VLSI III – Summary

written by Stephan Senn, D-ITET

25 Aug 2006

Contents

* belongs to VLSI II

Testable Design*

Rules of Tens

Status	Costs in $
chip on the wafer	0.1$
IC in package	1$
IC on a PCB	10$
IC in apparatus	100$
apparatus in the field	1000$

Trends

- rising Complexity of Chips (SoC): increased observability and controllability problem
- higher Pin Counts (over 6000)
- higher Gate Counts (over 10^9)
- longer Test Development Time (over 12m)
- higher Verification Costs (over 60%)
- longer Test Duration (over 100s)
- strong Collaboration of Test Engineers and IC Designers
- further Trends: see p. 46f.!

Defect Level

- Defect Level: $DL = \frac{\text{number of delivered defective chips}}{\text{number of delivered chips}} = 1 - Y^{1-T}$ (typical < 1%)
- Fault Coverage: $T = \frac{\text{amount of detectable faults}}{\text{amount of all possible faults}}$ (typical: 95.5-99.2%)
- Yield: $Y = \frac{\text{amount of error free chips}}{\text{amount of produced chips}}$ (typical: 20-80%)

Test Setup

- Primary Inputs and Outputs
- Ideally Testable Design: can be tested by stimulating the primary inputs and observing the primary outputs.

Types of Tests

Test Type	Target	Tested Items	Duration
design verification (prototype test)	wafers, prototype chips	10	months
production test	wafers, chips	> 500'000	< 1s
reliability test (burn-in)	chips	10'000	weeks
incoming inspection	chips	1'000	seconds
self test	built-in	after every reset	seconds
self check	built-in	during operation	continuous

chips: die-bonded, wire-bonded (or flip-chipped) and packaged

Test Development Methods

- ***Method 1:*** Separate Design and Test Development
- ***Method 2:*** Design for Testability (DFT): only Architecture, Test Generation afterwards
- ***Method 3:*** Design for Testability (DFT): Interleaved Architecture and Test Generation
- ***Rule:*** The ideally testable design provides access to inner nodes for easier testing by means of test interfaces or built-in self test circuits.

Observability and Controllability

- ***Controllability:*** A node is controllable if an input sequence can change and therefore control its value.
- ***Observability:*** A node is observable if changing its state (invoked by a certain input sequence) leads to a different output sequence.
- ***Rule:*** A node is testable for a given fault if it is both controllable and observable.
- ***Implication:*** In the ideally testable chip all nodes are controllable and observable...
  - ...either naturally,
  - ...by clever circuit design,
  - ...by additional test circuitry
  - ...or a combination of all.

General Testability Rules

- Reset: well-defined “homing sequence” for reproducible results at the primary inputs and outputs
- Clock Bypass: for bypassing the internal oscillator by a separate input pin
- Clock Divider Bypass: for bypassing the internal clock dividing unit and feeding the chip by an already divided clock signal
- Global Feedback Break: for breaking up all zero-latency loops
- Avoid Hazards and Races: for asynchronous interfaces and low-power designs
- Pulse Generation: no use of asynchronous delay elements that are technology dependent (ratioed

circuits)

- Partition of Large Counter: for saving states, $m2^{n/m}$ instead of 2^n
- Avoid Redundancy: do not use the classical redundant hazard reduction technique
- Insertion of Test Pads: for probing critical regions of prototypes
- Stay at Synchronous Design: avoid any asynchronous circuitry wherever possible
- Good Documentation

Testing Architectures

Block Isolation without Busses

Advantages:	**Disadvantages:**
• modular test development • fast tests • timing characterization independent for each block	• routing and area overhead • larger delays due to long signal lines and multiplexers • additional pins necessary • for complex circuits: test controller • sequential test generation for the individual blocks

Block Isolation with Busses

Advantages:	**Disadvantages:**
• less routing and area overhead • less additional signal delay • less additional pins necessary • modular test development • fast tests • timing characterization independent for each block	• non-bus signals and bus signals • special data flow controllers • bridge to busses if busses are shared due to pin reduction • in general: difficult

Scan Paths (see p. 20)

Advantages:	Disadvantages:
• no need of sequential test generation • low area overhead due to only one multiplexer per flipflop or special scan flipflops • low routing overhead • low pad overhead	• loss of design hierarchy • test generation for large combinational block • large test vectors • more severe clock skew problems • reset synchronization flipflop not included • activation of 'forbidden' states possible • no scans for latches • missing scan options for standard cells (sequential or combinational)

Combination of Scan Path and Block Isolation

Built-In Self Test (BIST) and Boundary Scan

On- and Off-Line

- *On-Line BIST:* chip verification while the IC is running
- *Off-Line BIST:* putting chip into test mode which verifies certain functions

