

Optimization Techniques for Digital VLSI Design
Dr. Chandan Karfa
Dr. Santosh Biswas
Department of Computer Science & Engineering
Indian Institute of Science, Guwahati

Module - 4
Lecture – 13
VLSI Testing

So, welcome to the fourth module on VLSI testing of the course on optimization techniques for digital VLSI design. So, as you have seen in the first three modules, the Doctor Chandan Karfa has taught you about what is the optimisation techniques which cut caters to the design part of the VLSI or the digital design flow, but one such it has been fabricated or it has been gone to the all the design steps and which is been sent for the fabrication, then after it comes out of the fab lab, it has to be tested because the idea is that in case of VLSI, we or in case of digital design at the higher level of integration.

So, what happens basically is a very large factor of these circuits may not be have be compared to traditional in traditional systems more large number of circuits may have some inbuilt manufacturing faults. So, all circuits has to be tested before they are sent to the market, otherwise, there can be issues like there can be system failures or you will be you will be sending as Chipley customer which will be not working properly and the system may have failures. So, to avoid all this, we try to test all the parts or all the chips before they are sent to the market. So, as we are discussing continuously in the course that this is actually the optimization techniques on basic cad flow.

So, this first two lectures basically one and two, in this module will quickly give you give some what is traditional VLSI testing and then will go for the optimization techniques which will be applicable on the basic test flow.

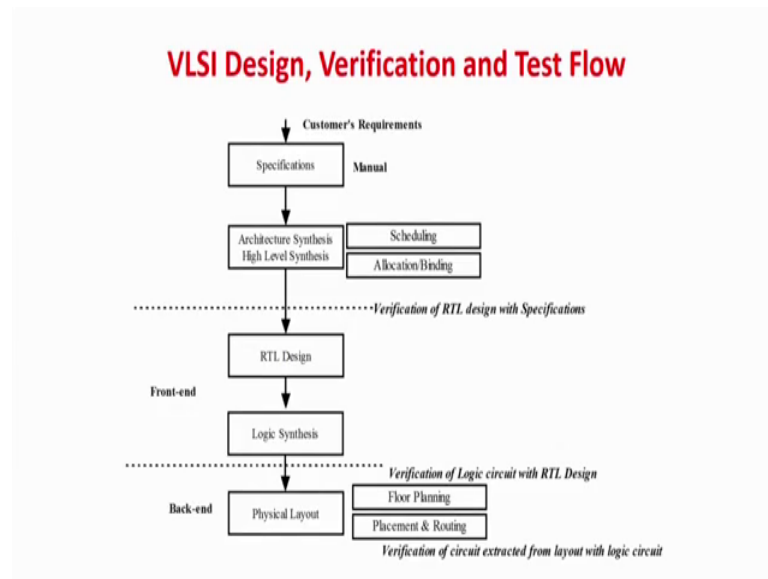
(Refer Slide Time: 02:00)

Course Plan
Module 4: VLSI Testing
Lecture 1 and 2: Introduction to Digital VLSI Testing, Automatic Test Pattern Generation (ATPG), Design for Testability
Lecture 3: Optimization Techniques for ATPG
Lecture 4: Optimization Techniques for Design for Testability
Lecture 5: High-level fault modeling
Lecture 6: RTL level Testing

So, today's and tomorrow's lectures basically will be introduction to digital VLSI automatic test pattern generation and design for testability. So, this part actually is a background which will be required to understand the other advance lectures on the optimization techniques for VLSI testing.

So, basically we are trying to keep the course as self contain. So, that you do not have to go back and forth with reading about the normal cad for VLSI or VLSI testing course and then trying to look at the advance stuff for optimization in testing. So, the first two lectures will be trying to quickly cramp up the stuff in VLSI testing which is used for traditional maybe with the chips which were fabricated till about early 2000s what are the techniques and how testing was performed.

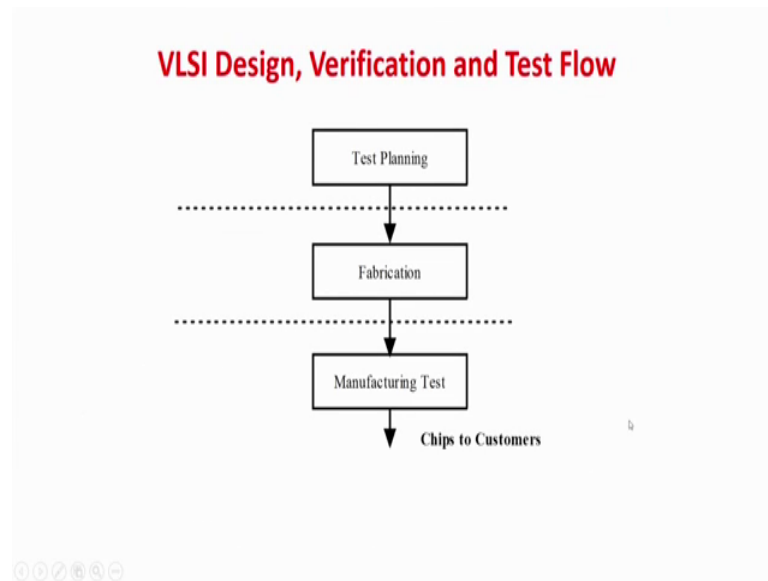
(Refer Slide Time: 02:44)



So, basically, as I have already discuss, this is a traditional VLSI flow that we start with basically these specification. So, this is started with specification, then we go for architectural synthesis then go for RTL design, then logic synthesis and then floor plan and layout.

