Chapter 6 VLSI Testing

Jin-Fu Li
Advanced Reliable Systems (ARES) Laboratory
Department of Electrical Engineering
National Central University
Jungli, Taiwan

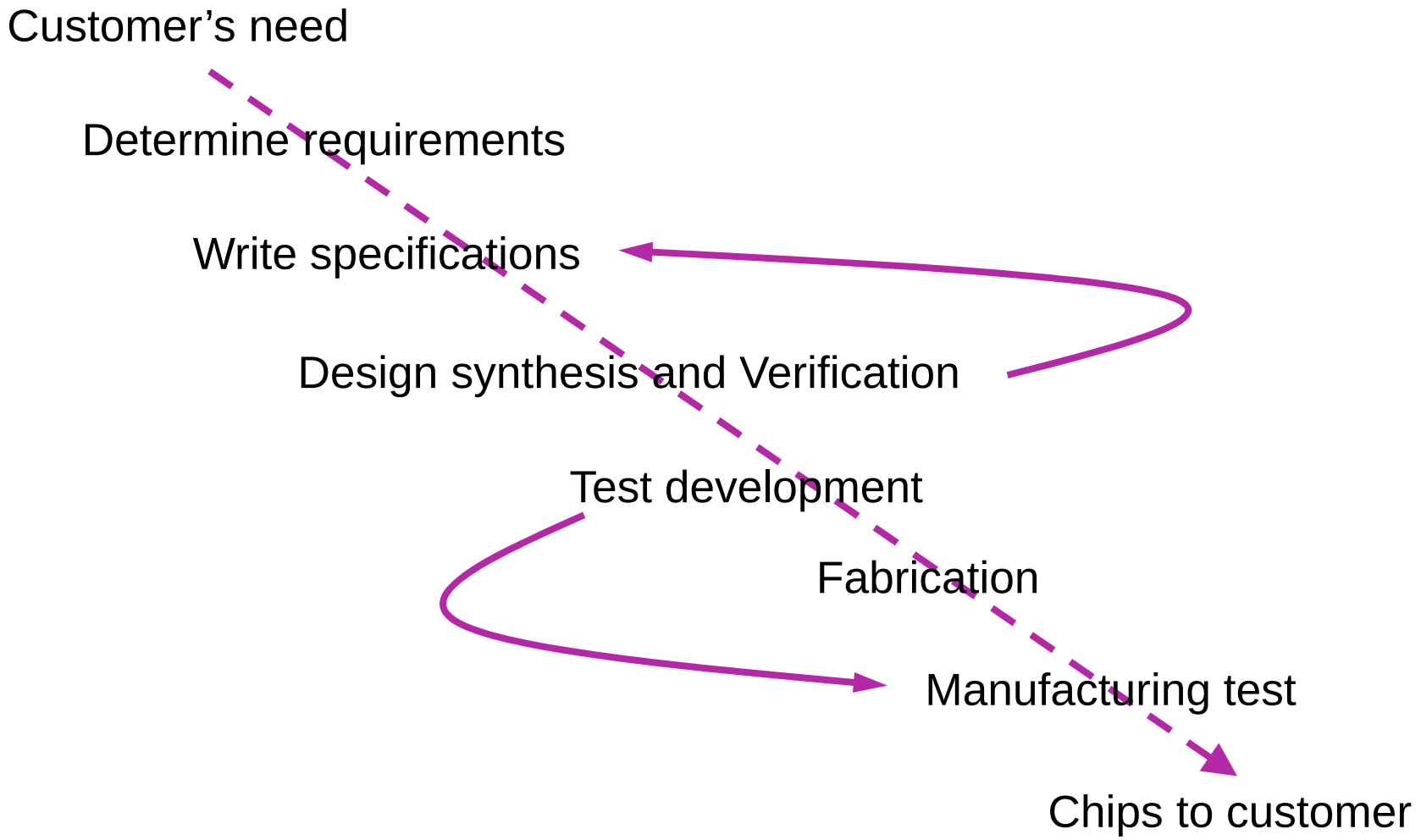
Outline

- Basics
- Fault Modeling
- Design-for-Testability

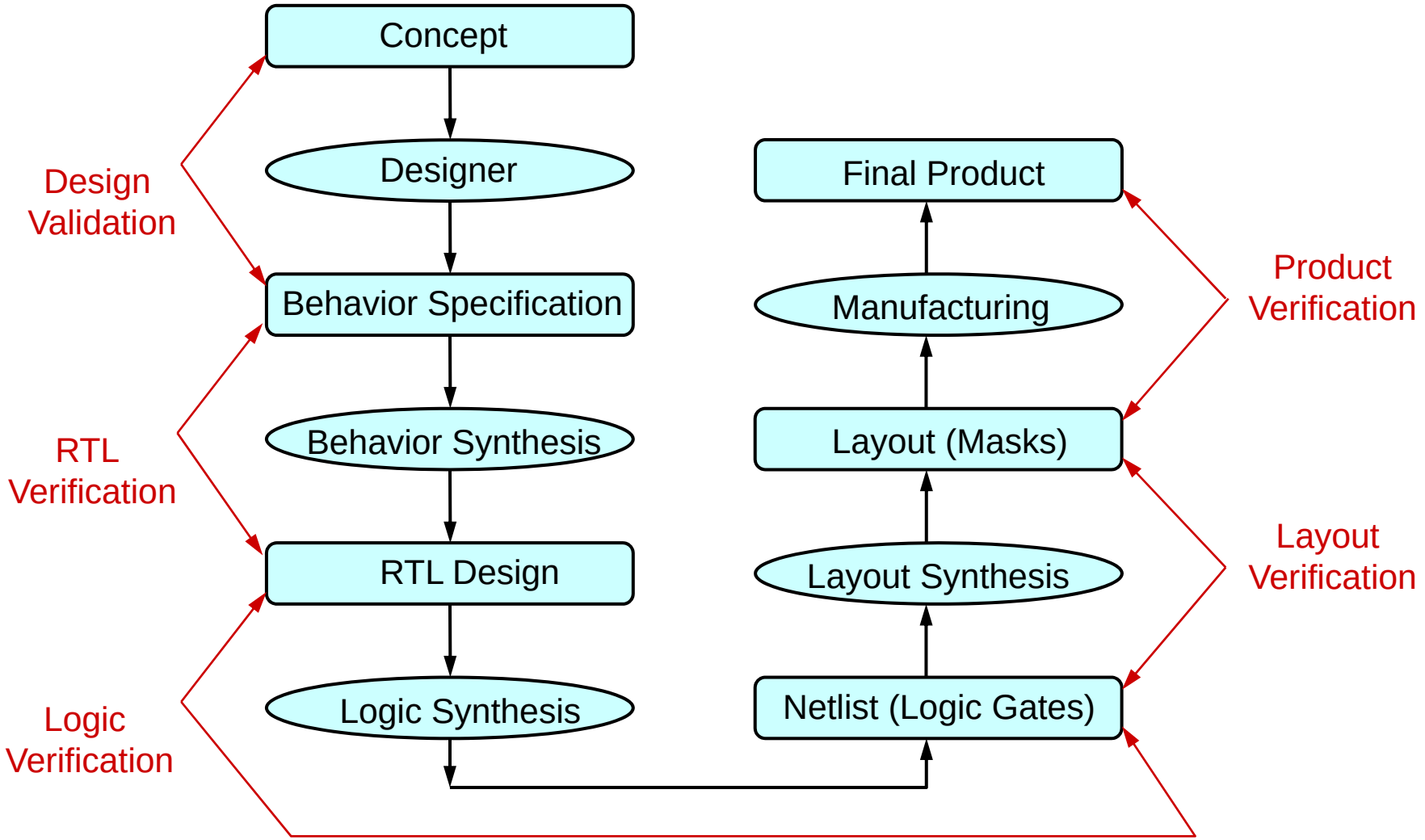
Outline

- Basics
- Fault Modeling
- Design-for-Testability

VLSI Realization Process



VLSI Design Cycle



Role of Testing

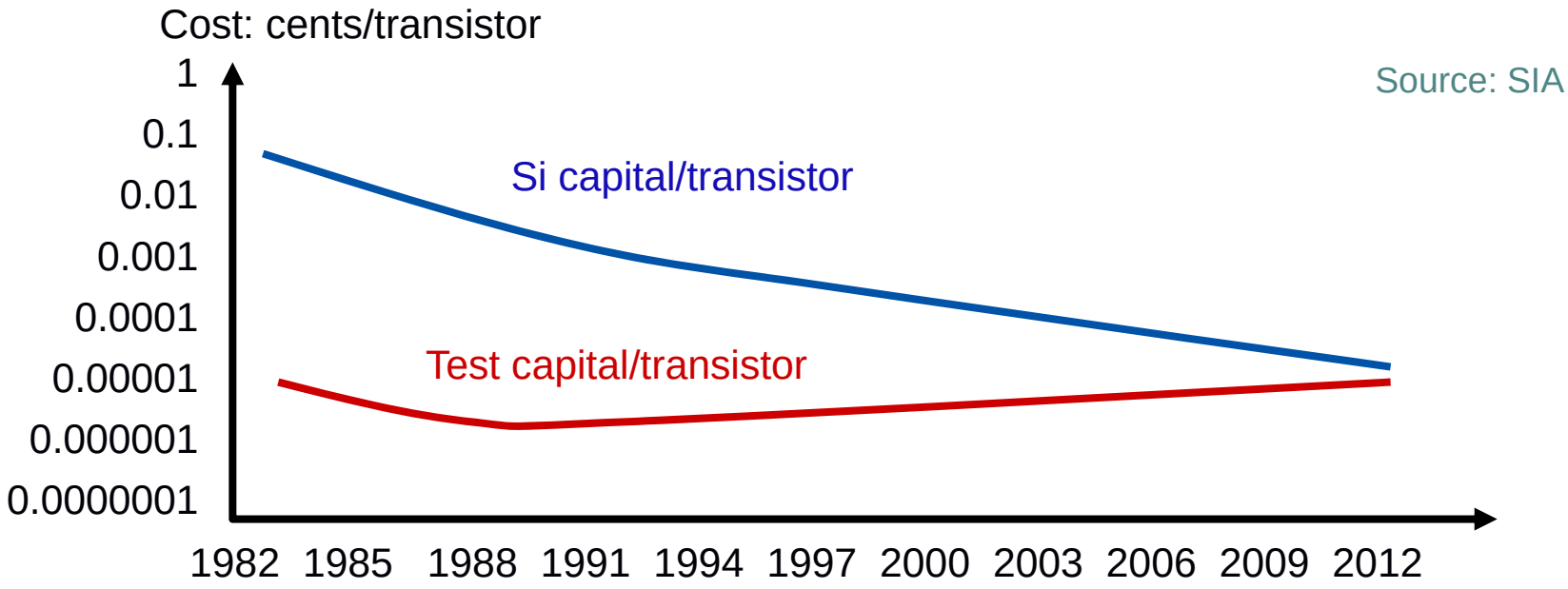
- If you design a product, fabricate, and test it, and it fails the test, then there must be a cause for the failure
  - Test was wrong
  - The fabrication process was faulty
  - The design was incorrect
  - The specification problem
- The role of *testing* is to detect whether something went wrong and the role of *diagnosis* is to determine exactly what went wrong
- Correctness and effectiveness of testing is most important for quality products

Benefits of Testing

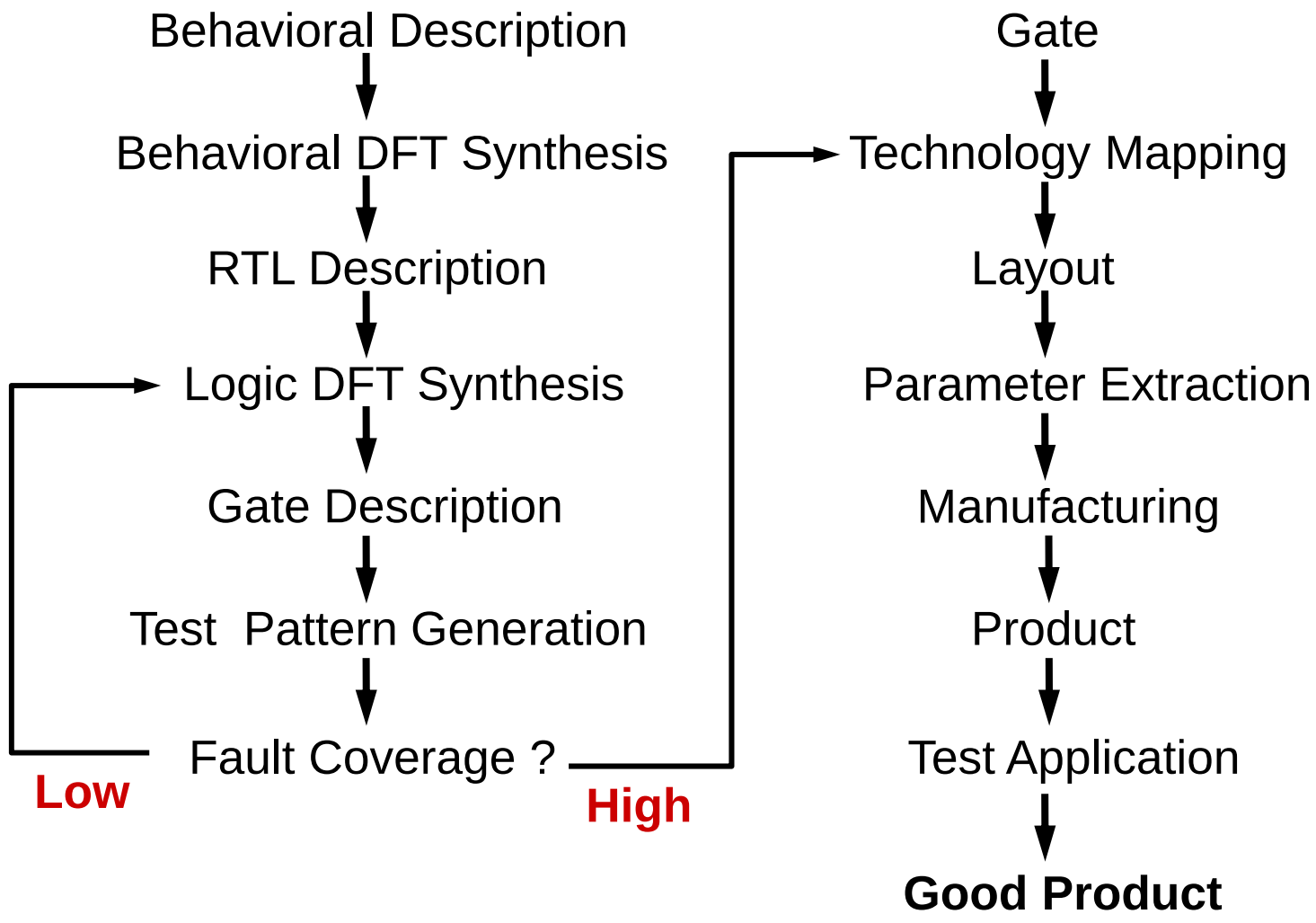
- *Quality* and *economy* are two major benefits of testing
- The two attributes are greatly dependent and can not be defined without the other
- Quality means satisfying the user's needs at a minimum cost
- The purpose of testing is to weed out all bad products before they reach the user
  - The number of bad products heavily affect the price of good products
- A profound understanding of the principles of manufacturing and test is essential for an engineer to design a quality product