Advantages and Disadvantages

Advantages:	**Disadvantages:**
• Large number of test vectors can be run inside the chip at maximal clock frequency. • Modular design: Test hierarchy is equal to the functional hierarchy! • minimum pin requirement • complex systems: diagnostic routines and power-up system tests	• fixed test set and fault coverage after tape out, workarounds possible (e.g. loadable test vectors, etc.) • fault grading already at design process (strong DFT, not really a disadvantage!) • uses additional silicon area • less yield

Test Stimulus Generators

- ROM
- Counter
- Linear Feedback Shift Register (LFSR)

Test Response Analysis

- ROM
- Compression Methods:
  - Ones Count Compression: $P(M|r)=\frac{\binom{N}{r}-1}{2^N-1}$ with $0 \leqslant r \leqslant N$
  - Transition Count Compression: $P(M|r)=\frac{\binom{N-1}{r}-1}{2^N-1}$ with $0 \leqslant r \leqslant N$
  - Parity Check Compression: $P(M) \approx 0.5$
  - Cyclic Redundancy Check (CRC) Signature Compression: $P(M)=\frac{1}{2^n-1}$
  - Syndrome Testing: $S=\frac{k}{2^m}$ with $0 \leqslant S \leqslant 1$

N : amount of random 1-bit vectors

r : number of ones in N

n : number of flipflops

M : amounts of bits that are masks (random variable)

m : amount of inputs

k : amount of ones in output for all input combinations

- Characteristics: see p. 165
  - Simple realization and low area consumption?
  - Speed?
  - Compression performance?
  - Information loss?
  - Special test sets required?
  - Modifications of CUT required?

Random Test and Random BIST

- No way!
- Fault Coverage Saturation typically at 70%!

Built-In Logic Block Observer (BILBO)

- claimed to be a BIST methodology
- see pp. 166 and 167!

Macrocells and Virtual Components

- Examples: RAM, Multiplier, USB interface, Microprocessors, etc.
- Rule: *The more complex a circuit block, the more important to get DFT with it!*

Boundary Scan

- standardized interface which allows test access to electronic systems, Printed Circuit Board (PCB) and VLSI chips
- Working Principle: see p. 173!
  - Boundary Scan Cells with Boundary Scan Path
  - Test Access Port (TAP) with well-defined controller and interfaces
- System Tests
- IEEE Standard 1149.1-1990, Joint Test Action Group (JTAG)
- Boundary-Scan Description Language (BSDL)

Testing Silicon*

DUT Board

Sometimes also called *load board* or *DUT fixture*.

- ***DUT powering:***
  - use standard DUT board for saving costs
  - standard DUT boards: power pins are fixed, signal pins can be freely chosen
- ***Signal Drive in a 50Ω Environment***
- ***Bidirectional Pins:***
  - avoid them if possible
  - tristate control capability
  - do not forget to include direction control bits into the test vectors
- ***Auxiliary Circuits during Testing:*** Add test facility into the chip or provide suitable test vectors that simulate the environment by using a well-established testbench.

Test Vectors

- Limited High-Speed Memory: Be aware that tester memory is limited! Look for break vectors and loading facilities! Also look for the size of fast SRAMs and the size of slower DRAMs.
- Limited Number of ATE Tester Channels: Be aware of that an ATE can only test chips with a certain maximum number of pins!

Timing Scheme

Stick to the following timing scheme: ↑Δ↓T↑ .

Fault Model*

The Single-Stuck-At-Fault Model (SSAF Model)

- Only one fault is present in the whole circuit at a time.
- ***Short Faults:*** shorts between ground and power
  - node- or output-stuck-at-0 (s-a-0): shorts to ground
  - node- or output-stuck-at-1 (s-a-1): shorts to power
- ***Open Faults:*** open connections
  - input-stuck-at-0 (s-a-0): input-open-0 fault
  - input-stuck-at-1 (s-a-1): input-open-1 fault
- ***Functional Faults:*** not considered for the SSAF model
- ***Delay Faults:*** not considered for the SSAF model
- ***Equivalent Faults:*** Two faults A and B are said to be (functionally) equivalent iff $f_A(x) = f_B(x)$.
- ***Redundant Faults:*** A fault A is said to be redundant iff $f_A(x) = f(x)$.
- ***Problem of Redundant Faults:***
  - do not change logical behaviour
  - but they can mask other detectable faults
- ***Fault Dictionary:*** includes equivalent stuck-at-faults, also called fault data base
- ***Problems:***
  - *intended redundancy:* for hazard suppression
  - *unintended redundancy:* interconnecting two or more irredundant modules for fault-tolerant designs
  - *CMOS circuits:* inherent redundancy due to the n- and p-network, low performance for the SSAF model
- ***The Proof for Redundancy:*** NP[1]-complete problems
- ***Fault Coverage:*** $T = \dfrac{\text{number of detected faults}}{\text{number of existing faults according to fault model M}}$
- ***Software Tools:***
  - *Fault Simulator:*