So, basically this is what is the design part and how basically advanced for advance systems how what are the optimization techniques how things have been getting how things are getting modified or how algorithms are getting modified when the very large scale systems has already been covered in the last 3 modules.

(Refer Slide Time: 03:17)



Next part basically is the test that is this planning then basically we have to plan test before the fabrication starts and in fact whenever you talk about real test that is we apply the test patterns and see whether all the manufacturing manufacturer chips are operating properly or not is actually call the manufacturing test. So, this last step basically.

So, in this part, basically we did plan; what are the test patterns what are the test patterns to be applied, how we are going to apply, there is there anything extra unit to put in the circuit to do the testing etcetera and finally, after fabrication, we put the actual patterns signals and look at the outputs and see whether everything is proper or not. So, this is the basically VLSI design and test flow and in fact every party will find that there should be a when you are moving from one phase to another there should be equivalence.

So, basically we go for formal very formal verification methods which will be covering in the next module that how formal verification can be optimized two and a large systems.

(Refer Slide Time: 04:13)

Introduction to Philosophy of Testing

- “If anything can go wrong, it will”--A very well known statement known as Murphy’s Law.
- **Testing** a system comprises subjecting it to inputs and checking its outputs to verify whether it behaves as per the specifications targeted during design.

So, anyway; so, as told what is testing why it is required basically? So, basically a very well known theory we call is Murphy law if anything can go wrong it will; that means, if basically whatever system you make there is a very there is a chance that it may malfunction. So, in fact testing means basically as simple as you take a system applies a inputs and you will find out whether the desired outputs are appearing at the output of the output or it is behaving as per the specification or not.

So, simple testing in case of VLSI’s also that means the philosophy is similar, but unless of physical system why we require testing as a special litigation or say special part into a VLSI or digital circuits or large systems is one of the very important things, we should understand and which will also look at in this preliminary lectures.

(Refer Slide Time: 05:03)

Example: Electrical Iron

- Plug it in 220V AC and see if it is heating.
 - “functional” specification, *that also partially.*
- Safety:
 - All exposed metal parts of the iron are grounded
 - Auto-off on overheating
- Detailed Functionality
 - Heating when powered ON.
 - Glowing of LED to indicate power ON.
 - Temperature matching with specification for different ranges that can be set using the regulator (e.g., woolen, silk, cotton etc.)

For example, I take a very classical system like a like an electrical iron I want to test it. So, we know that it is a very traditional system, not many of the electrical irons, I am taking talking about the very simple iron may be which we are using in daily life not much of them will have faults. So, basically what you do when we purchase it, we put it in a 220 volts AC source and see that whether it is functionally proper or not it is getting heated up or not this is actually a testing part.

Now, basically this is actually call a partial testing because testing can be as elaborate checking whether the steal or material which is made the on from which the electrical iron is made is above, how much it is prone to failure, how much it is brittle all those testing can be another extreme, but basically nobody does this because we have to test within a proper amount of time and also it should be giving the required quality of service to the customer.

Basically this is maybe done for very partially that whether it is function or not may be you are your purchasing a very cheap iron only this amount of test will be done and it will be sold to the customer maybe basically, if you are going to slightly higher function than people will also check for whether basically, they all the parts of the iron electrically grounded, there is no chance for electrical shocks, etcetera, then you also check for the thermal sate, etcetera detailed functionalities, rarely people go for that level of testing

that if there is a thermal state, you have different temperatures tools silk different type of clothes, you select whether for everything the temperature range is maintained or not.

(Refer Slide Time: 06:40)

Example: Electrical Iron

- Performance
 - Power consumption as per the specifications
 - Time required to reach the desired temperature when range is changed using the regulator

Tests for ONLY electrical parameters.

- Tests for *mechanical parameters*, like maximum height from which there is resistance to breaking of plastic parts if dropped on a tiled floor etc.
- Number of tests performed depends on the time, equipments etc. which in turn is decided by the target price of the product.