Trends of Testing

- Two key factors are changing the way of VLSI ICs testing
  - The manufacturing test cost has been not scaling
  - The effort to generate tests has been growing geometrically along with product complexity



DFT Cycle


Behavioral Description
Behavioral DFT Synthesis
RTL Description
Logic DFT Synthesis
Gate Description
Test Pattern Generation
Fault Coverage ?
Low
High
Gate
Technology Mapping
Layout
Parameter Extraction
Manufacturing
Product
Test Application
Good Product

As Technology Scales Continuously

- Die size, chip yield, and design productivity have so far limited transistor integration in a VLSI design
- Now the focus has shifted to energy consumption, power dissipation, and power delivery
- As technology scales further we will face new challenges, such as variability, single-event upsets (soft errors), and device (transistor performance) degradation– these effects manifesting as inherent ***unreliability*** of the components, posing design and test challenges

Source: S. Borkar (Intel Corp.), *IEEE Micro*, 2005

Possible Solution to Conquer Unreliability

- *The key to the reliability problem might be to exploit the abundance of transistors—use Moore's low to advantage. Instead of relying on higher and higher frequency of operation to deliver higher performance, a shift toward **parallelism** to deliver higher performance is in order, and thus **multi** might be the solution at all levels—from multiplicity of functional blocks to multiple processor cores in a system*

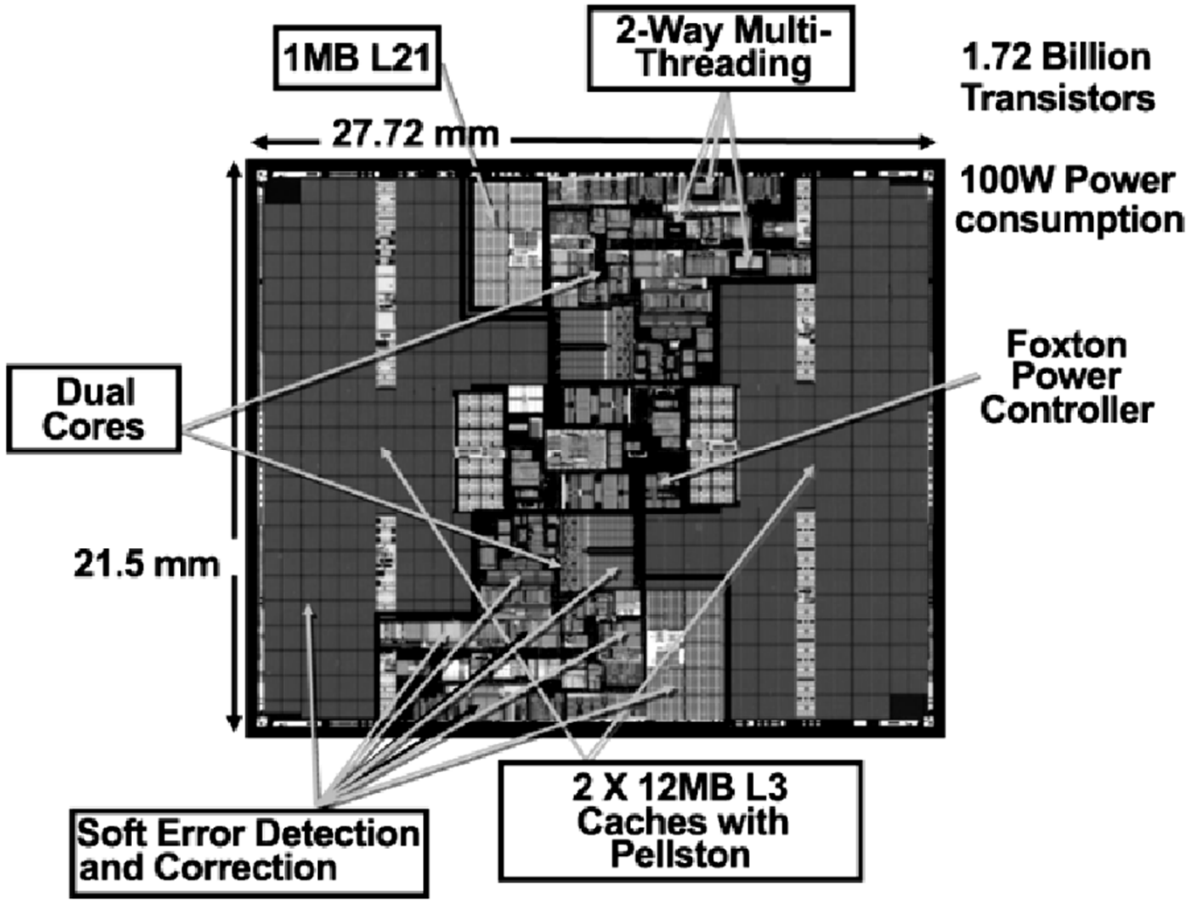
Source: S. Borkar (Intel Corp.), *IEEE Micro*, 2005

Possible Solution to Conquer Unreliability

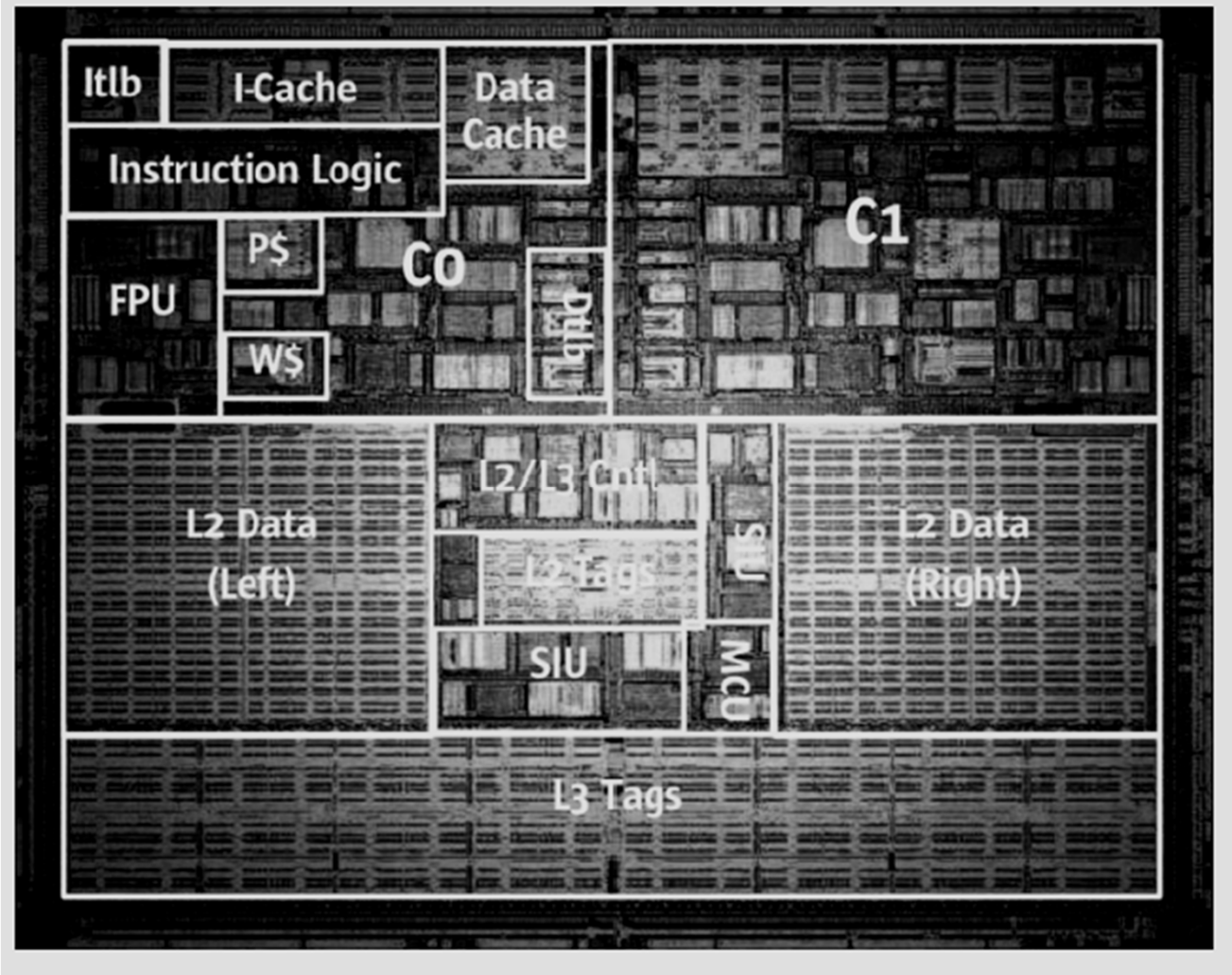
- *We could distribute test functionality as a part of the hardware to dynamically detect errors, or to correct and isolate aging and faulty hardware. Or a subset of cores in the multicore design could perform this work. This microarchitecture strategy, with multicores to assist in redundancy, is called* ***resilient microarchitecture****. It continuously detects errors, isolates faults, confines faults, reconfigures the hardware, and thus adapts. If we can make such a strategy work, there is no need for on-time factory testing, burn in, since the system is capable of testing and reconfiguring itself to make itself work reliably throughout its lifetime.*

Source: S. Borkar (Intel Corp.), *IEEE Micro*, 2005

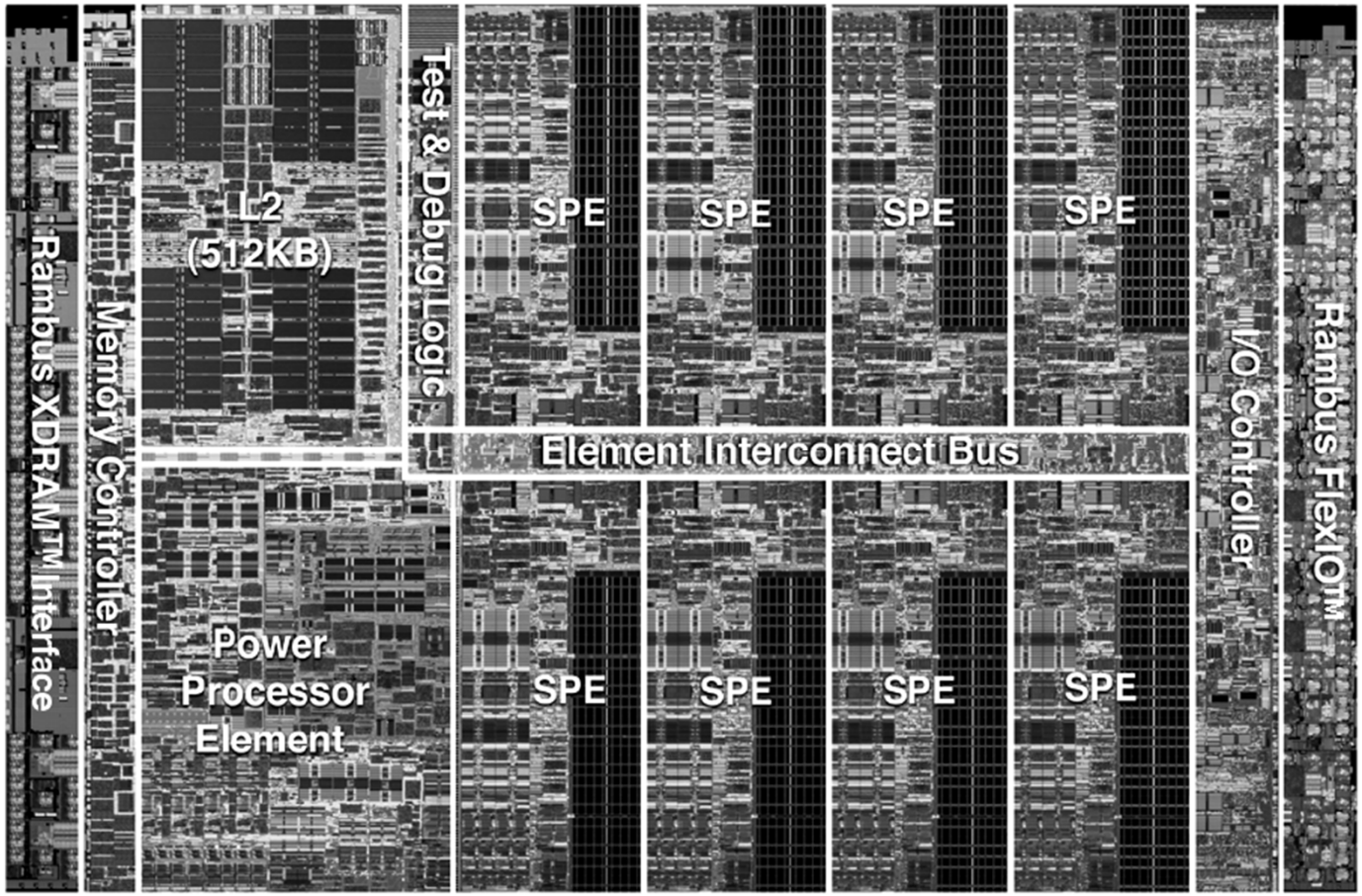
Itanium (JSSC, Jan. 2006)