    Working Principle:
    - step 1: derivation of fault data base from the netlist (mostly at gate-level)
    - step 2: construction of fault netlist by inserting one fault from the fault data base into the fault-free netlist
    - step 3: simulation of the fault-free and the fault netlist
    - step 4: comparison of the outputs
    - step 5: Is the output vector of the simulated fault netlist equal to the output vector of the

1 **N**on-deterministic in **P**olynomial time

simulated fault-free netlist?

➔ Yes: -

➔ No: Insert the fault information into the fault data base.

- Have all faults in the database been already simulated?

  ➔ Yes: End.

  ➔ No: Go to step 2!

Additional Information:

- often modified due to performance issues
- statistical or even heuristic methods in use
- fault grading, fault dropping, etc.

➢ *Automatic Test Pattern Generation (ATPG):*

Working Principle:

- step 1: derivation of fault data base from the netlist (mostly at gate-level)
- step 2: construction of fault netlist by inserting one fault from the fault data base into the fault-free netlist
- step 3: find a stimuli that detects the fault by applying an algorithm (for example the D-algorithm)
- Have all faults in the database been already simulated?

  ➔ Yes: End.

  ➔ No: Go to step 2!

Additional Information:

- NP-complete problem
- no stimuli needed
- good results for combinational circuits (complete generation possible for example with D-algorithm)
- bad results for sequential circuits (no complete generation possible)
- make use of scan chains for avoiding sequential circuits

CMOS Fault Models

Physical Failures

- Possible Defects:
  - *wafer fabrication:* lattice defects, pollution
  - *epitaxy:* lattice defects
  - *oxidation:* pinholes
  - *deposition:* masking particles
  - *diffusion and implantation:* lattice defects, particles
  - *etching:* over- and underetching
  - *photolithography:* cracks, dust, misalignment
- Global Defects: defects that affect all dies on a wafer in the same way

  examples: PTV, mask failures, etc.
- Local Defects: defects that affect only certain dies on a wafer (statistically distributed)

  examples: OCV, etc.

Abstraction Layers for Fault Models

Algorithmic Level	
Architectural Level	
Register Transfer Level (RTL)	
Gate Level	*Single-Stuck-At-Fault Model*
Switch Level	
Transistor Level	*for CMOS: models for behaviour analysis*
Layout Level	
Physical Level	

CMOS Faults – Transistor-Level Fault Models

- Main Problem of CMOS: Static CMOS circuits are – due to its n- and p-network – inherently redundant.
- Robust Test: An input sequence that correctly tests a complex CMOS gate is called robust test.
- Faults in CMOS Circuits:
  - *interrupted wires:* unpredictable voltage levels, long settling points due to leakage currents
  - *transient input pattern faults:* undetectable faults due to transient effects
  - *transistor-stuck-on faults:* bridging fault between drain and source of a transistor, settling somewhere between VSS and VDD, difficult to model
  - *delay faults:* partial broken connections may lead to slow ramp times and therefore to excessive delays

A Specific CMOS Fault Model: Jacomet's Layout-Dependent Fault Model

- *Stuck-Open-Faults:* transistor-stuck-open faults, a vector sequence is sufficient for detecting a fault
- *Stuck-Toggle-Faults:* transistor-stuck-open faults, a toggle-test is sufficient for detecting a fault
- *Stuck-At-Faults:* according to the classical SSAF model
- *Wired-Logic-Faults:* bridging faults
- *Large-Scope-Short Faults:* wired-logic-faults that result in different behaviour of several gates in the fan-out tree due to unequal threshold levels

IDDQ-Test for CMOS Circuits

- Quiescent Supply Current
- Measured after test application when the circuit nodes become settled.
- Practical Observation: Defect chips have got a high quiescent current!
- Other Methods: use of reference chips, on-chip current measurement, etc.
- Problems:
  - long waiting time for settling due to high dynamic CMOS currents
  - circuits with static power dissipation (constant currents)
  - increasing leakage currents
  - sensitivity limit of 200μA
- Improvement: IRRQ test for the future (see p. 40f.)

Testability Analysis, Fault Simulation

Basic Idea

- Gathering information about the testability of a circuit!
- Problem: Design hierarchy is not a test hierarchy!