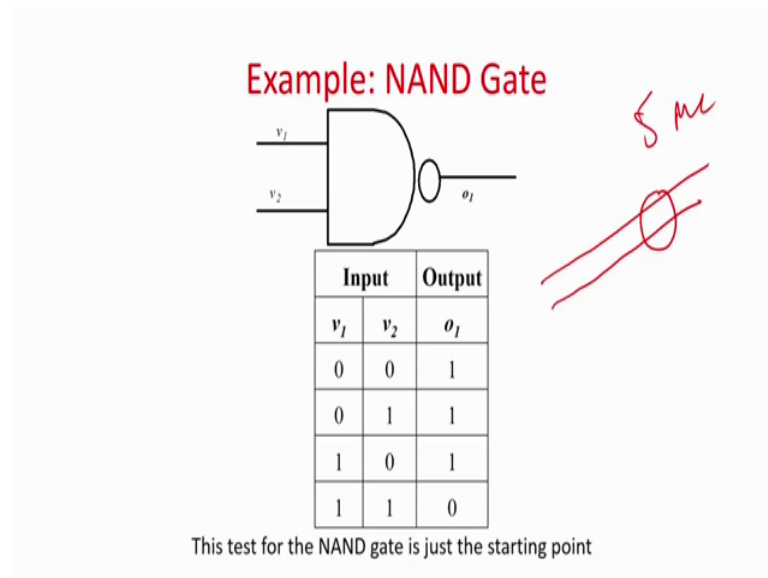
So, similarly you can find out a huge number of test parameters like performance of power if it claims that it is 5 star rating or 4 star rating in terms of electrical power consumption whether it is appearing whether it is matching to that or not, then as I told you like a very rare case you can also check mechanical parameters like if it falls down whether it will break, etcetera, etcetera, etcetera.

So, testing can be huge in size, but practically nobody does all those stuff because that will neither make it logically feasible within the time limit for the for a product under consideration; obviously, if it is meant for some avionics or some rocket technology, of course, we will do all positive parts of the test and all extent of testing. So, that nothing fails or very very rare chance of failure happens, but when we are talking about home appliances nobody does testing on that level. So, testing basically is to verify whether a subsystem is operating as per the specifications subject to time and cost benefits. So, depending on this system I will decide how much to test. So, that I get it in proper time and also it is profitable.

Now, if I tell all electrical iron tested or whenever you are purchasing as simple as a tube light or bicycles or a motorbikes, how much testing is done, of course, compared to

circuit is very very few because these are very traditional, if you compare last 200 years, the electrical iron or and all the classical engineering problems are more or less similar slight modification has been done over the years.

(Refer Slide Time: 07:55)



So, so what happens basically the design part is more or less stabilizes. So, if you find some kind of errors which is happening a large number of samples you rectify the design process. So, maybe one part will one thousand will have defect or maybe 1 in 1 lakh maybe 2 or 3 parts will be defective.

So, testing is a important, but but not as important as in case of circuits. So, in classical system testing is done, but it is actually because more or less we know that mostly is all the systems are operating fine and it has been tested overtime and not much technology changes has been there because if you compare of fan which is running. Now pro fan which is running more than say about 50 to 60 years back, more or less the technology remain same. So, more or less the technology is matured and basically not of that much of deviation happens and so, testing is not of that important as you consider as a circuit.

Now, what is the special issue about circuits? So, you know that within last 20 to 30 years, we have come from high micron technology to a was about say 45 nanometer technology. So, when you are actually going for such sophisticated technology charms

like you like we all know the electronic gadget changes that is happened over the last 20 years is phenomenal a person who has seen a mobile phones last 10 year or a telephone.

Last 10 years back, he will be astonish to see a handle gadget or a smart phone or a tab that what is the amount of phenomenal changes that has happened in the electronic technology. So, why it is possible because you are able to make very very complicated systems like NOC and SOC on a single electronic circuit chip, how it is possible because you are continuously shrinking the technology for 5 nanometres sorry for 5 microns to 0.45 for even lower micron technology.

So, lower micron means whatever fabric in the fabrication technology whatever lines and wires you are drawing the distance between them is started becoming shrinking. So, once you started shrinking the distances the problem happens is the lot of failures happen, why because when I working in 5 micron technology, it takes some time for the technology to be matured and to find out what are the manufacturing defects that is happening regularly how to change the technology for manufacturing, etcetera.

But before further technology matures, a new technology comes up like for 5 micron, you go to 3 microns, then you quickly go to 1.8 microns, why it happen because you have to market sophisticated and complex product to the market to get get high end products and catch ward of the market because nobody wants to see a how a technology matures weight and find out it is very robust by the times, some the other vendor or other company will go for a higher micron lower micron technology where you can put more transistors in a chip and this system becomes more complex.

So, in VLSI, unlike traditional systems huge change in technology is happening. So, due to the huge change in technology, what is the problem that happened is before a fabrication technology matches, we quickly jump to another technology or more sophisticated or lower submicron technology and it keeps on happening. So, lower technology is getting enough time to be matured. So, that mostly devices are fault free. So, what is the profit idea? They say that I will manufacture using a newer technology. So, that I get sophisticated circuits, but then what, then what will happen you are going to have lower yield?

So, lower yield means say if you are manufacturing 100 ics, maybe 20 to 30 will be having defects manufacturing defects because the technology which which which I am manufacturing the circuits have not yield matured, but still I have take a risk because otherwise some other people or other companies will use a newer technology will give you more sophisticated electronic gadget and I will be left out in the market. So, what I have to I have to manufacture circuits with the more complex technology of course, there will be manufacturing defects, but what I will do test it, I have to test. Now all the chips because any because if you are saying that a product line has 30 percent failure in it or 70 percent yield, let 30 out of 100 may have defects.