SPARC V9 (JSSC, Jan. 2006)



Cell Processor (JSSC, Jan. 2006)



Verification & Test

Verification

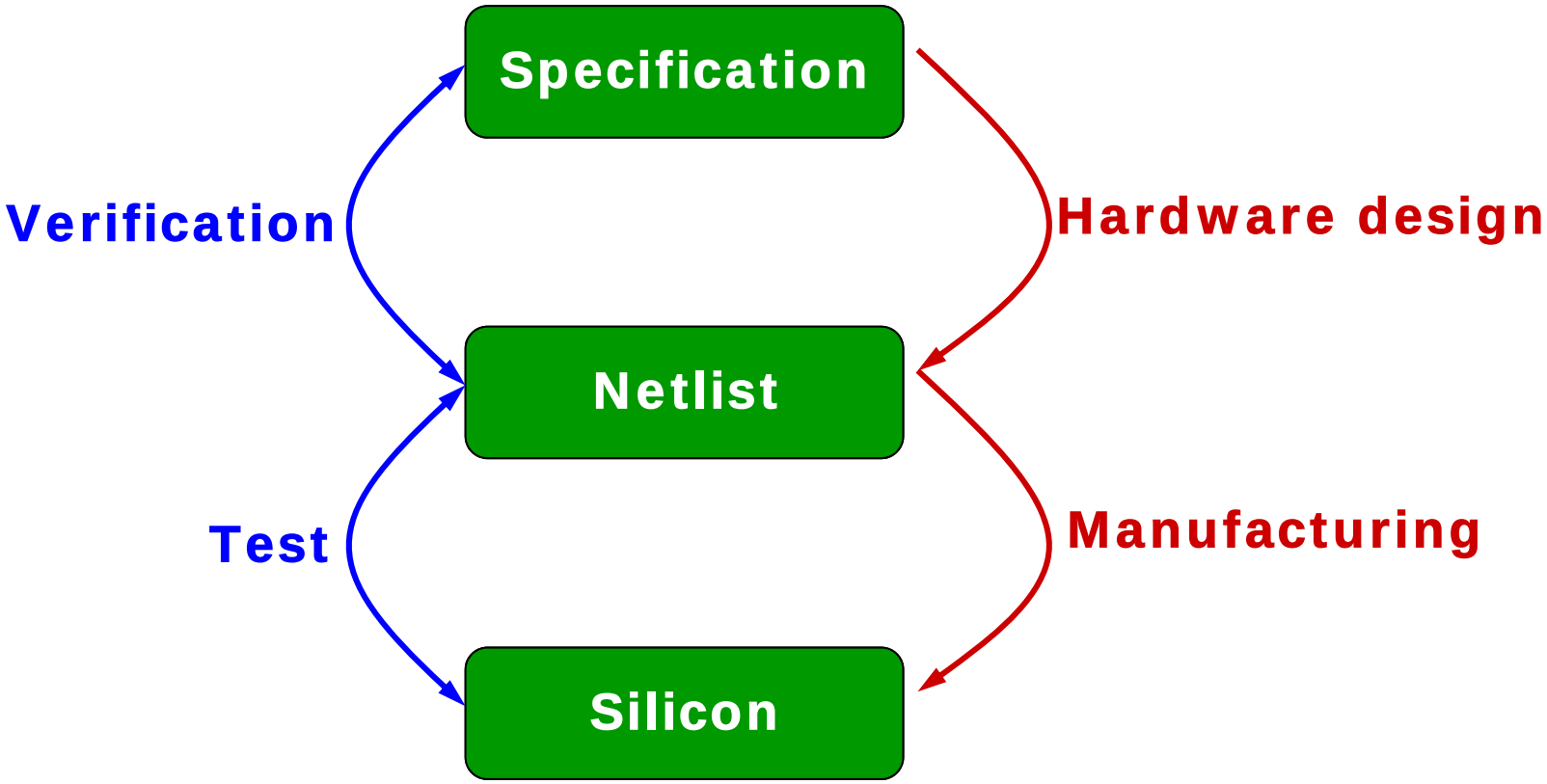
- Verifies correctness of design
- Performed by simulation, hardware emulation, or formal methods
- Perform once before manufacturing
- Responsible for quality of design

Test

- Verifies correctness of manufactured hardware
- Two-part process
  - Test generation: software process executed once during design
  - Test application: electrical tests applied to hardware
- Test application performed on every manufactured device
- Responsible for quality of device

Verification & Test

- Reconvergent path model



Defect, Fault, and Error

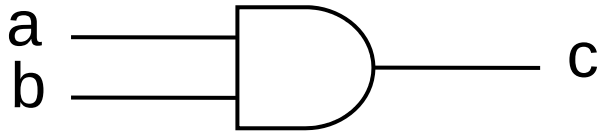
- Defect
  - A defect is the unintended difference between the implemented hardware and its intended design
  - Defects occur either during manufacture or during the use of devices
- Fault
  - A representation of a *defect* at the abstracted function level
- Error
  - A wrong output signal produced by a defective system
  - An error is caused by a *Fault* or a design error

Typical Types of Defects

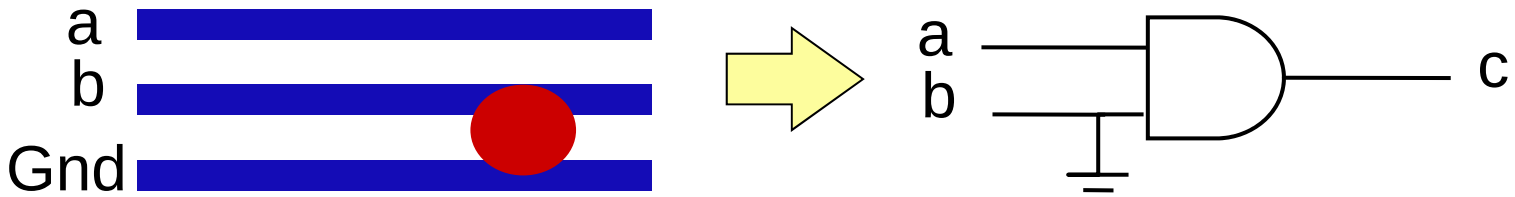
- Extra and missing material
  - Primarily caused by dust particles on the mask or wafer surface, or in the processing chemicals
- Oxide breakdown
  - Primarily caused by insufficient oxygen at the interface of silicon (Si) and silicon dioxide (SiO_2), chemical contamination, and crystal defects
- Electromigration
  - Primarily caused by the transport of metal atoms when a current flows through the wire
    - Because of a low melting point, aluminum has large self-diffusion properties, which increase its electromigration liability

Example

- Consider one two-input AND gate



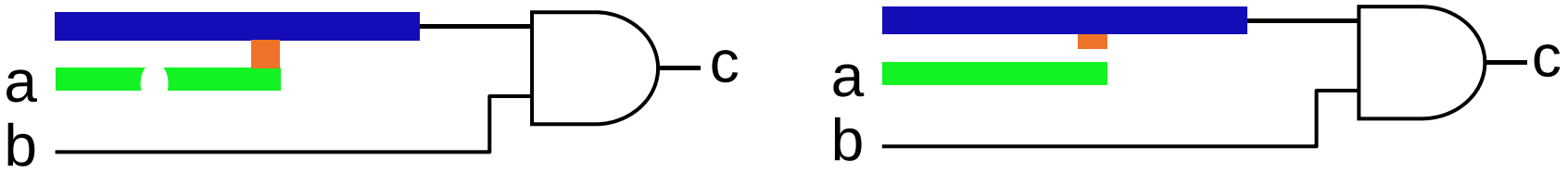
- Defect: a short to ground



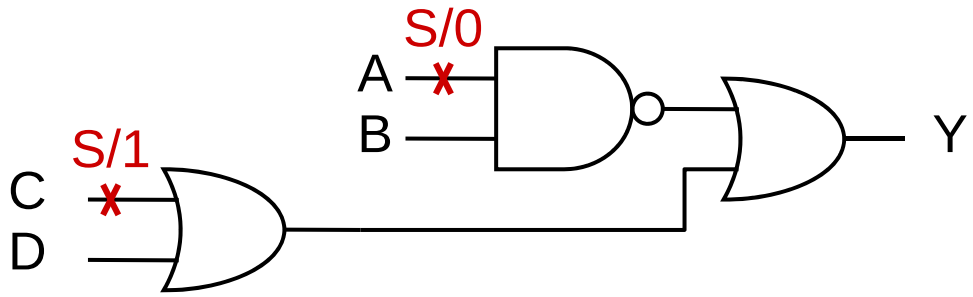
- Fault: signal b stuck at logic 0
- Error: a=1, b=1, c=0 (correct output c=1)
- Note that the error is not permanent. As long as at least one input is 0, there is no error in the output

Defect, Fault, and Error

- Different types of defects may cause the same fault



- Different types of faults may cause the same error
  - E.g., A stuck-at-0, Y=1; C stuck-at-1, Y=1



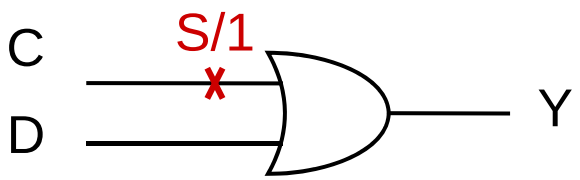
The Test Problem

Defect → Fault → Test pattern → Fault coverage

Fault modeling

Test pattern generation

Fault simulation

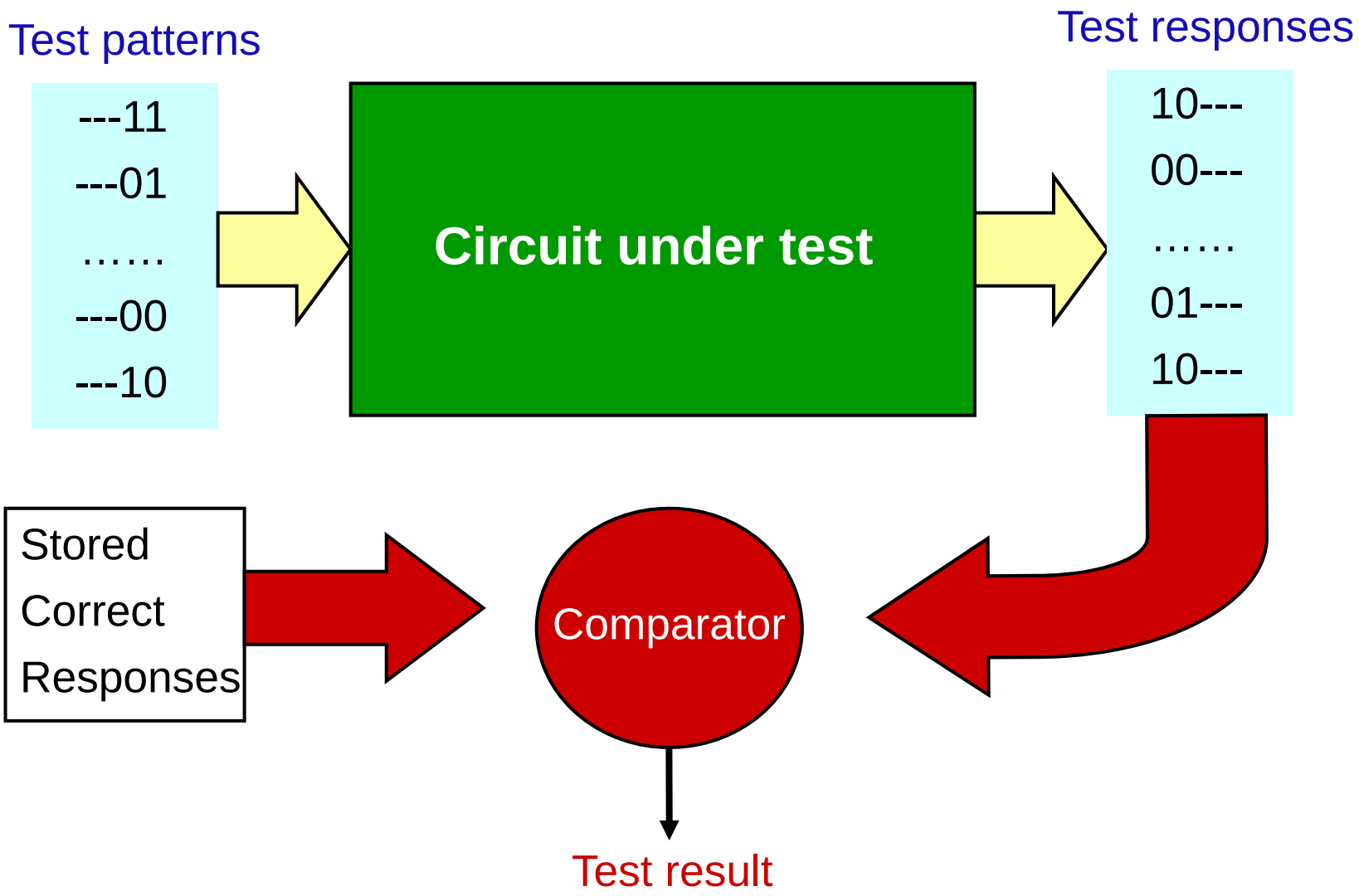


C	D	Y	Y(C is S/1)
0	0	0	**1**
0	1	1	1
1	0	1	1
1	1	1	1

Ideal Tests & Real Tests

- The problems of ideal tests
  - Ideal tests detect all defects produced in the manufacturing process
  - Ideal tests pass all functionally good devices
  - Very large numbers and varieties of possible defects need to be tested
  - Difficult to generate tests for some real defects
- Real tests
  - Based on analyzable fault models, which may not map on real defects
  - Incomplete coverage of modeled faults due to high complexity
  - Some good chips are rejected. The fraction (or percentage) of such chips is called the *yield loss*
  - Some bad chips pass tests. The fraction (or percentage) of bad chips among all passing chips is called the *defect level*

How to Test Chips?