Sandia Controllability Observability Analysis Program (SCOAP)

- Basic Idea: Every node gets a number that describes the testability in terms of controllability and observability.
- suitable for sequential and combinational circuits
- no need of stimuli vectors
- ***Controllability Equations:***
  - combinational controllability of node X with bit level n: $CC^n(X)$
  - incremental term for describing the combinational depth of the circuit:

    $CC^n(Y) = CC^m(X) + ... \underline{+ 1}$
  - sequential controllability of node X with bit level n: $SC^n(X)$
  - incremental term for describing the sequential depth of the circuit (only for sequential bistable equations):

    $SC^n(Y) = SC^m(X) + ... \underline{+ 1}$
  - example: 2-XOR-gate

    $CC^0(Y) = \min\{CC^1(X_1) + CC^1(X_2), CC^0(X_1) + CC^0(X_2)\} + 1$

    $CC^1(Y) = \min\{CC^0(X_1) + CC^1(X_2), CC^1(X_1) + CC^0(X_2)\} + 1$

    $SC^0(Y) = \min\{SC^1(X_1) + SC^1(X_2), SC^0(X_1) + SC^0(X_2)\}$

    $SC^1(Y) = \min\{SC^0(X_1) + SC^1(X_2), SC^1(X_1) + SC^0(X_2)\}$
  - example: positive edge-triggered flipflop with active-low reset

    $CC^0(Q) = \min\{CC^0(R), CC^1(R) + CC^0(D) + CC^1(C) + CC^0(C)\}$

    $CC^1(Q) = CC^1(R) + CC^1(D) + CC^1(C) + CC^0(C)$

    $SC^0(Q) = \min\{SC^0(R), SC^1(R) + SC^0(D) + SC^1(C) + SC^0(C)\} + 1$

    $SC^1(Q) = SC^1(R) + SC^1(D) + SC^1(C) + SC^0(C) + 1$
- ***Observability Equations:***
  - Build a D-cube for the input of interest:
    - Set the input (A) to be analysed to D or $\bar{D}$.
    - Set the remaining inputs (B_i) to 0, 1 or X so that the combinations produce a D or $\bar{D}$ at the output (out_i).
  - Build observability equation as followed:

    $CO(A) = \min\{\sum CO(out_i) + \sum CC^{0,1}(B_i)\} + 1$

    X states do not need to be included!
  - The incremental rule is the same as for the controllability.

- Attention, special rules for bistables! See p. 46f.!

- ***SCOAP Algorithm:***
  - Step 1: Initialization of Primary Inputs (I) and Internal Nodes (N).
    - $CC^0(I) = CC^1(I) = 1$
    - $CC^0(N) = CC^1(N) = \infty$
    - $SC^0(I) = SC^1(I) = 0$
    - $SC^0(N) = SC^1(N) = \infty$
  - Step 2: Determine the controllability numbers for all Internal Nodes beginning at the Primary Inputs. This is an iterative process.
  - Step 3: Initialize the Primary Outputs (Y).
    - $CO(Y) = 0$
    - $SO(Y) = 0$
    - $CO(N) = \infty$
    - $SO(N) = \infty$
  - Step 4: Determine the observability numbers for all Internal Nodes beginning at the Primary Outputs. This is an iterative process.

- ***Advantages and Disadvantages:***

Advantages:	**Disadvantages:**
• fast calculation	• no exact improvement
• for combinational and sequential circuits	• testability analysis based on arbitrary assumptions

Fault Simulation

Fault simulation is normally carried out on a gate-level netlist.

Overview

- Defect Level:
  - percentage of delivered chips with faults
  - equation: $DL = 1 - Y^{1-T}$
- Fault Coverage: $T = \frac{\text{amount of detectable faults}}{\text{amount of all possible faults}}$ (typical: 99%)
- Yield: $Y = \frac{\text{amount of error free chips}}{\text{amount of produced chips}}$ (typical: 20-80%)

Aims of Fault Simulation

- justification of test vectors (for example from ATPG)
- improvement of ATPG
- fault coverage calculation

- diagnostics and debugging of faulty circuits
- diagnostics of faults
- improvement of testability, DFT

Methods for Fault Simulation of Combinational Circuits

- Serial and Parallel Fault Simulation:
  - basic idea: serial and parallel simulation of the circuit for every input fault
  - advantages: very simple
  - disadvantage: very inefficient
  - today: multi-cores in favour of parallel simulation
- Deductive Fault Simulation:
  - basic idea: list of faults are propagated and calculated
  - disadvantages: simulation leads to huge lists, updating of lists at paths without any activities, high computation effort
- Concurrent Fault Simulation:
  - basic idea: Link new faults just when they happen into the simulation and diverge them. But remove them if they converge.
  - Terms: Fault Origin (FO), Fault Effect (FE), Input / Output Fault Origin (IFO / OFO)
  - advantages: efficient, flexible with respective to different fault models,
  - disadvantages: high memory usage, a lot page faults possible, prediction of performance difficult