So, I have to test all the 100 chips, I have to throw out the 30 and the 70, I will ship into the market make the price higher, but still I will make profit because I will I will be able to have a huge market in place, then I will be able to make something which is not being sold or not being marketed by my competitors. So, that is the idea of the VLSI court on court, I will take very immatured technology I will manufacture a large number of chips test all the chips because I know that yield maybe as low as 70 percent or 60 percent even no fabrication company basically tells what is the yield.

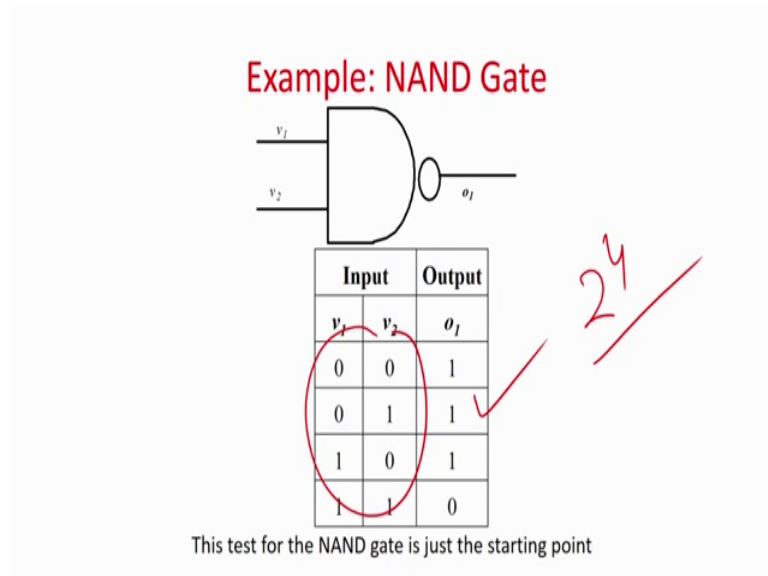
But the number is not as high as 90 percent plus it is lower, I test all the chips throw away the fail faulty ones and ship the correct once to the customer with the higher price value, but still I can give you a more complex and a sophisticated product and I come into the market, but that does not happen in the classical technology because in classical technology more or less the technology has remain same in over 100S of or tens of or 50 of a 50 years or so. So technology have matured, the yield is very very high say about more than 99 percent.

So, that is why testing in VLSI is very very important compared to classical system. Now if I tell you what is testing in VLSI, it is as simple as this slide take a NAND gate, apply all the 4 patterns and you are going to get the output. So, if you are going to imply the input get the output, you are fine, but you have to note that this is the hardware. So, I have to apply it to the all the chips, for example, if I have manufacture say 10,000 NAND gates may be out of that 100 will have some failures because the technology to use fabricate the NAND gate is not matured, but still I using the technology because now my NAND gates size will be very very small in the nanometer technology and I can

cramp up millions of transistor and gates in one chip. So, there I can give a good product to the customer that basically is the idea.

So, if you see, this very fine, I can apply all the patterns. So, my job is done, but now think that this is a simple NAND gate.

(Refer Slide Time: 13:48)



So, if with come see that the number of inputs are say only 2. So, 2 to the power 4 inputs are there. So, this may be the story which maybe around 30 1970s when small electronics circuits were made this technology could have been applied, but again as I told you testing can be as simple as this or it can be quite complicated.

(Refer Slide Time: 14:03)

Detailed tests for the NAND gate

- Digital Functionality
 - Verify input/output of Table 1
- Delay Test
 - 0 to 1: time taken by the gate to rise from 0 to 1.
 - $v_1=1, v_2=1$ changed to $v_1=1, v_2=0$; After this change in input, time taken by o_1 to change from 0 to 1.
 - $v_1=1, v_2=1$ changed to $v_1=0, v_2=1$; After this change in input, time taken by o_1 to change from 0 to 1.
 - $v_1=1, v_2=1$ changed to $v_1=0, v_2=0$; After this change in input, time taken by o_1 to change from 0 to 1.

So, one is functionality means the just the verify the inputs and outputs. Now there is something called which is coming into the picture is called the delay because now chips are operating in a very fast rate. So, I from if I apply 0 0 1 0, the output is 1, in the next time pulse I apply 1 1, then the output is 0, what is the time taken from for the output to change from 1 because now chips are of being operated at the Giga hertz level.

So, around tens of gigs or at least 4 3 or in the Giga hertz paradigm I have to be very sure that what is the time required for the output to change from 1 to 0 or 0 to 1. So, in this case there is something called delay test. So, along with functionality I have to find what is the time taken for the signals to change from 0 to 1 and 1 to 0.

