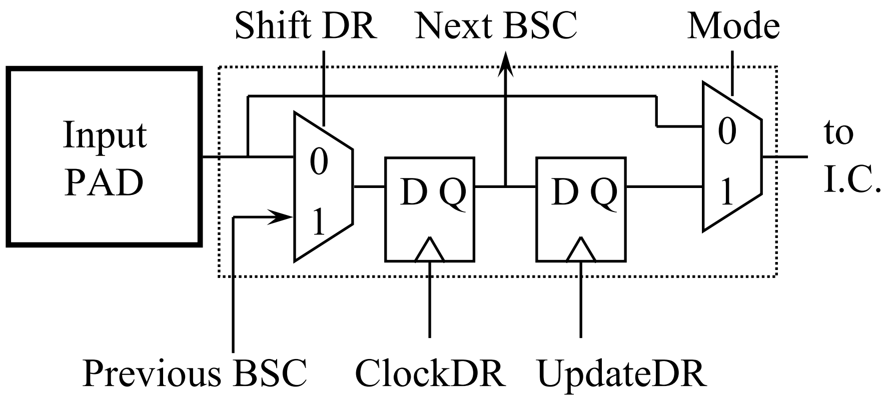
Registers are built with BSCs (Boundary Scan Cells)

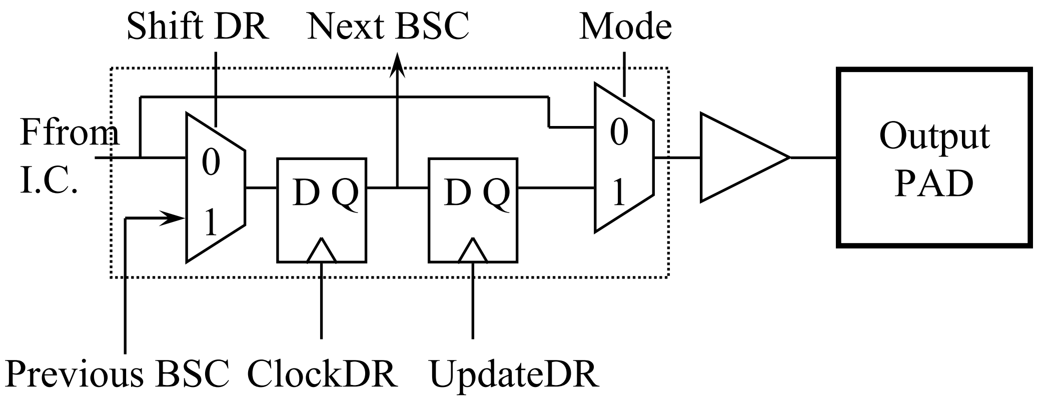


## 5.5.3. Boundary Register

Surrounds the IC through all I/O pads..
Boundary Scan Cells (BSC) are included in pads.

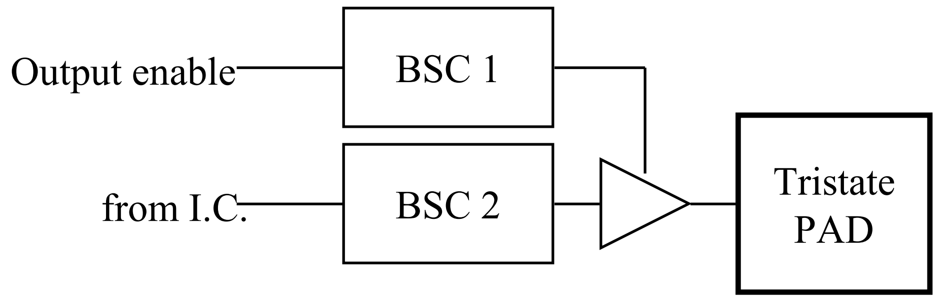




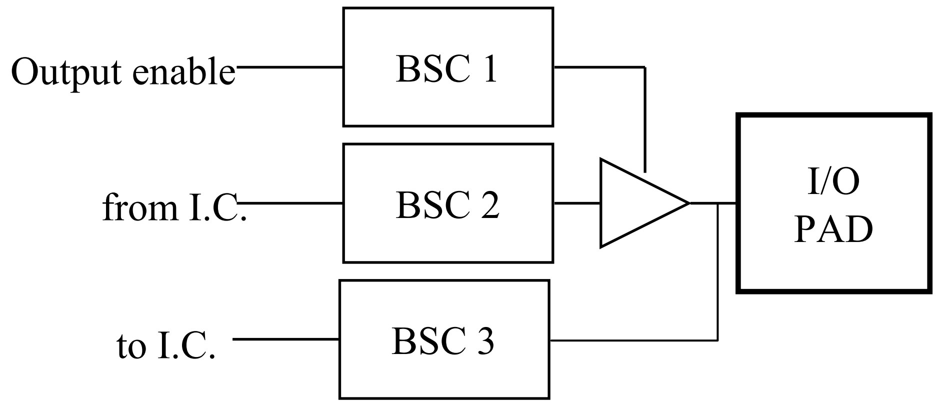


## 5.5. System-level test

For a tristate output pad, two BSCs:are required



In a bidirectional pad, three BSCs are required:



## 5.5.4. Test Access Port (TAP)

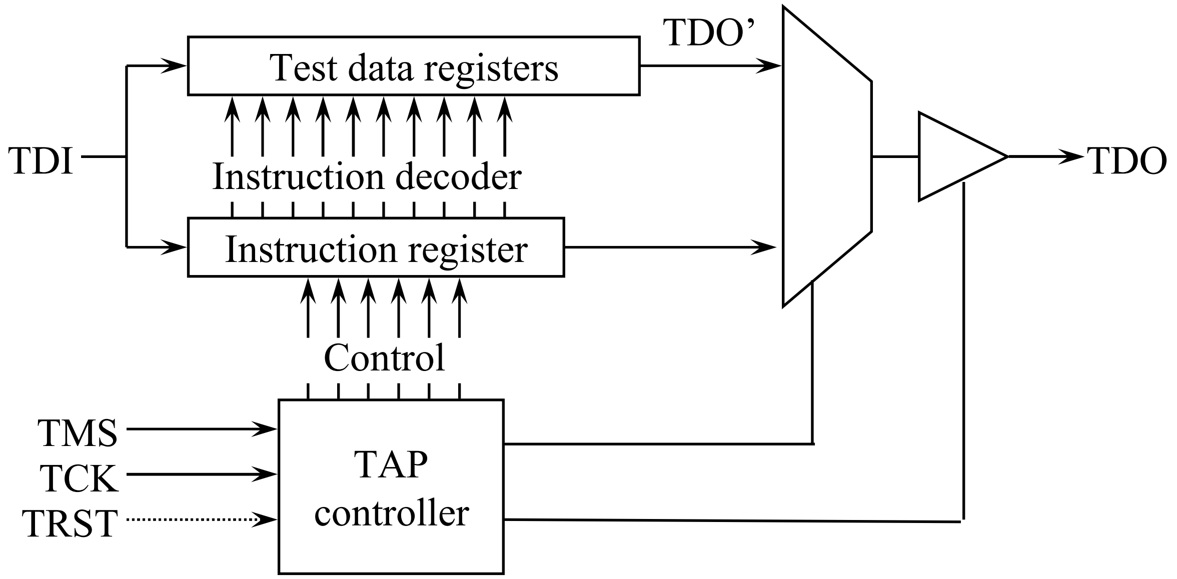
Interface to be included in an IC to support the IEEE 1149 architecture.
Four 1-bit lines (+ optional one) :

- TCK (Test Clock Input). Clock during test.
- TMS (Test Mode Select). Controls test operations
- TDI (Test Data Input). Serial in test data line.
- TDO (Test Data Output). Serial out test data line.

- TRST (Test Reset). TAP controller asynchronous reset optional line.

**Architecture**

- TAP interface pins
- Test data register
- Instruction register
- TAP controller

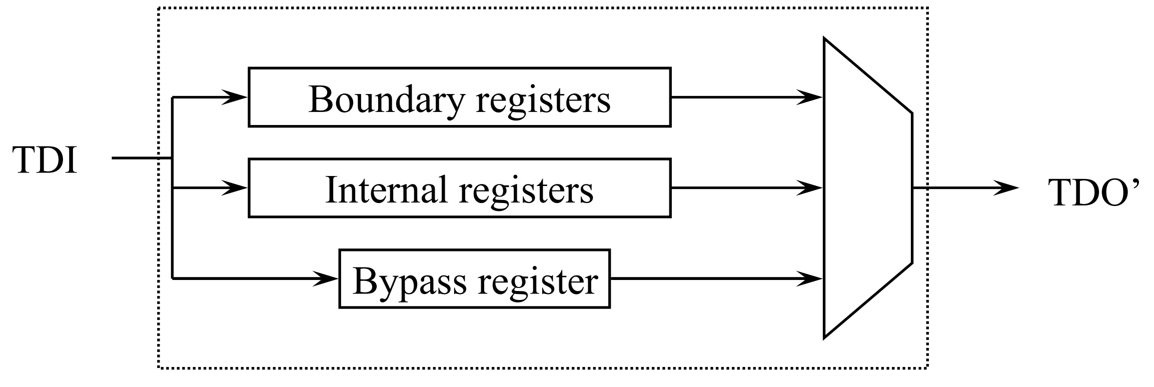


## Test data registers

Transmit test vectors to the combinational circuits and capture the outputs, that are sifted to the serial test data output line (TDO).

They can be:

- Boundary: Formed by BSCs
- Internal: Internal logic registers
- Bypass (1 bit): Connects TDI to TDO with 1 clock period delay.