Methods for Sequential Fault Simulation of Sequential Circuits

- The test vector set also depends on the order of the vectors!
- high computation effort due to rising fault effects

Performance Issues of Fault Simulation

- mostly depends on the size of the circuit
- no optimal fault simulation for every circuit type
- performance improvement:
  - fault dropping
  - circuit ordering
  - statistical fault simulation with fault coverage estimation
  - states applied analysis

Obtaining Test Vectors*

Sources for Test Vectors

- *from Simulation:* good fault coverage (around 90%) possible
- *from ATPG:* very compact test vectors with very high fault coverage, suitable for production test
- *from BIST:* enhanced test speed and also very good fault coverage possible

General Guidance

- Make compact test vector sets by rearranging circuit components!
- Use standard timing scheme!
- Document your test files properly!
- Provide separate vector files for functional and for scan test!

Important Points for Test Vector File

- well-defined data for each pin
- identification of each vector column with a pin name
- direction control signal for each bidirectional pin
- IEEE 1164 compatible characters for indicating the values
- time stamp per vector line

Automatic Test Pattern Generation (ATPG)

- often enhanced
- with scan chains good observability and controllability possible
- stick to full-scan instead of partial scan
- block isolation for RAMs or other special components of the chip

Sensitive Cases

- Asynchronous Interfaces:
  - suitable synchronization mechanism for testing (for example plesiochronous interfaces)
  - testing of the synchronization mechanism itself
  - advices: keep simple, predictable synchronous data transfer, guarantee full controllability and observability
- Tristate and Bidirectional Signals and Pins:
  - much more logic states
  - functional verification, fault modelling and test generation is aggravated
  - special timing requirements
  - special IC tests

Testing Complex Circuits

- State Scan Tests
- Co-Simulation of External Circuits
- Expanding Chip Functionality for Simulation and Test

Automatic Test Vector Generation

Boolean Difference

- Basics:
  - syntax notes:

a	b	a xor b		+ : or
0	0	0		· : and
0	1	1		- : nor
1	0	1		
1	1	0		

  - boolean input vector: $X=\{x_1,\dots,x_n\}$
  - one dimensional difference:

$$\frac{dF(X)}{dx_i}=F_i(x_1,\dots,x_i,\dots,x_n)\,\text{xor}\,F_i(x_1,\dots,\bar{x}_i,\dots,x_n)$$

$$\frac{dF(X)}{dx_i}=F_i(1)\,\text{xor}\,F_i(0)\quad\text{with}\quad F_i(a)=F(x_1,\dots,x_{i-1},a,x_{i+1},\dots,x_n)$$

  - two dimensional difference:

$$\frac{dF(X)}{d(x_i,x_j)}=F(x_1,\dots,x_i,\dots,x_j,\dots,x_n)\,\text{xor}\,F(x_1,\dots,\bar{x}_i,\dots,\bar{x}_j,\dots,x_n)\quad\text{with}\quad i<j$$

- Rules:
  - $\overline{\frac{dF(X)}{dx_i}}=\frac{dF(X)}{dx_i}$
  - $\frac{dF(X)}{dx_i}=\frac{dF(X)}{d\,\bar{x}_i}$
  - $\frac{d}{dx_i}\frac{dF(X)}{dx_j}=\frac{d}{dx_j}\frac{dF(X)}{dx_i}$
  - $\frac{d[F(X)\,\text{xor}\,G(X)]}{dx_i}=\frac{dF(X)}{dx_i}\,\text{xor}\,\frac{dG(X)}{dx_i}$
  - $\frac{d[F(X)G(X)]}{dx_i}=F(X)\frac{dG(X)}{dx_i}\,\text{nor}\,G(X)\frac{dF(X)}{dx_i}\,\text{nor}\,\frac{dG(X)}{dx_i}\frac{dF(X)}{dx_i}$
  - $\frac{d[F(X)+G(X)]}{dx_i}=\overline{F(X)}\frac{dG(X)}{dx_i}\,\text{nor}\,\overline{G(X)}\frac{dF(X)}{dx_i}\,\text{nor}\,\frac{dG(X)}{dx_i}\frac{dF(X)}{dx_i}$