(Refer Slide Time: 14:48)

Detailed tests for the NAND gate

- 1 to 0: time taken by the gate to fall from 1 to 0.
 - $v_1=0, v_2=0$ changed to $v_1=1, v_2=1$; After this change in input, time taken by o_1 to change from 1 to 0.
 - $v_1=1, v_2=0$ changed to $v_1=1, v_2=1$; After this change in input, time taken by o_1 to change from 1 to 0.
 - $v_1=0, v_2=1$ changed to $v_1=1, v_2=1$; After this change in input, time taken by o_1 to change from 1 to 0.
- Fan-out capability:
 - Number of gates connected at o_1 which can be driven by the NAND gate.

Again there is something called I am not going into the details because basically, they are all parts of cad for VLSI, I am just going to give you the idea in brief, there is something also call a fan out capability that one gate can drive how many circuits, if you are using to many gates to drive, then what happens basically the chips become the gate the gate becomes slower and you have put buffers.

(Refer Slide Time: 15:11)

Detailed tests for the NAND gate

- Power consumption of the gate
 - Static power: measurement of power when the output of the gate is not switching.
 - Dynamic power: measurement of power when the output of the gate switches from 0 to 1 and from 1 to 0.
- Threshold Level
 - Minimum voltage at input considered at logic 1
 - Maximum voltage at input considered at logic 0
 - Voltage at output for logic 1
 - Voltage at output for logic 0
- Test at extreme conditions
 - Performing the tests at temperatures (Low and High Extremes) as claimed in the specification document.

Tests are for the "logic level" of the NAND gate.

So, that is also one point of test that what is the drive capability of the gate. Now all chips actually one to work at low power because all devices are mostly handled. So, what

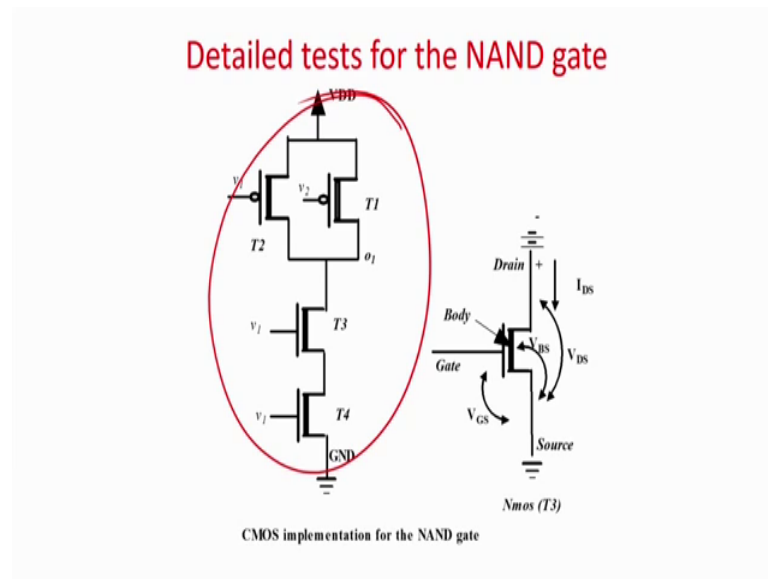
is the power consumption of the gate? So, it also test and find out that due to switching static currents or basically when the chip is not operating there will some drainage. So, actually what is the power requirement of the NAND gate that is also is to be tested similarly there other extreme test like threshold level that we know that there is something logic 0 logic one.

But what up to what voltage level it is treating as logic 0 and upper beyond what level this being treated as logic one like when we say 0 volts 2.3 or 0.4 0 volts should be say treated as logic 0 maybe if is a 1.8 volt technology, maybe 1.5 and above should be treated as logic one. So, that is also has to be verified finally, extreme condition test that if there is a if the temperature is very high the temperature is very low whether there will be able to do all the tests on it.

Now, think if you want to do it for 100s and 1000s of gates in a circuit, how much time it will require and what will be the cost. So, practically nobody has been ever attempted attempt to do all these tests for a circuit because it will takes 100s and 1000s of years to do it and it will be infeasible to do it in a practical amount of time. So, basically what we do, we are basically we generally go for logic level test of the AND gate and sometimes also at I mean as we will see down the line sometimes, we also go for this extreme test because nowadays we are operating in very higher frequency.

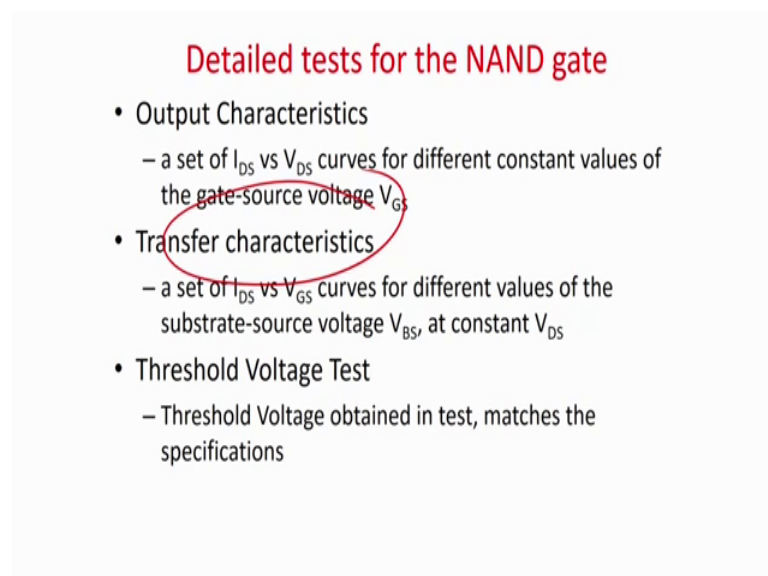
But an also power consumption to a certain level, but generally huge list of test I have shown you basically are infeasible and in practical.