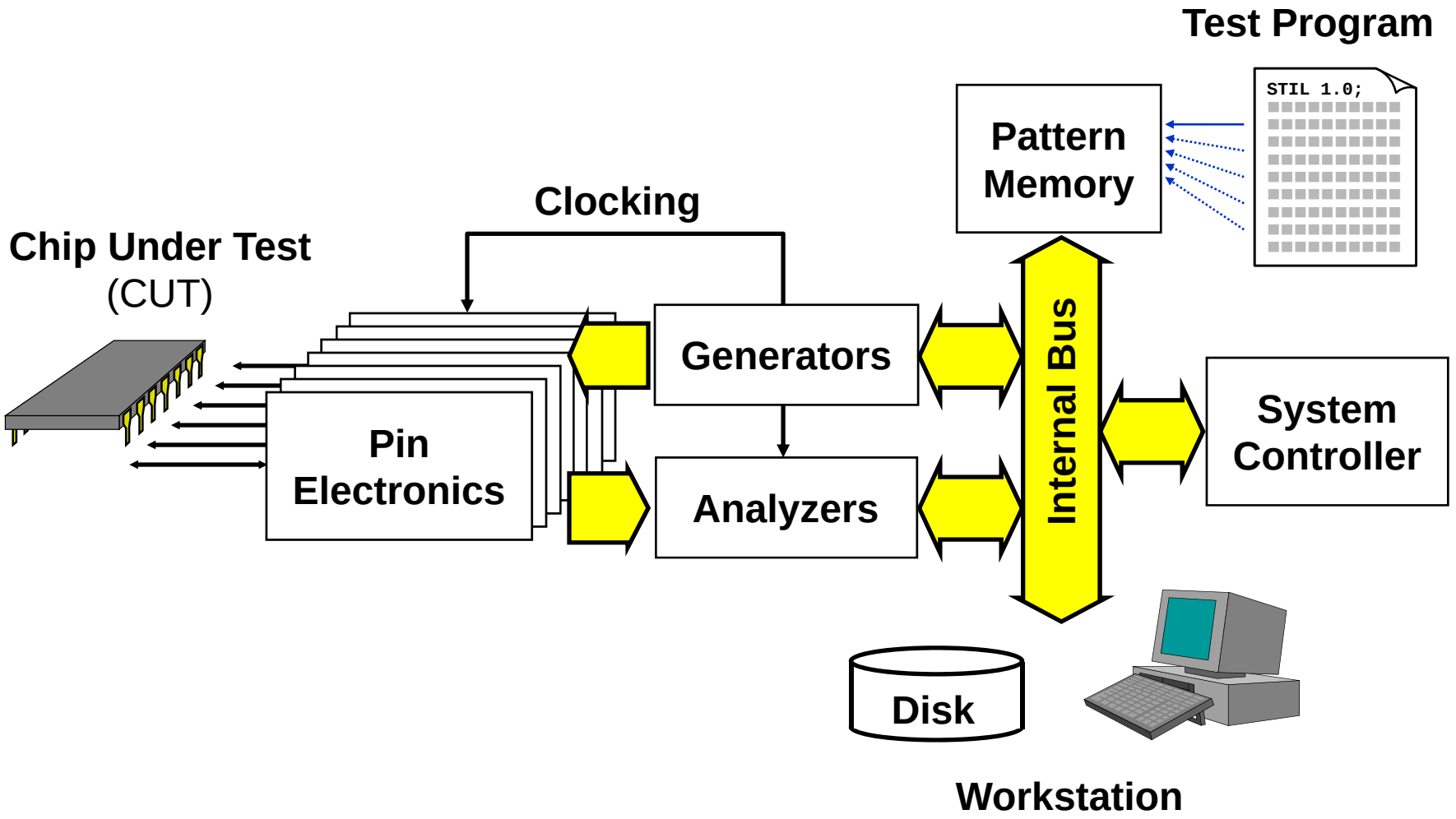
Cost of Test

- Design for testability (DFT)
  - Chip area overhead and yield reduction
  - Performance overhead
- Software processes of test
  - Test generation and fault simulation
  - Test programming and debugging
- Manufacturing test
  - *Automatic test equipment* (ATE) capital cost
  - Test center operational cost

ADVENTEST Model T6682 ATE

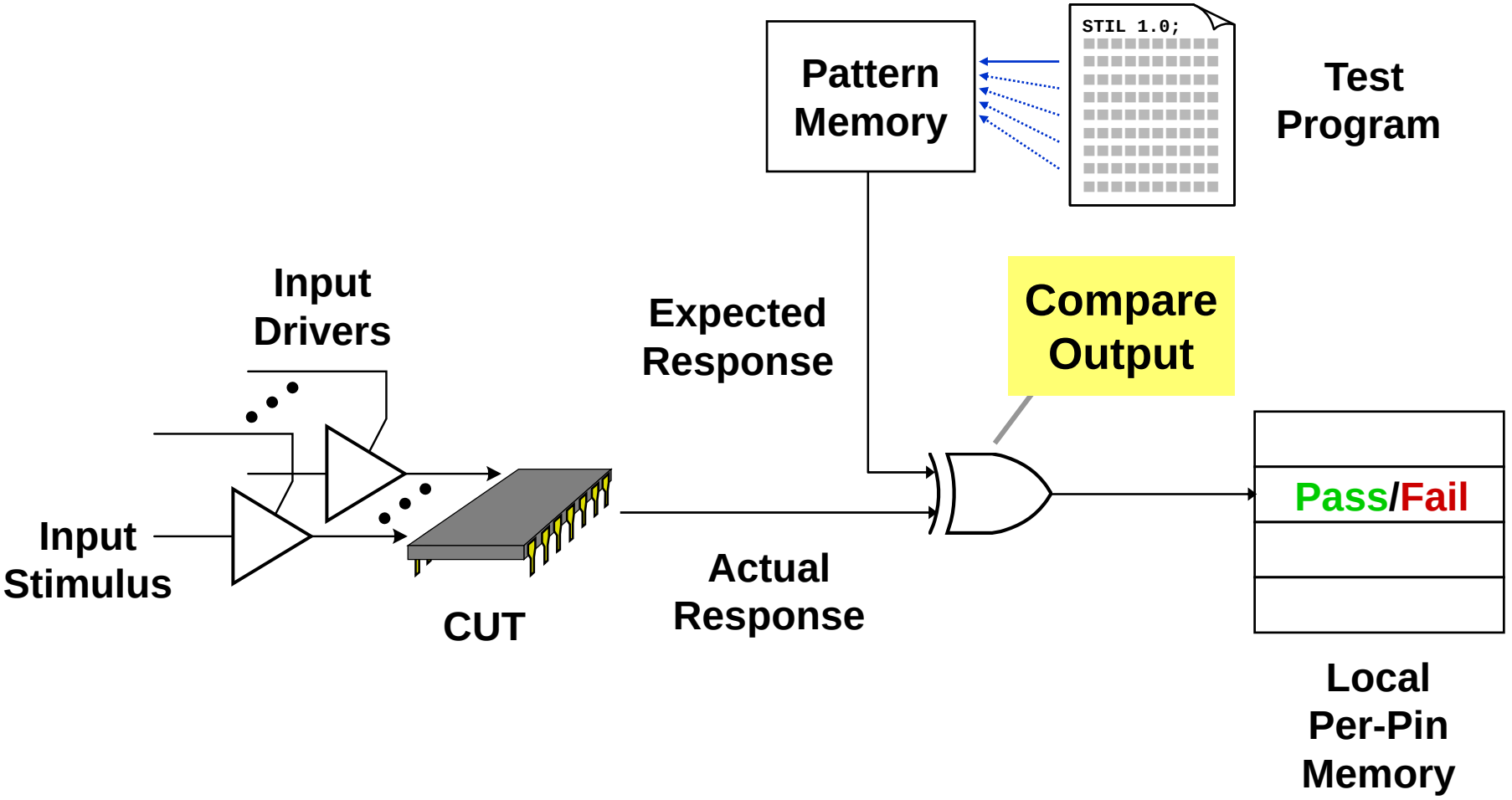
- Consists of
  - Powerful computer
  - Powerful 32-bit digital signal processor (DSP) for analog testing
  - Probe head: actually touches the bare dies or packaged chips to perform fault detection experiments
  - Probe card: contains electronics to measure chip pin or pad

Internal Structure of the ATE



Source: H.-J. Huang, CIC

ATE Test Operation



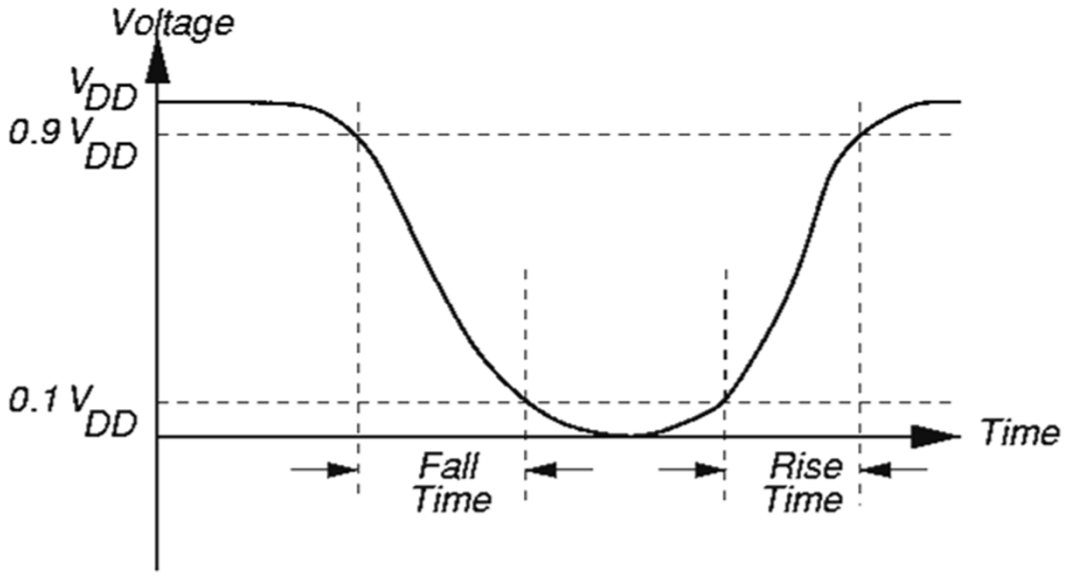
Source: H.-J. Huang, CIC

Types of Test

- *Characterization testing*
  - A.k.s. *design debug* or *verification testing*
  - Performed on a new design before it is sent to production
  - Verify whether the design is correct and the device will meet all specifications
  - Functional tests and comprehensive AC and DC measurements are made
  - A *characterization test* determines the exact limits of device operation values
- *DC Parameter tests*
  - Measure steady-state electrical characteristics
  - For example, threshold test
    - $0<V_{OL}<V_{IL}$
    - $V_{IH}<V_{OH}<V_{CC}$

Types of Test

- *AC parametric tests*
  - Measure transient electronic characteristics
  - For example:
    - Rise time & fall time tests



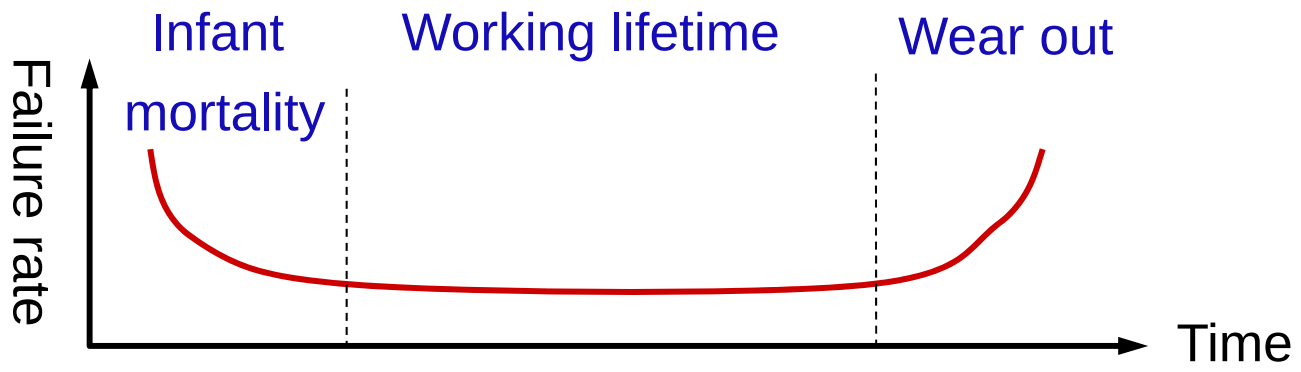
Types of Test

- *Production testing*
  - Every fabricated chip is subjected to production tests
  - The test patterns may not cover all possible functions and data patterns but must have a high fault coverage of modeled faults
  - The main driver is cost, since every device must be tested. Test time must be absolutely minimized
  - Only a go/no-go decision is made
  - Test whether some device-under-test parameters are met to the device specifications under normal operating conditions
- *Burn-In testing*
  - Ensure reliability of tested devices by testing
  - Detect the devices with potential failures

Types of Test

- The potential failures can be accelerated at elevated temperatures
- The devices with *infant mortality failures* may be screened out by a short-term burn-in test in an accelerate

- Failure rate versus product lifetime (*bathtub curve*)