Test Generation for Combinational Circuits

Test Generation with Boolean Differences

- Basics:
  - $F(X)$: circuit without faults
  - $F_a(X)$: circuit that contains one fault

    for s-a-0: $F_a(X)=F_i(0)$ for s-a-1: $F_a(X)=F_i(1)$
  - $F(X)$ nor $F_a(X)=1$
  - condition for s-a-0: $x_i\frac{dF(X)}{dx_i}=1$
  - condition for s-a-1: $\overline{x_i}\frac{dF(X)}{dx_i}=1$
- Partial Analysis: $F^*(X,h)=F(X)$ with $h=h(X)$
  - condition for s-a-0: $h\frac{dF^*(X,h)}{dh}=1$
  - condition for s-a-1: $\overline{h}\frac{dF^*(X,h)}{dh}=1$
- Advantages:
  - multiples faults can be also handled
  - also parts of a circuit can be analysed
  - finds all tests for a certain fault
- Disadvantages:
  - high calculation effort and memory usage
  - algebraic manipulations

D-Algorithm of Roth (1966)

- Meaning of 'D':

		Circuit with Fault	
		0	**1**
Circuit without Fault	**0**	0	$\bar{D}$
	1	D	1

- Singular Cover (SC), Singular Cube (SC) or Primitive Cube (PC): logic function of a gate

  e.g. SC for an AND-gate:

a	b	a AND b
0	x	0
x	0	0
1	1	1

- Primitive D-Cube of the Fault (PDCF): intersection of SC pairs with correct and incorrect output value

  e.g. PDCF for an AND-gate:

a	b	a AND b
0	x	$\bar{D}$
x	0	$\bar{D}$
1	1	D

- Propagation D-Cube (PDC): intersection of SC pairs with different output values

  e.g. PDC for an AND-gate:

a	b	a AND b
D	x	D
1	D	D
$\bar{D}$	1	$\bar{D}$
1	$\bar{D}$	$\bar{D}$

- Core Elements of the Algorithm:
  - Single Path Sensitization: forward-trace, error propagation phase
  - Line Justification: backward-trace, justification phase
  - works with backtracking
- Algorithm:
  - Step 1: Initialize test cube (set all nodes to X).
  - Step 2: Select a PDCF (Fault Production).
  - Step 3: Intersect the selected cube with the test cube and perform Implication (Fault Propagation).
  - Question: Is a primary output reached?
    - ➔ No: Go to Step 4!
    - ➔ Yes: Go to Step 5!
  - Step 4: Choose a PDC on a path to the primary outputs (D-Drive).
  - Step 5: Perform Line Justification.
  - Question: Was Line Justification successfully terminated?
    - ➔ No: Go to Step 2 and choose another PDCF! If there is no other PDCF available, terminate!
    - ➔ Yes: A test has been generated successfully. Put the test cube (final test set) into the fault dictionary. Jump to Step 2 for generating a new test set, if there is another PDCF.
  - End.

Implication:

- Step 1: Look at the input and output values of a gate.
- Step 2: Is the gate fully defined?
  - ➔ No: End.
  - ➔ Yes: Intersect the test cube with the corresponding input and output values of the gate.
- End.

Line Justification:

- Question: Does the test cube include Xs?
  - ➔ No: End.
  - ➔ Yes: Go to next step!
- Step 1: Select an unjustified line and a well-defined gate with a singular cube to justify it.
- Step 2: Intersect the chosen singular cube with the test cube.
- Question: Was Line justification successful? - The Line Justification is successful if there is no empty entry (Ø) in the test cube.
  - ➔ No: Go to step 1 and choose another gate! If no other gate is available, terminate with "Line Justification failed!".
  - ➔ Yes: Terminate with "Line Justification was successful!".
- End.

- Advantages:
  - no boolean expression handling
  - complete: It delivers a test set for a fault if one exists!
- Disadvantages:
  - cannot be used for sequential circuits

Test Generation for Synchronous Sequential Circuits

Extended D-Algorithm

- Iterative Combinational Circuit with Iterative Depth of p: Decompose the sequential circuit into so called time frames (duplicates).
- Problems:
  - not complete: It does not deliver a test set for a fault even if one exists!
  - Growing depth p leads to a higher probability of success but also requires more computation effort.
  - A single error corresponds to a p-fold error. It may happen that a fault blocks another one.
  - Line Justification often fails due to the requirement that the initial state must remain X.
  - At least the primary outputs of one time frame must deliver a distinguishable result.

- Improved Line Justification: A dummy duplicate often solves the problem that the initial states must remain at X. However, this improved Line Justification does not work in every case.
- Loop Detection to avoid unforeseen test results.