(Refer Slide Time: 16:47)



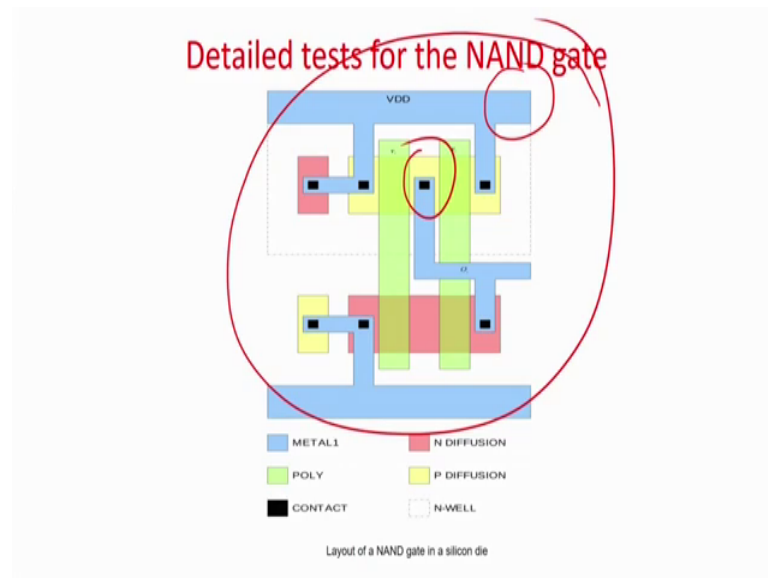
Now I am not even limited that now that was at the gate level, but all the and gates basically are developed using a such a transistor configuration in the CMOS logic. Now if I say that I want actually go for the transistor level test that can also be possible for all the NAND gates not only at the gate level I want see whether if and only if all the transistors are operating properly or not.

(Refer Slide Time: 17:07)



Then you have to see the output characteristics of the transistors transfer characteristic of the transistors threshold voltages of the transistors is the huge these sub test for the transistors.

(Refer Slide Time: 17:18)



So, even I can do all those test and not only at that this is actually the final layout of the circuit, you can even start looking at the layout from the physics point of view that whether this line this layout or this width of the copper implant is fine or not whether the vias are fine or not. So, even you can use a microscope or there is something called a there and basically you can even open up the chip and you can look it through a high powerful microscope and see whether all the part of the layout has been done properly or not.

So, basically what I mean to say is that testing in case of VLSI can be as simple as testing the functionality of a gate to as extreme as looking of the lines of the fabricated IC. So, now, you can understand for a simple NAND gate it can become so complex. Now we are living in a world of system on chips and network on chips. So, what will be the optimization required and what all you have to abstract and what level you have to go?

So, this module will basically try to tell you that; what is a practical level possible to do a test within a particular amount of time? So, that it becomes possible; profitable, if you

ask me, what is the golden rule, I will tell you even if you have a cruise control IC where you have millions of NAND gates, nor gates or gates of that order of size you go and test all the gates at the layout level using a microscope, but the time will be infinite nobody will be practically able to do.

So, you have to take a reasonable time, but at the same time as I told you in case of circuits the yield is not 90 percent plus it is lower you cannot keep on shipping faulty parts of the customer so that your market will terribly go down. So, I have to make a very nice optimal plane where I have to decide that I do not supply bad chips to the customer at the same time, I do not go too much too much time on testing which will actually make it the infeasible time frame and infeasible cost Paradigm.

(Refer Slide Time: 19:09)

Optimal Quality of Test

- Given a digital logic gate, what tests are to be performed to assure an acceptable quality of product at reasonable price".
- Test for the NAND gate should be such that results are accurate (say 99% above) yet time for testing is low (less than a millisecond).
 - Table 1 for the NAND gate and at proper time
- DIGITAL TESTING is not testing digital circuits (comprised of logic gates).

DIGITAL TESTING is defined as testing a digital circuit to verify that it performs the specified logic functions and in proper time.

Handwritten red annotations: Circles around "Test for the NAND gate", "DIGITAL TESTING", and "proper time". A red arrow points from the definition to the word "Power" written in a circle.