Testing Economics

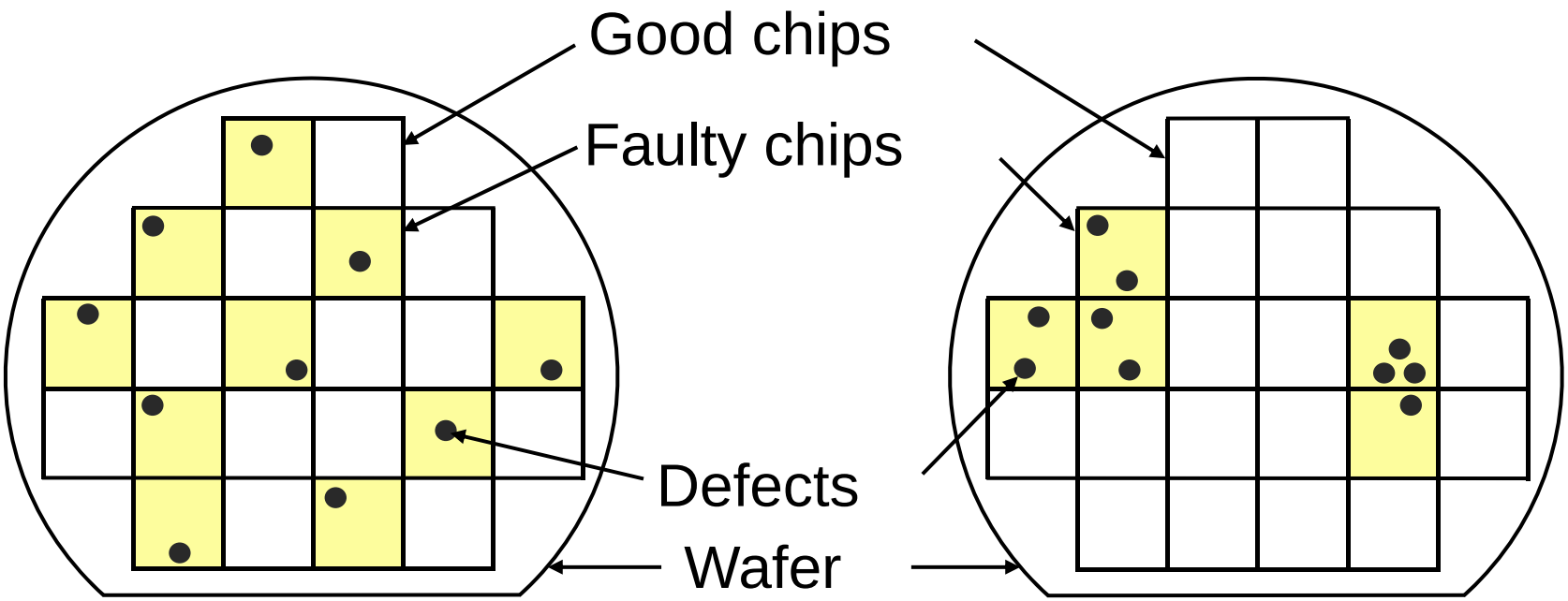
- Chips must be tested before they are assembled onto PCBs, which, in turn, must be tested before they are assembled into systems
- The rule of ten
  - If a chip fault is not detected by chip testing, then finding the fault costs 10 times as much at the PCB level as at the chip level
  - Similarly, if a board fault is not found by PCB testing, then finding the fault costs 10 times as much at the system level as at the board level
- Some claim that the rule of ten should be renamed the rule of twenty
  - Chips, boards, and systems are more complex

VLSI Chip Yield

- A manufacturing defect is a finite chip area with electrically malfunctioning circuitry caused by errors in the fabrication process
- A chip with no manufacturing defect is called a good chip
- Fraction (or percentage) of good chips produced in a manufacturing process is called the *yield*. Yield is denoted by symbol ***Y***
- Cost of a chip

$$\frac{\text{Cost of fabricating and testing a wafer}}{\text{Yield x Number of chip sites on the wafer}}$$

VLSI Chip Yield



Wafer yield = 12/22 = 0.55

Wafer yield = 17/22 = 0.77

Fault Coverage & Defect Level

- Fault coverage (FC)
  - The measure of the ability of a test (a collection of test patterns) to detect a given faults that may occur on the device under test
  - FC=#(detected faults)/#(possible faults)
- Defect level (DL)
  - *The ratio of faulty chips among the chips that pass tests*
  - DL is measured as *defects per million* (DPM)
  - DL is a measure of the effectiveness of tests
  - DL is a quantitative measure of the manufactured product quality. For commercial VLSI chips a DL greater than 500 DPM is considered unacceptable
- $DL = 1 - Y^{(1-FC)}$ and $0 < DL \leq 1 - Y$

Defect Level & Quality Level

- For example, required FC for DL=200 DPM

Y(%)	10	50	90	95	99
FC(%)	99.991	99.97	99.8	99.6	98

- Quality level (QL)
  - The fraction of good parts among the parts that pass all the tests and are shipped
  - $QL = 1 - DL = Y^{(1-FC)}$ and $0 \le QL \le 1$
- Consequently, fault coverage affects the quality level

Outline

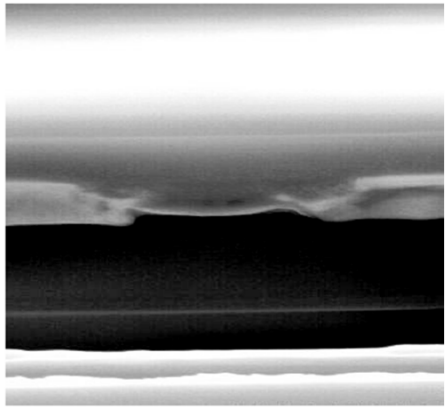
- Basics
- Fault Modeling
- Design-for-Testability

Test Process

- The testing problem
  - *Given a set of faults in the circuit under test (or device under test), how do we obtain a certain (small) number of test patterns which guarantees a certain (high) fault coverage?*
- Test process
  - What faults to test? (***fault modeling***)
  - How are test pattern obtained? (***test pattern generation***)
  - How is test quality (fault coverage) measured? (***fault simulation***)?
  - How are test vectors applied and results evaluated? (***ATE/BIST***)

Defect Categories

- Defect categories
  - Random defects, which are independent of designs and processes
  - Systematic defects, which depend on designs and processes used for manufacturing
- For example, random defects might be caused by random particles scattered on a wafer during manufacturing



A resistive open defect [Source: Cadence]

Logical Fault Models

- Systematic defects might be caused by process variations, signal integrity, and design integrity issues.
- It is possible both random and systematic defects could happen on a single die
- With the continuous shrinking of feature sizes, somewhere below the 180nm technology node, system defects have a larger impact on yield than random defects
- Logical faults
  - *Logical faults represent the physical defects on the behaviors of the systems*

Why Model Faults

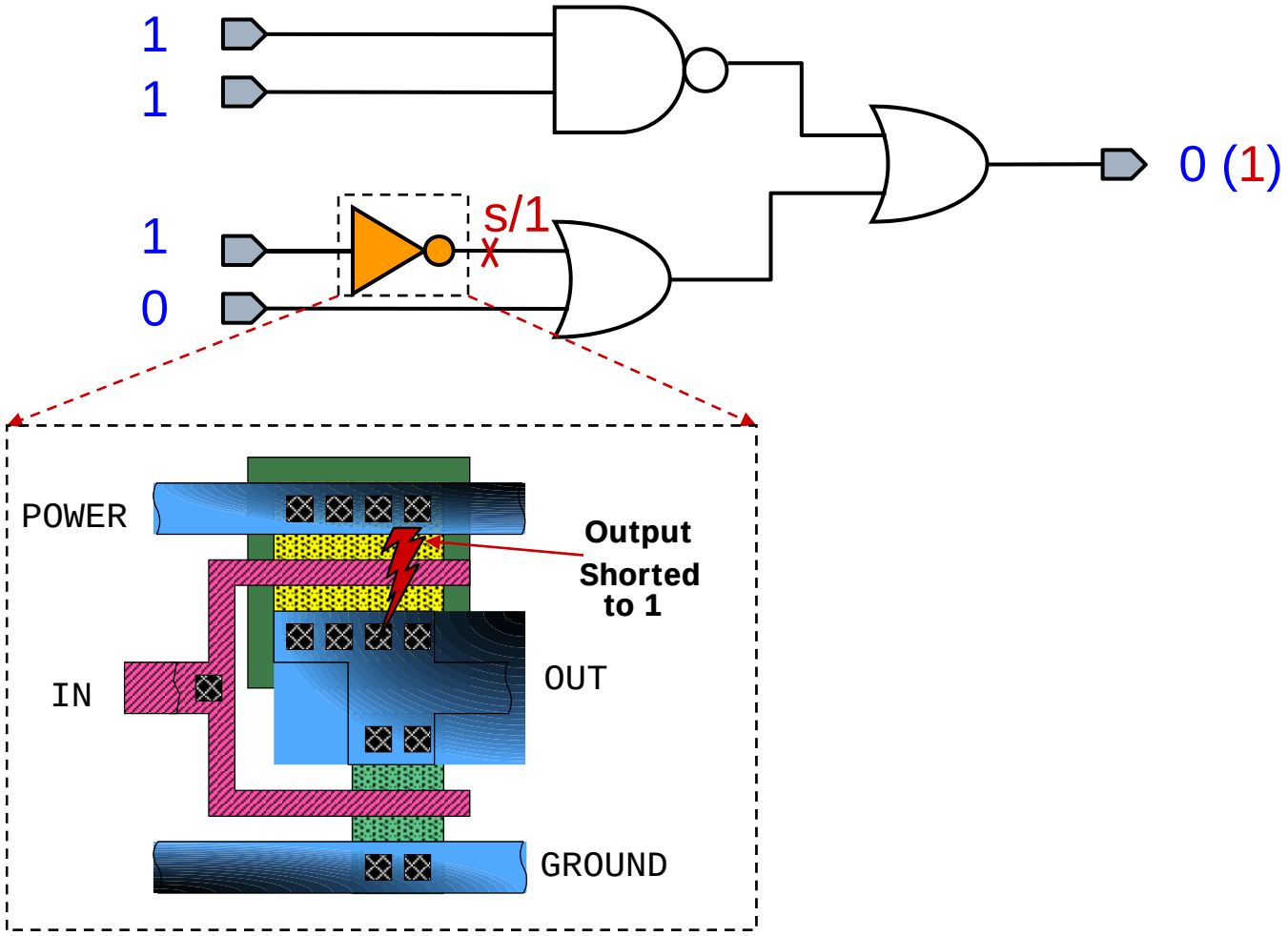
- I/O function tests inadequate for manufacturing (functionality versus component and interconnection testing)
- Real defects (often mechanical) too numerous and often not analyzable
- A fault model identifies targets for testing
- A fault model makes analysis possible
- Effectiveness measurable by experiments

Single Stuck-At Fault

- *Single (line) stuck-at fault*
  - The given line has a constant value (0/1) independent of other signal values in the circuit
- Properties
  - Only one line is faulty
  - The faulty line is permanently set to 0 or 1
  - The fault can be at an input or output of a gate
  - Simple logical model is independent of technology details
  - It reduces the complexity of fault-detection algorithms
- One stuck-at fault can model more than one kind of defect

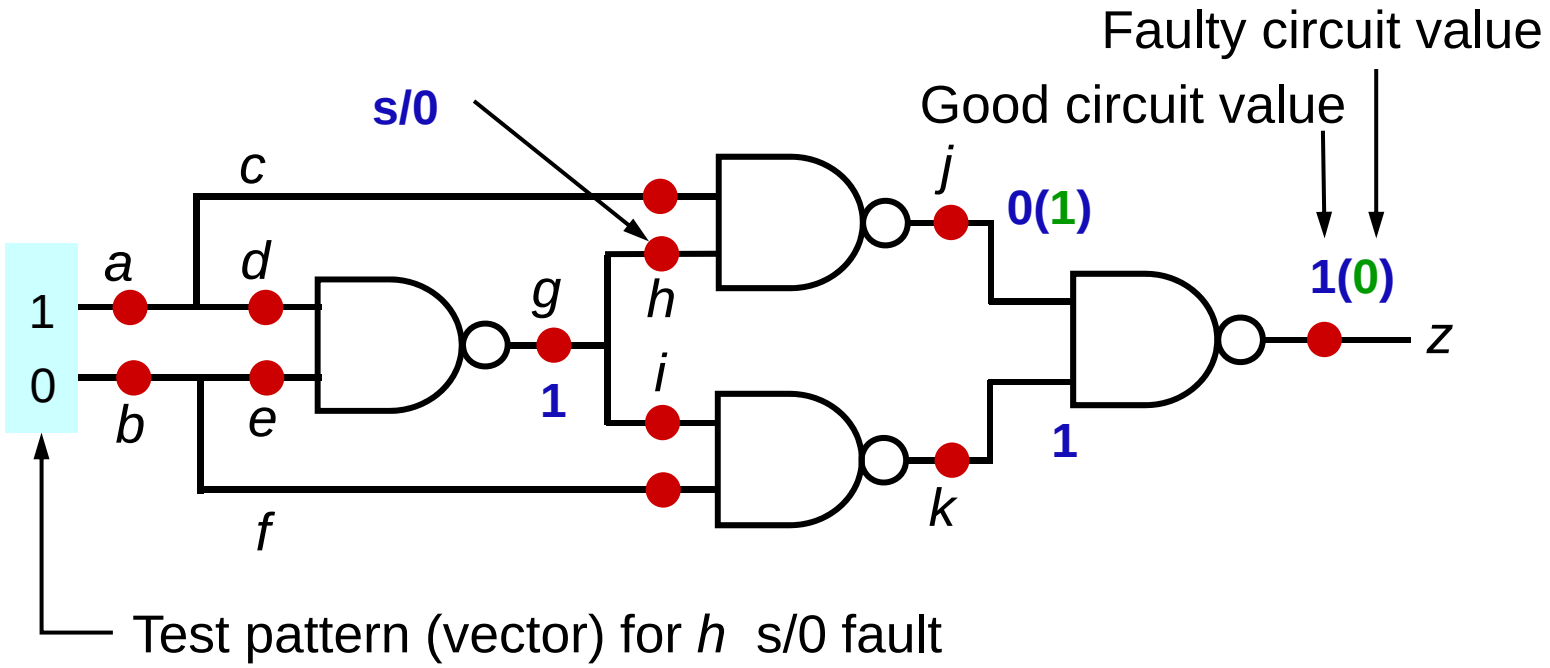
Single Stuck-At Fault Example

- A circuit with single stuck-at fault



Number of Single Stuck-At Faults

- Number of fault sites in a Boolean gate circuit
  - #PI + #gates + #(fanout branches)
- Example: XOR circuit has 12 fault sites (●) and 24 single stuck-at faults