D-Algorithm enhanced with a 9-Value Model

- Syntax of a Value: $n_i = a_i / b_i$

  a_i : circuit without faults b_i : circuit with faults

- 9-Value Model:

$0/0 = 0$	$1/0 = D$	$X/1$
$X/X = X$	$0/1 = \bar{D}$	$0/X$
$1/1 = 1$	$1/X$	$X/0$

- Intersection of two values: $n_1 \cap n_2 = (a_1 \cap a_2) / (b_1 \cap b_2)$
- Advantages:
  - more degree of freedom
  - Line Justification problems of the classical D-Algorithm are solved.
  - complete: It delivers a test set for a fault if one exists!
- Disadvantages:
  - More complexity leads to more computation effort!

Other Algorithms (mostly based on D-Algorithm)

- Path Oriented DEcision Making (PODEM):
  - developed in 1981 by Goel
  - The D-Algorithm is exponentially complex to the number of internal circuit nodes. XOR-gates make the complexity of the D-Algorithm approach this limit.
  - PODEM is a branch-and-bound algorithm which is exponentially complex ot the number ofr circuit inputs – usually a much smaller number than circuit nodes.
  - It has troubles with areas of reconvergent fanout (also a problem for the D-Algorithm): They can cause complex interactions between internal circuit nodes!
- FANout oriented algorithm (FAN):
  - improvement of PODEM
  - designed to utilize circuit topology information to increase search efficiency
  - solves the reconvergent problem of the D- and the PODEM Algorithm
- others:
  - heuristics methods
  - pruning the search space!

Test Generation for Asynchronous Sequential Circuits

- Problems:
  - oscillations and hazards
  - fault manifestations difficult to predict
  - very complex models combined with high computation effort
- Test generation algorithms were developed!
- Advices:
  - Stay away whenever possible.
  - Design only locally restricted asynchronous circuits.
  - Make them testable by special circuitry. For example by making them synchronous in test mode or alike.

Automatic Test Equipment

Design Verification

- Pre-Silicon Verification: timing verification with gate-level simulation
- Post-Silicon Verification: timing specification with chip tester

First Post-Silicon Verification

- Verification of Functionality
- Detection of Potential Fabrication Faults
- Timing Specification and Characterisation

Strobe Formats

- Edge Compare (standard)
- Window Compare

Stimulus Formats

- Delayed Non-Return-to-Zero (DNRZ) (standard format)
- Return-to-Zero (RZ)
- Return-to-One (R1)
- Return-to-Complement (RC)

Measurements

- Setup and Hold Time Measurement: see p. 94ff.
- Propagation and Contamination Delay Measurement: see p. 97ff.
- Delay Measurement for Bidirectional Pins: see p. 98ff.
- Measurement of High-Impedance and Low-Impedance Delays: see p. 100
- Characterization of Maximum Operating Frequency: see p. 100ff.

Signal Integrity

- Test Setup: see p. 103!
- General Rule:
  - *output pins of a DUT:* corresponds to the wave impedance of the transmission line (normally 50Ω), often additional resistors on the PCB necessary
  - *input pins of a DUT:* very high resistance
- Reflection r: $r = \frac{R_{drain} - R_{source}}{R_{drain} + R_{source}}$
- Power Connections
- Signal Connections
- Testing of Bidirectional Connections
- see pp. 106 and 107!

Automatic Test Equipment (ATE)

- ***Automatic Parametric Measurements:***
  - Continuity Test (Diode Test)
  - Standby Current
  - Input Current Test
  - Output Voltage Test
  - Operation Current Test
- ***Interactive Parametric Measurements:***
  - *Linear Sweep Tests:* stepping of certain parameters
  - *Shmoo Tests:* two-dimensional grid, for example operating frequency versus supply voltage
  - *Waveform Measurements:* one sample (standard), multiple samples (hires), threshold-sampling (scope)
- Memory related Problems: see p. 113
  - break vectors: static and dynamic CMOS logic
  - sequencer loops
  - changing parameter sets

Design Verification and Debugging

Diagnostics with ASIC Testers

- only access to primary inputs and outputs
- no access to inner nodes
- fault diagnostics with fault dictionaries:
  - generation of fault dictionaries with fault simulators
  - decision tree (see p. 118ff.)