So, basically what people say what are the basic paradigms or what is the basic what do I have basically required for test

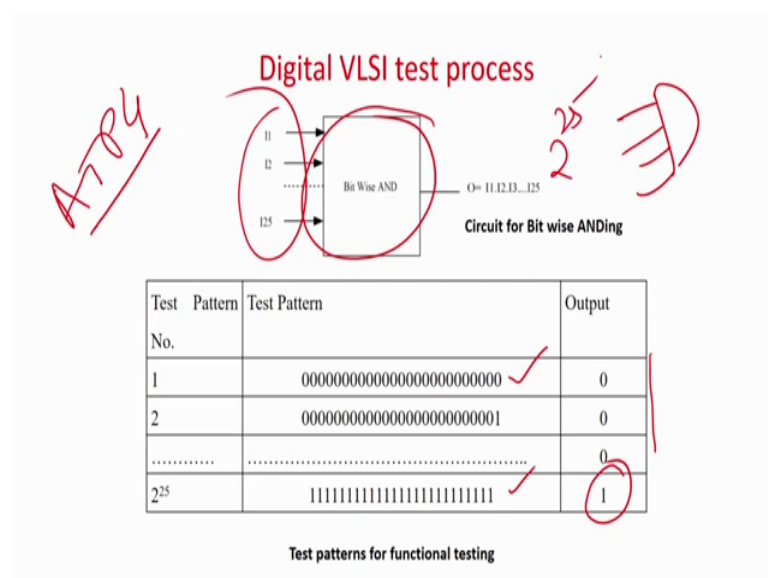
So, basically many people say that we require 99 percent plus fault coverage or guarantee will tell you what is the what a fault coverage I did about more than 99 percent guarantee that the chips will whatever I will send you as a proper chip, basically is a correct; that means, out of 100 chips, you manufactures 30 may have defect that is a yield problem, but basically when I am saying 99 percent accuracy; that means, if I am sending some 70 chips out of that it should be more than 99 percent or 90 percent 99

percent probability that whatever I am sending as the proper chip should not have any manufacturing defect that is what is the idea/

So, basically digital testing is not testing the digital circuits it is it is a something controversial statement basically digital testing is defined as testing a digital circuit to verify if that perform this specified logic operations and in proper time that is basically the definition because I can also tell that power functionality transistor level problems all these, I can add as the paradigms, but that will make it infeasible in the time frame to do a test. So, people say that digital testing means you take a circuit applied inputs at the logical level see whether the functions logical functions are made and of course, these speed should be within a proper time because nowadays we are very much concerned about the timing.

So, basically that is what I have to do power measurement, we will do it in a several different way that what actually put all the input patterns and check it there is a different way of measuring power. So, people think in that manner, but again, this is sometime I can also say power, but nothing more than that also partially I can have, but mainly I am concerned about logic function any proper time.

(Refer Slide Time: 20:49)



So, now I will start basically moving into the complexity and then I will see tell you what actually what up to what people have done in the VLSI level and what

optimizations will be required when you are going for the ultra large scale systems, say for example, I have a system and where I have 25 inputs basically is a AND gate 25 input AND gates, it is a AND gate with 25 inputs.

Now, how will you test it of course, you have to apply all 0s to all ones all the patterns you have to apply and see whether the output is proper or not. So, basically in all cases the output will be 0 except in the case with all one the outputs should be one very simple way of this is actually call automatic test patterns generation ATPG or test pattern generation, it is not automatic because I am doing it manually. So, all possible patterns have to apply and your job is done.

But now what is the number of patterns the number patterns even for such a circuit is 2 to the power 25. So, if you are that is you need to apply 2 to the power 25 patterns if so much huge number of patterns are applied per second using a megahertz tester it requires 30 second 33 seconds per chip.

(Refer Slide Time: 21:40)

Introduction

- Need to apply 2^{25} test patterns.
- 1000000 patterns per second (Mega Hz Tester) time required is 33 Seconds per chip.
- About million chips are to be tested in a run

33000000 Seconds or 550000 Hours or 22916 Days or 62 years.

Functional Testing cannot be performed due to extremely high testing time.

Now, as I told you in case of circuits all the chips has to be tested, it is not a software, it is an hardware. So, and also the yield is not very high. So, all the chips you have to apply so many number of patterns and you have to see whether the chip is operating properly or not, if it proper you ship it otherwise you throw it in the dustbin, but now basically if

you are applying 2 to the power 24 patterns, it is going to take a huge- extremely huge number of infeasible number of he has to do the testing.

So, therefore, this is actually called functional testing in which case you apply the patterns you know the functionality of the circuit and see whether the functionality matches. So, this functional testing cannot be performed due to the extremely high test time. So, one key optimization that was done in even in the case of VLSI era because in this lecture when I will be talking about GLSI you can think about basically a two processor level or slightly lower scale circuits.

But when I will be taking talking about SOC and NOC, you can think about the high scale computing platforms, which is the main emphasis of the course. So, functional testing was at all not even possible for the VLSI part. So, just of a lower scale interrogation people have thrown it up; now there is something actually called structural testing.

(Refer Slide Time: 22:56)

Structural Testing

- Structural testing, introduced by Eldred, verifies the correctness of the specific structure of the circuit in terms of gates and interconnects
- Structural Testing takes many fold less time compared Functional Testing yet maintaining the quality of test solution.
- Structural testing does not check the functionality of the entire circuit rather verifies if all the structural units (gates) are fault free. So structural testing is a kind of functional testing at unit (gate) level.