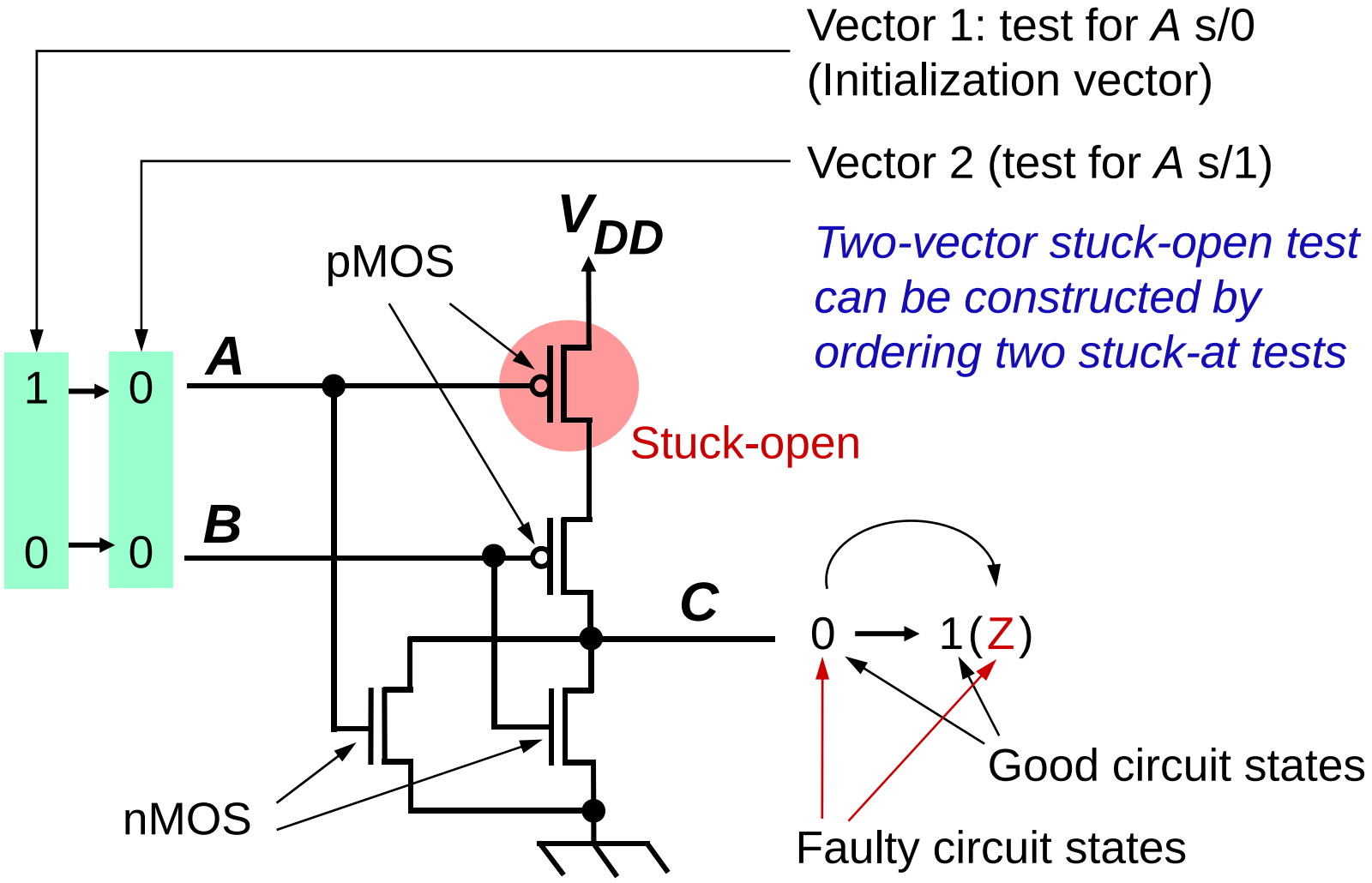


Transistor Faults

- MOS transistor is considered an ideal switch and two types of faults are modeled
  - ***Stuck-open*** -- a single transistor is permanently stuck in the open state
    - Turn the circuit into a sequential one
    - Need a sequence of at least 2 tests to detect a single fault
    - Unique to CMOS circuits
  - ***Stuck-on*** -- a single transistor is permanently shorted irrespective of its gate voltage
- Detection of a stuck-open fault requires two vectors
- Detection of a stuck-short fault requires the measurement of quiescent current (I_{DDQ})

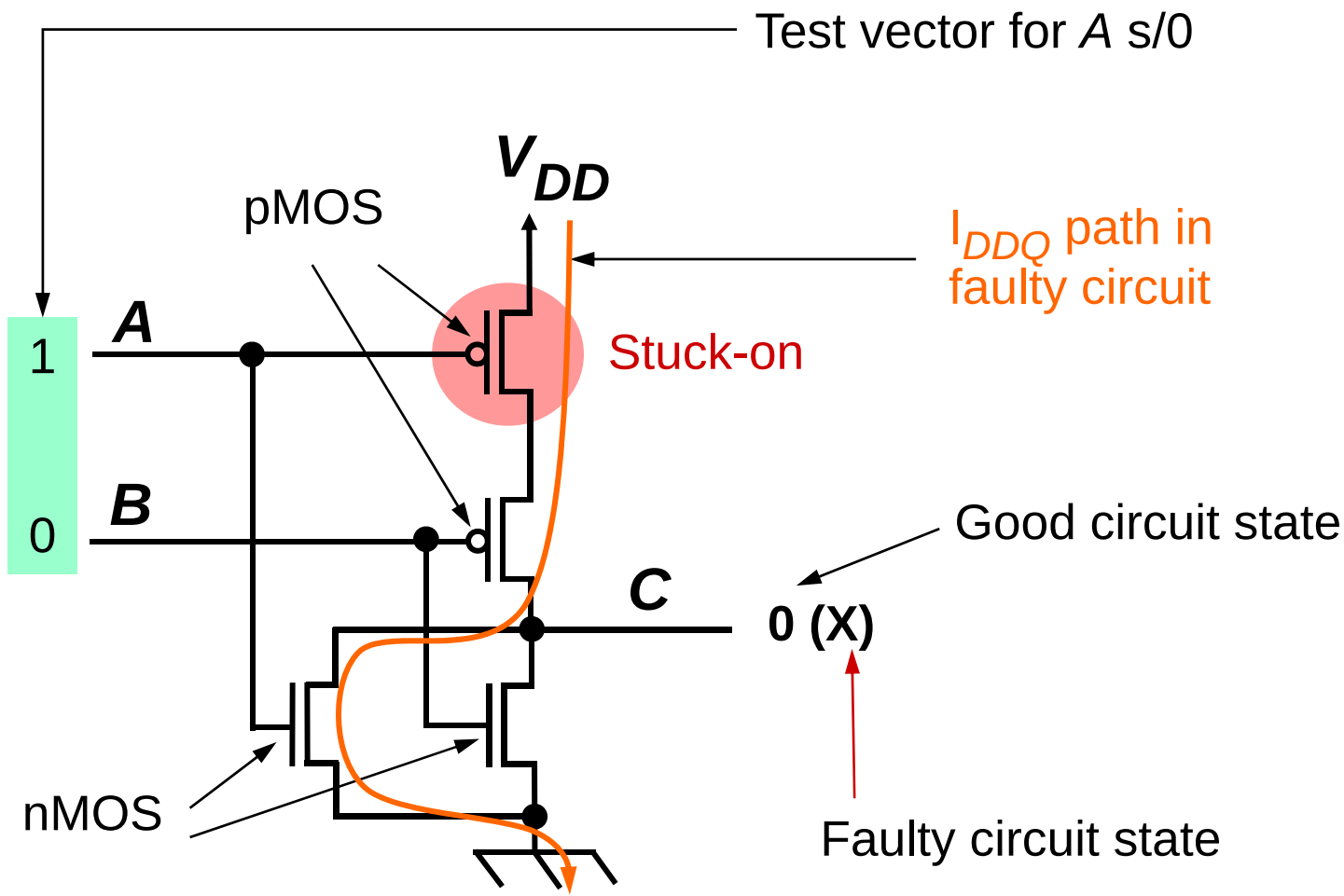
Transistor Stuck-Open Fault

- Example:



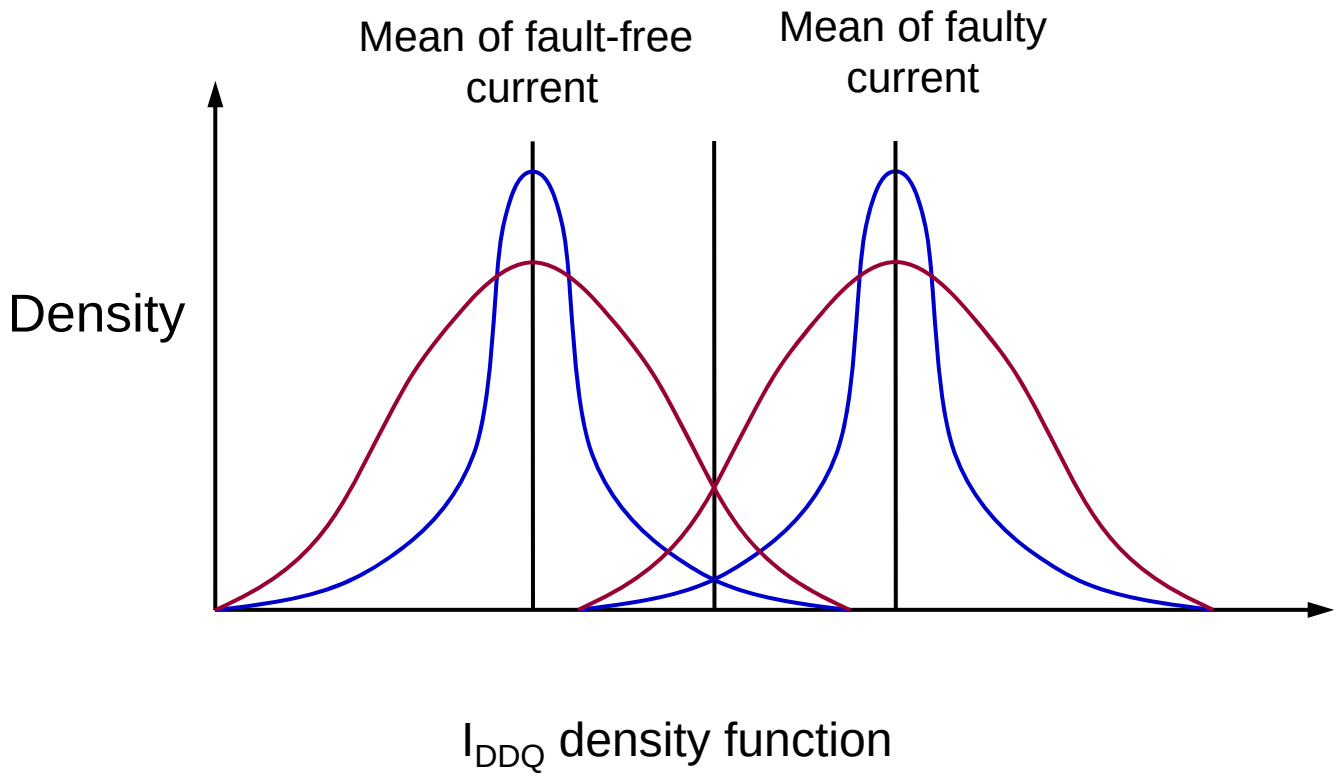
Test Stuck-On Fault Using I_{DDQ}

- Example:



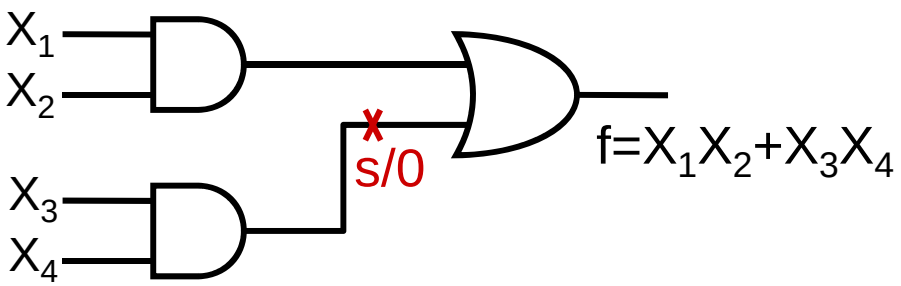
I_{DDQ} Test in Nano-scale Era

- Major problem: may results in unacceptable yield loss



Test & Test Set

- A ***test*** for a fault α in a circuit *C* is an input combination for which the output(s) of *C* is different when α is present than when it is not.
  - A.k.a. test pattern or test vector
  - X detect α then $f(X) \oplus f_\alpha(X) = 1$
- A ***test set*** for a class of faults *A* is a set of tests *T* such that $\forall \alpha \in A, \exists t \in T$ and t detects α
  - The test set for a fault α is $T_\alpha = f \oplus f_\alpha$
  - For example,



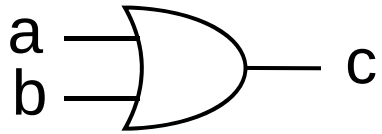
$$T_\alpha = f \oplus f_\alpha$$
$$= (X_1X_2 + X_3X_4) \oplus X_1X_2$$
$$= \overline{X_1}X_3X_4 + \overline{X_2}X_3X_4$$
$$= \{0011, 0111, 1011\}$$

Testing & Diagnosis

- ***Testing*** is a process which includes test pattern generation, test pattern application, and output evaluation.
- ***Fault detection*** tells whether a circuit is fault-free or not
- ***Fault location*** provides the location of the detected fault
- ***Fault diagnosis*** provides the location and the type of the detected fault
  - The input X distinguishes a fault α from another fault β iff $f_\alpha(X) \neq f_\beta(X)$, i.e., $f_\alpha(X) \oplus f_\beta(X) = 1$

Testing & Diagnosis

- Example:



a	b	c	$c_{a/0}$	$c_{a/1}$	$c_{b/0}$	$c_{b/1}$	$c_{c/0}$	$c_{c/1}$
0	0	0	0	**1**	0	**1**	0	**1**
0	1	1	1	1	**0**	1	**0**	1
1	0	1	**0**	1	1	1	**0**	1
1	1	1	1	1	1	1	**0**	1

- $C_{a/0}$ and $C_{c/0}$ are detected by the test pattern (1,0)
- If we apply two test patterns: (1,0) & (0, 1)
  - Two corresponding outputs are faulty→$C_{c/0}$
  - Only the output with respect to the input (1,0) is faulty→$C_{a/0}$