Diagnostics with Special Test Tools

- Liquid Crystal Fault Diagnostics
- Infra-red Camera
- Mechanical Probing:
  - Guided Probing – Signature Testing
  - Guided Probing with Signature Compression
- E-Beam Methods:
  - Qualitative Voltage Measurement:
    - *Static Voltage Contrast:* blocking of electrons (black positive region), non-blocking of electrons (white negative region)
    - *Voltage Coding:* video sequences of voltage contrast changing
  - Quantitative Voltage Measurement:

$$I_{SE} = I_{PE} \cdot \int_{e(V_P - V_G)}^{50eV} \delta(E)\, dE = const \quad \text{with} \quad V_P - V_G = const \Rightarrow \Delta V_P = \Delta V_G$$

    V_P : voltage at hitting the surface V_G : voltage at the detector I_{PE} : current of the e-beam

    $\delta(E)$: yield of electrons (secondary electrons or backscattered electrons)

  - Sampling Techniques:
    - Stroboscopial Voltage Contrast
    - Logic State Representation
    - Timing Diagram
    - Waveform Measurements
  - Combination with CAD systems possible and very effective!
  - Disadvantages:
    - high acquisition costs
    - needs open chips and preparation effort
    - destructive measurement
    - time-consuming method
    - needs a lot of experience
    - simulation of the IC with an IC tester with suitable test vectors

- Focused Ion Beam (FIB) Method:
  - imaging resolution: 4-5nm
  - Gas-Assisted Etching (GAE) of materials: with e.g. XeF_2, F_2, Cl_2, ...
  - deposition of materials (insulators and conductors): e.g. SiO_2, $W(CO)_6$, ...
  - most often with SEM ability
  - placing probe points
  - make use of so called “sewing kits”
  - good contrast mechanism: material, topographic, crystalline (or channelling) and voltage

Surface Analysis in Microelectronics

- Main Goals:
  - Structural Characteristics
  - Compositional Characteristics
- Defects in Semiconductors:
  - pure electrical effects: impurities
  - electrical and mechanical effects: stress (tensile and compressive forces)
- Choosing an Analytical Technique:
  - intention: What tests are needed? Detection of elements? Visual inspection? ...
  - destructibility: Will the IC be used afterwards?
  - detection limits: Is the method able to detect the intended elements, compounds or alike?
  - sampling depths and spatial resolution: Is the resolution and sampling depth sufficient?

Surface Analysis Techniques

- ***Surface Structure Analysis:*** Identification of the morphology of a sample surface
  - Scanning Electron Microscopy (SEM):
    - Detection of Secondary Electrons
    - Detection of Backscattered Electrons
    - Detection of Auger Electrons
  - Scanning Tunnelling Microscopy (STM)
  - Atomic Force Microscopy (AFM)
- ***Elemental Analysis:*** Identification of elemental constituents of a sample surface
  - Auger Electron Spectroscopy (AES)
  - Characteristic X-Ray Analysis:
    - Energy-Dispersive X-Ray Spectroscopy (EDX or EDS)
    - Wavelength-Dispersive X-Ray Spectroscopy (WDX or WDS)
  - Ion Scattering Spectroscopy (ISS):
    - Low Energy Ion Scattering (LEIS)

- Rutherford Backscattering Spectrometry (RBS)

- ***Compound Analysis:*** Identification of molecular constituents of a sample surface
  - X-Ray Photoelectron Spectroscopy (XPS), also called Electron Spectroscopy for Chemical Analysis (ESCA)
  - Secondary Ion Mass Spectrometry (SIMS)
  - Laser Microprobe Mass Spectrometry (LMMS)
- ***Complementary Methods:***
  - Optical Probing Techniques:
    - IR and UV Spectroscopy
    - Raman Spectroscopy
    - Light Scattering Techniques
  - Mass Spectroscopy
  - Gas or Liquid Chromatography

	SEM	**EDX or WDX**	**AES**	**ISS**
excitation source	electron beam	electrons	electrons	ion beam
detected emission	secondary electrons, backscattered electrons or Auger electrons	characteristics X-ray	Auger electrons	scattered ions
	SIMS	**XPS**	**LMMS**	
excitation source	ion beam	X-ray	laser beam	
detected emission	sputtered ions	electrons	evaporated ions	
	AFM		**STM**	
detection principle	A cantilever with a tip is in proximity to the surface. Forces deflect the cantilever. The deflection can be measured either piezoelectrically or optically.		A tunnelling current begins to flow at a certain distance between probe tip and surface. A feedback controller looks for constant current by adjusting piezoelectrically the distance between the tip and the surface.	

Appendix

Combinational and Sequential Circuits

- **Combinational Circuits:** The output signals depend only on the present input signals. Therefore combinational circuits are memoryless.
- **Sequential Circuits:** The output signals depend on the present input signals and on inner signal states that are stored in memory components called bistables. Therefore sequential circuits are memorizing. They are subdivided into the following categories:
  - Synchronous Circuits: All events are triggered by clock signals. Normally there is only one clock signal. When there are more than one clock signal, the circuit is subdivided into so called clock domains.
  - Asynchronous Circuits: Events are triggered by any input signals.