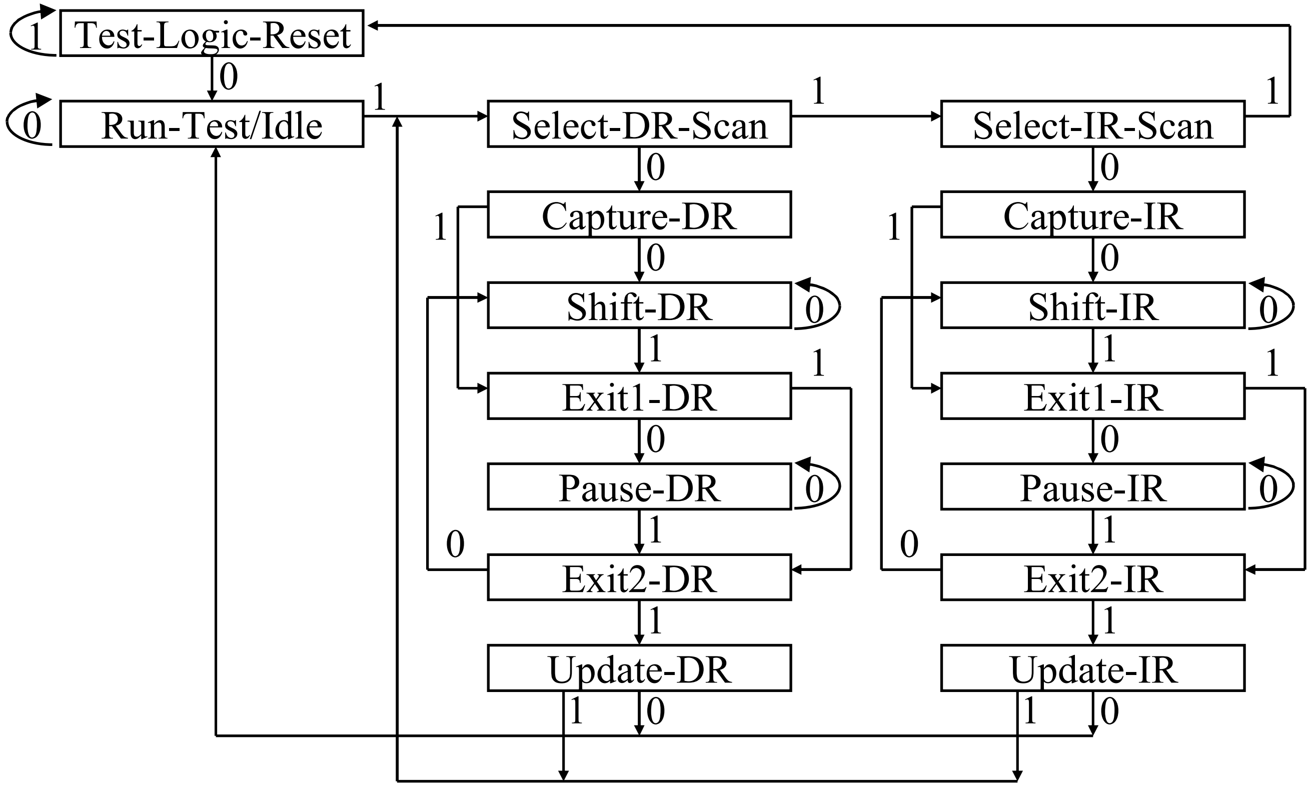
Test data registers

## TAP controller

16-state Finite State Machine (FSM)

- TCK rising edge transition
- Control variable: TMS

State diagram:

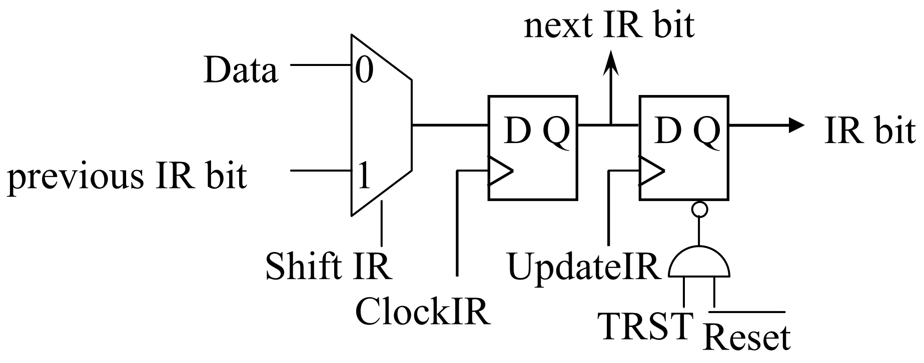


## 5.5.5. Instruction register

Minimum length: 2 bits.
3 mandatory instructions:
- BYPASS. Code IR = 1···1. Programs IC bypass selecting the bypass 1-bit register. Thus, a system IC can be tested without the delay of the complete test chain.
- EXTEST. Code IR = 0···0. External circuit test.
- SAMPLE/PRELOAD. Selects the boundary registers as a DR chain and preloads or samples IC I/Os.

The following instructions are recommended:
- INTEST. Internal circuits test step through boundary registers.
- RUNBIST. Triggers the IC self-test circuits.

Diagram of one bit of the IR:


next IR bit
Data
0
previous IR bit
1
D Q
D Q
IR bit
Shift IR
ClockIR
UpdateIR
TRST
$\overline{\text{Reset}}$