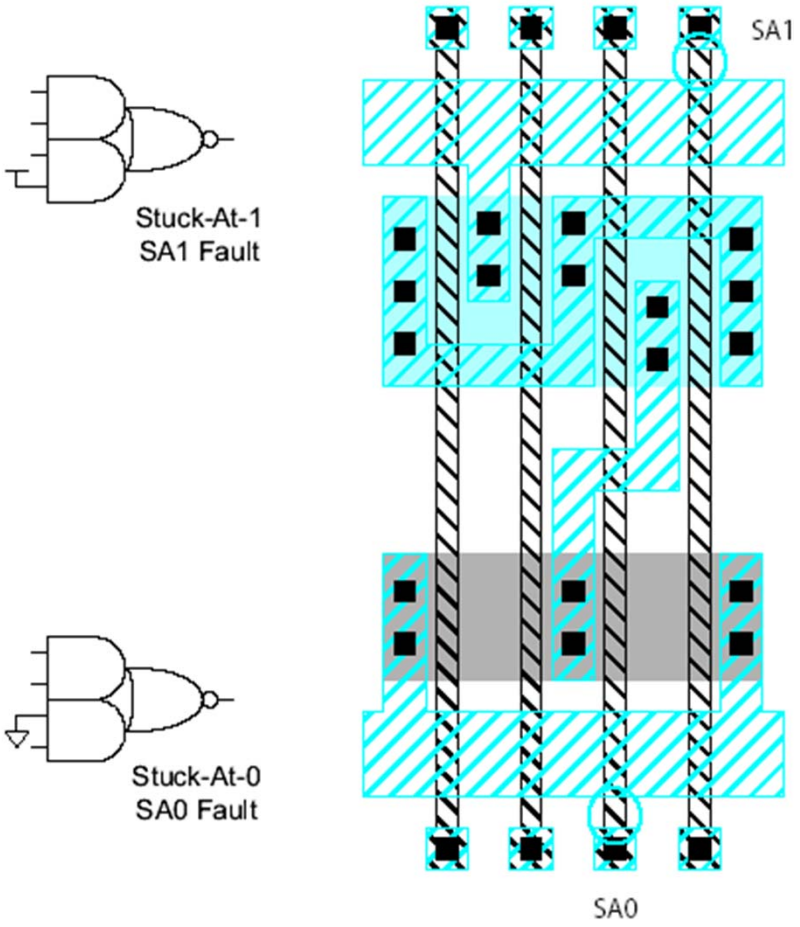
Outline

- Basics
- Fault Modeling
- Design-for-Testability (Source: Prof. David Harris)

Stuck-At Faults

- How does a chip fail?
  - Usually failures are shorts between two conductors or opens in a conductor
  - This can cause very complicated behavior
- A simpler model: *Stuck-At*
  - Assume all failures cause nodes to be "stuck-at" 0 or 1, i.e. shorted to GND or V_{DD}
  - Not quite true, but works well in practice

Examples



Observability & Controllability

- *Observability*: ease of observing a node by watching external output pins of the chip
- *Controllability*: ease of forcing a node to 0 or 1 by driving input pins of the chip

- Combinational logic is usually easy to observe and control
- Finite state machines can be very difficult, requiring many cycles to enter desired state
  - Especially if state transition diagram is not known to the test engineer

Test Pattern Generation

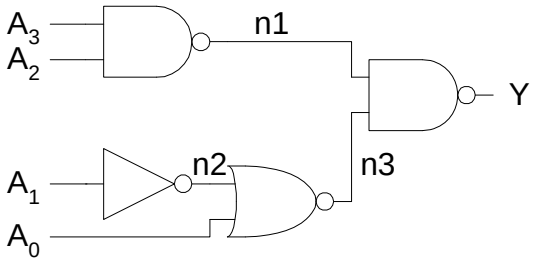
- Manufacturing test ideally would check every node in the circuit to prove it is not stuck.
- Apply the smallest sequence of test vectors necessary to prove each node is not stuck.

- Good observability and controllability reduces number of test vectors required for manufacturing test.
  - Reduces the cost of testing
  - Motivates design-for-test

Test Example

SA1 SA0

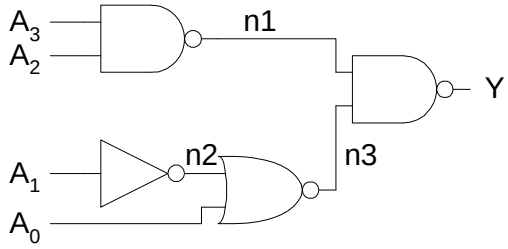
- A_3
- A_2
- A_1
- A_0
- n1
- n2
- n3
- Y



- Minimum set:

Test Example

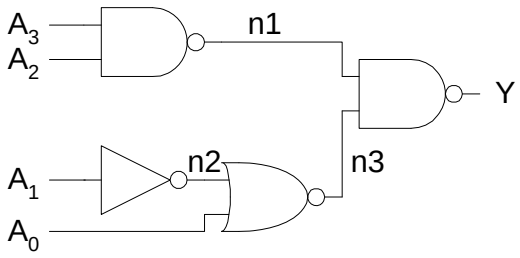
	SA1	SA0
A_3	{0110}	{1110}
A_2		
A_1		
A_0		
n1		
n2		
n3		
Y		



- Minimum set:

Test Example

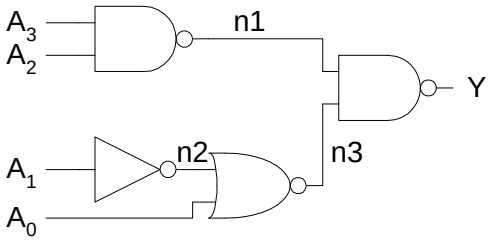
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1		
A_0		
n1		
n2		
n3		
Y		



- Minimum set:

Test Example

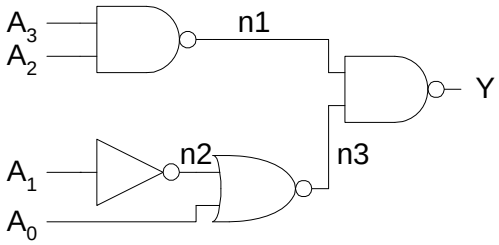
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0		
n1		
n2		
n3		
Y		



- Minimum set:

Test Example

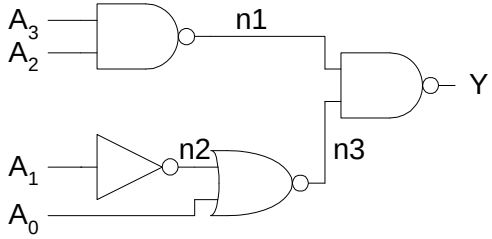
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0	{0110}	{0111}
n1		
n2		
n3		
Y		



- Minimum set:

Test Example

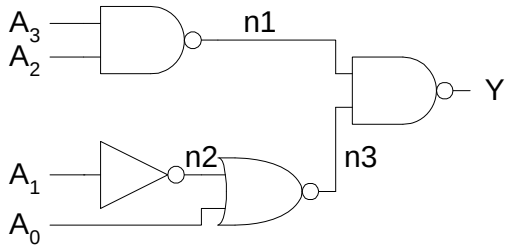
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0	{0110}	{0111}
n1	{1110}	{0110}
n2		
n3		
Y		



- Minimum set:

Test Example

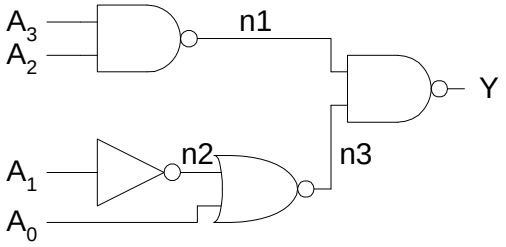
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0	{0110}	{0111}
n1	{1110}	{0110}
n2	{0110}	{0100}
n3		
Y		



- Minimum set:

Test Example

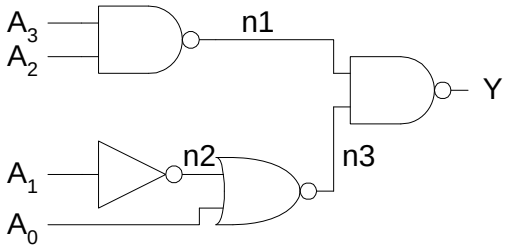
	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0	{0110}	{0111}
n1	{1110}	{0110}
n2	{0110}	{0100}
n3	{0101}	{0110}
Y		



- Minimum set:

Test Example

	SA1	SA0
A_3	{0110}	{1110}
A_2	{1010}	{1110}
A_1	{0100}	{0110}
A_0	{0110}	{0111}
n1	{1110}	{0110}
n2	{0110}	{0100}
n3	{0101}	{0110}
Y	{0110}	{1110}



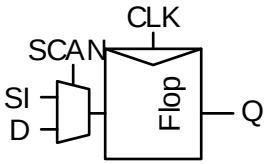
- Minimum set: {0100, 0101, 0110, 0111, 1010, 1110}

Design for Test

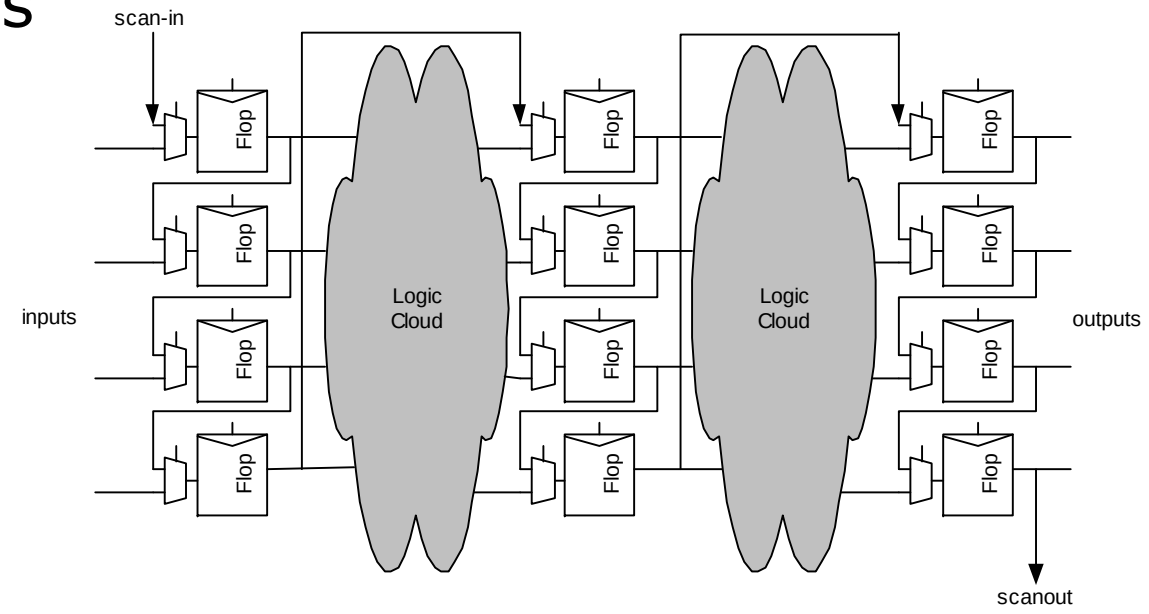
- Design the chip to increase observability and controllability
- If each register could be observed and controlled, test problem reduces to testing combinational logic between registers.
- Better yet, logic blocks could enter test mode where they generate test patterns and report the results automatically.

Scan

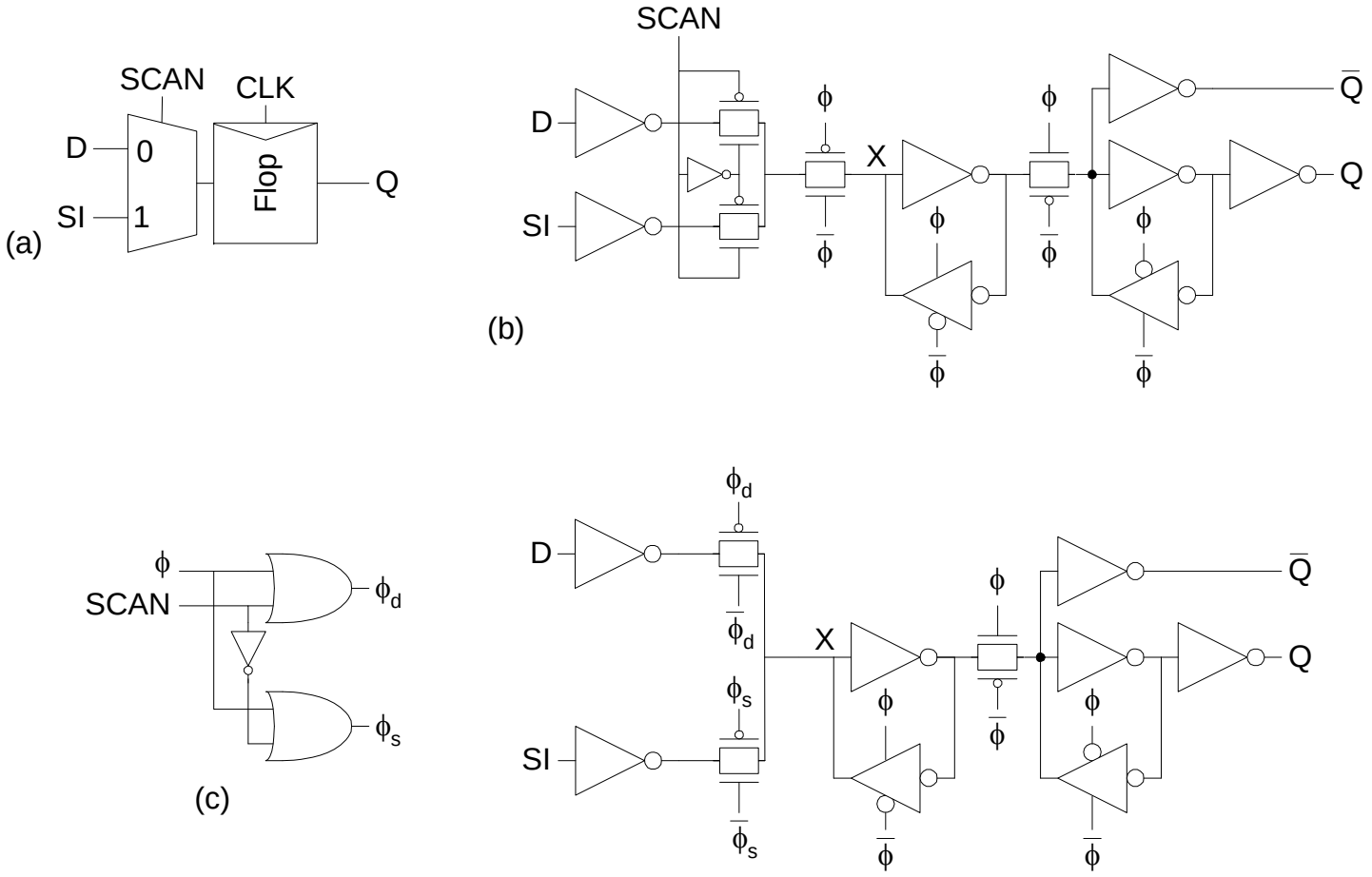
- Convert each flip-flop to a scan register
  - Only costs one extra multiplexer
- Normal mode: flip-flops behave as usual
- Scan mode: flip-flops behave as shift register



- Contents of flops can be scanned out and new values scanned in



Scannable Flip-flops

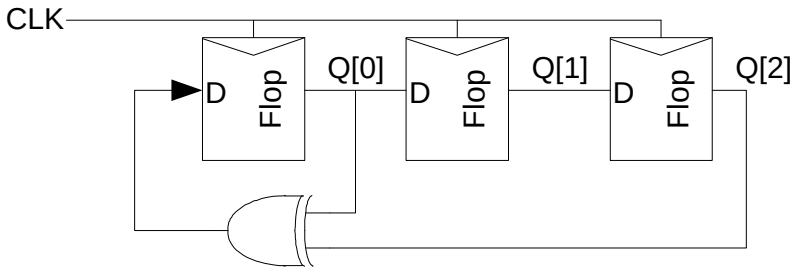


Built-in Self-test

- Built-in self-test lets blocks test themselves
  - Generate pseudo-random inputs to comb. logic
  - Combine outputs into a *syndrome*
  - With high probability, block is fault-free if it produces the expected syndrome

PRSG

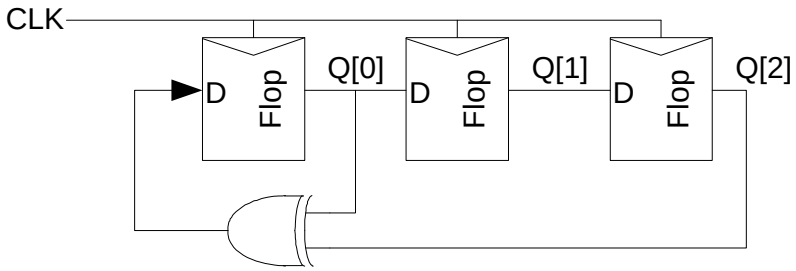
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	
2	
3	
4	
5	
6	
7	

PRSG

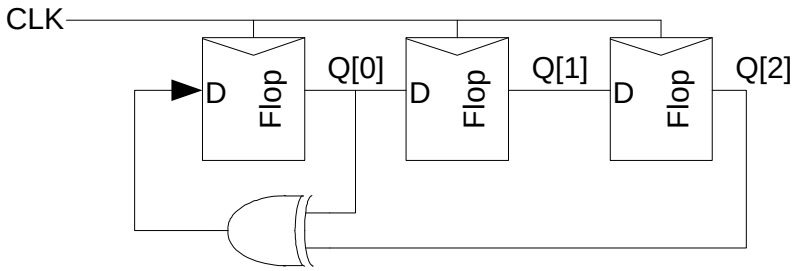
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	
3	
4	
5	
6	
7	

PRSG

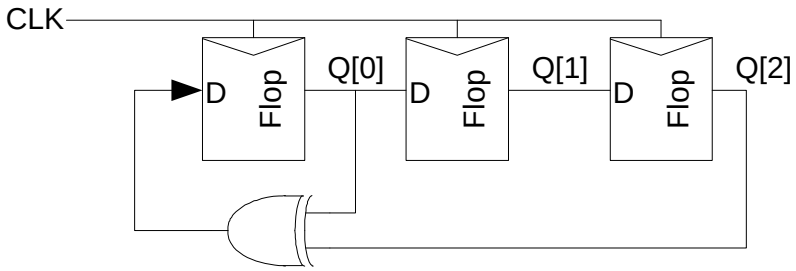
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	
4	
5	
6	
7	

PRSG

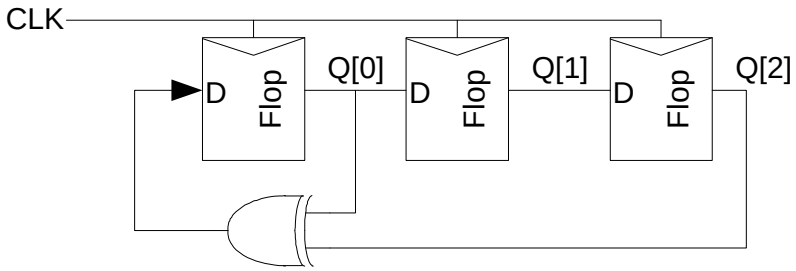
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	
5	
6	
7	

PRSG

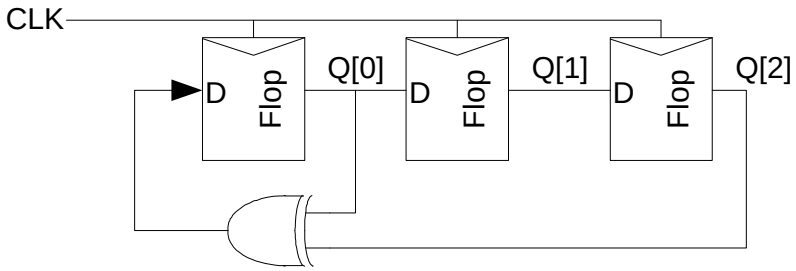
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	
6	
7	

PRSG

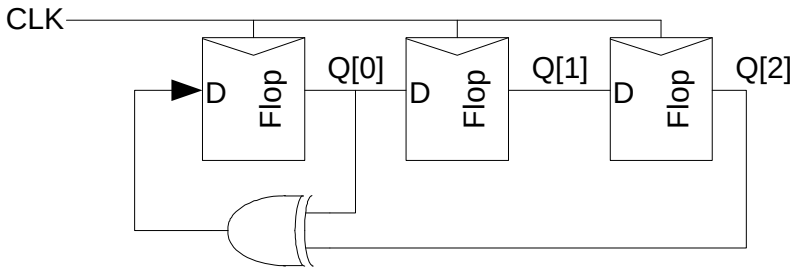
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	
7	

PRSG

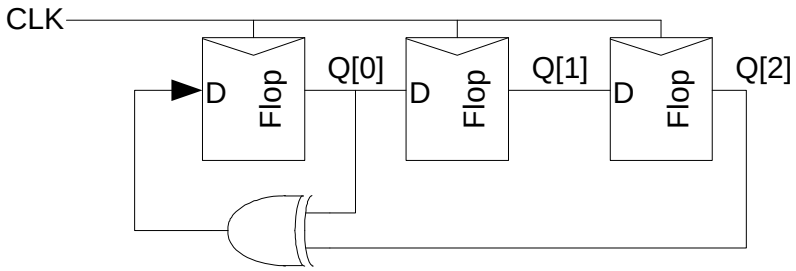
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	011
7	

PRSG

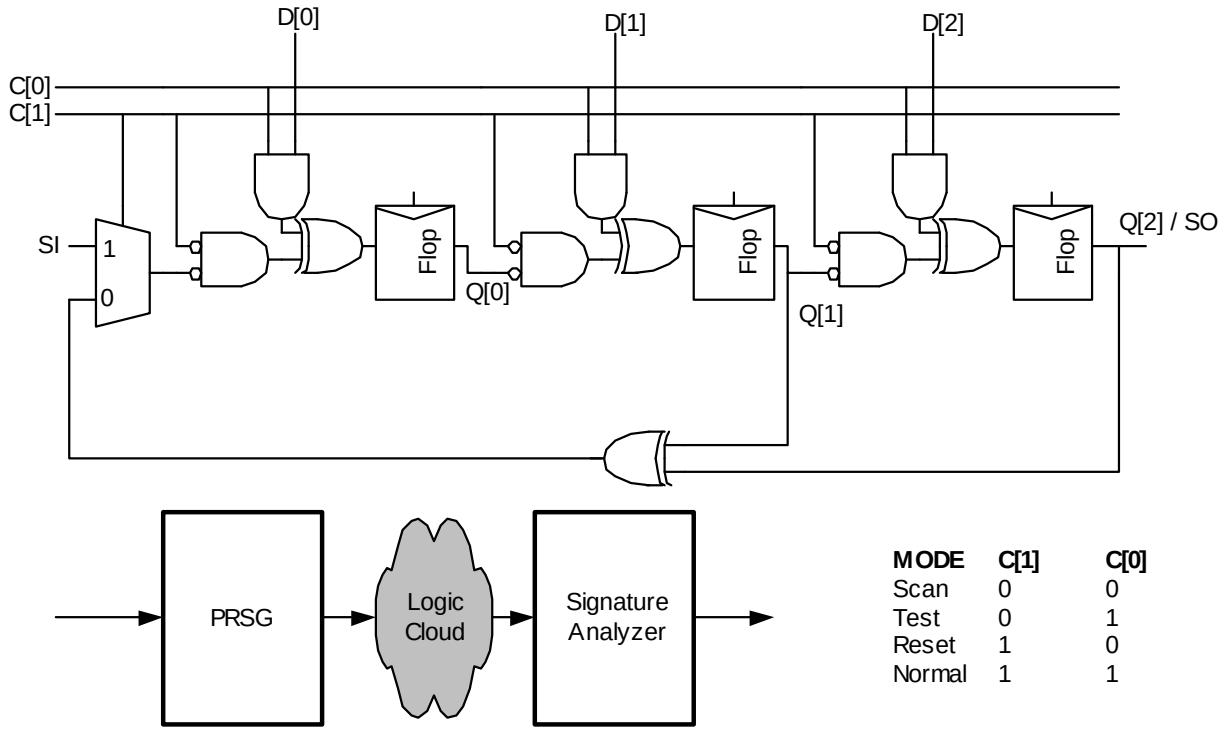
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	011
7	111 (repeats)

BILBO

- Built-in Logic Block Observer
  - Combine scan with PRSG & signature analysis

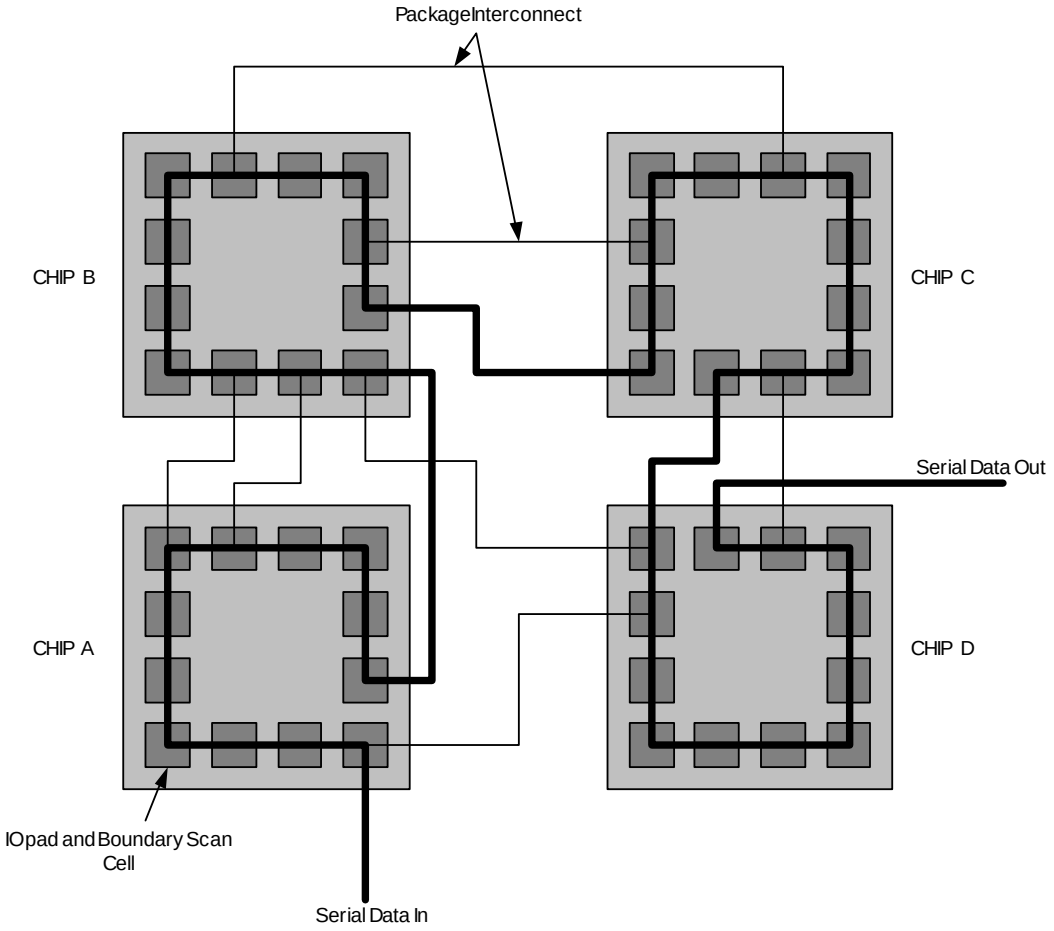


MODE	C[1]	C[0]
Scan	0	0
Test	0	1
Reset	1	0
Normal	1	1

Boundary Scan

- Testing boards is also difficult
  - Need to verify solder joints are good
    - Drive a pin to 0, then to 1
    - Check that all connected pins get the values
- Through-hold boards used “bed of nails”
- SMT and BGA boards cannot easily contact pins
- Build capability of observing and controlling pins into each chip to make board test easier

Boundary Scan Example



Boundary Scan Interface

- Boundary scan is accessed through five pins
  - TCK: test clock
  - TMS: test mode select
  - TDI: test data in
  - TDO: test data out
  - TRST*: test reset (optional)

- Chips with internal scan chains can access the chains through boundary scan for unified test strategy.

Summary

- Think about testing from the beginning
  - Simulate as you go
  - Plan for test after fabrication

- "If you don't test it, it won't work! (Guaranteed)"