So, what is the structural testing in case of structural testing, I will not be concerned much about the whole I whole functionality of the circuit I will basically look at the structure and I have to find out whether everything is proper or not. So, basically structural testing was introduced by the gentleman. So, if says the verify the correctness

and the specific structure of the circuit in terms of gates and interconnects. So, basically what is the idea, I will not look at the entire circuit as the functionality.

So, it maybe a processor, it maybe a calculator or it may be a some kind of GCD calculator, it can be something like comparator. So, my idea is that I want to see whether all the gates which are the basic building blocks of the circuits and interconnects whether they are the properly functional or not, I will not see I will expect that if all the gates and interconnects are proper, then I will expect that the whole circuits should operate fine of course, but anyway in this case, we are not verify that if even if all the gates and the circuit lines are proper still they can be functional problem.

But that is a very very rare case. So, what idea is that they say that all the gates and the wires are individually tested and inter phase we are not much bothered about whether the inter phasing is happening properly or not all the components like gates and interconnect lines are tested and we are integrating it and then we are expecting like it should operate fine and truly speaking statically, it has been found out over last 20 to 30 years that more or less, it is a very good strategy.

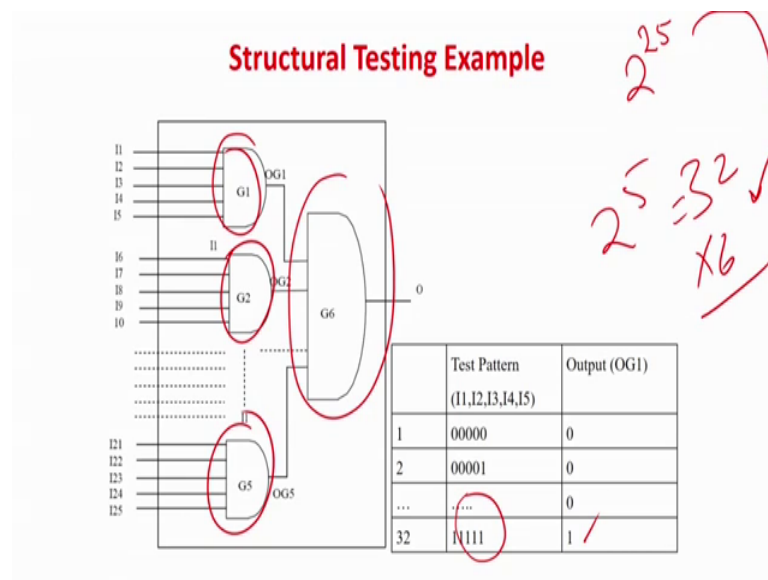
Because anyway functional testing is out of scope and you will find that functional tes[t]-structural testing takes very less amount of time, cons compared to functional testing. Structural testing is a much faster process of doing it, but only what I compromise here is that I do not actually test the basic in I means the inter phase and over all functionality.

So, basically structural testing is the kind of functional testing at the unit level that is only at the gate and interconnect level, whole level I am not going to do. So, one key point I am going to tell or thumb rule whenever you are going for optimisation in cad for VLSI, there optimisation at the level of the algorithms or time to generate the circuit or do the testing.

So, when will be compromising; we will be compromising mainly at the quality of the basically little bit compromising; obviously, do not compromise much. So, compromise with the quality of test quality of design optimisation or the area and power requirement of the circuits that is the quality of service expected will be slightly compromised, but the algorithms will be very fast so that you can get the output in a reasonable amount of time.

So, optimisation is not in the performance of the circuit. So, this you can take is the thumb rule kind of a thing, not a very regular one, but mostly thumb rule when you are talking about optimisation it is mainly at the cad part of it or the algorithms to design the circuit or test part of it. So, there even if it is the very big system I can do my job very quickly. So, that, but there will very minimal quality compromise in the device or the circuit like whether I am going from structural testing to functional testing, I am not testing the functionality and also I am I am not very explicitly or rigorously testing the interface I will take an example.

(Refer Slide Time: 25:58)



So, maybe this is my circuit, there is 25 inputs and 1 output. So, last case, what I have done? I have given 2 to the power 25 inputs and I have done the functional test. So, once the chip is functionally tested, I am very happy that everything will be properly done, but now what is structural testing because anyway I cannot applied that pattern. So, what I will do here is I will take test all this all I will test all the individual gates at point of time. So, all the gates have how many inputs? They have 5 inputs. So, 2 to the power 5 that is equal to 32 inputs will be applied to the gates all 0s to all 1s and gates.

So, basically on the, for the all ones the output will be 1 and so forth. So, I will test all the individual gates at the level. So, 32 into 6, there will be some 100 patterns will be required compared to 2 to the power 5, 2 to the power from 2 to the power 25; I have

come to the value of 32 into 6. So, huge jump is going to happen use this something like divide and conquer policy.

So, unless from 2 to the power 25, you are actually break breaking up into 6 into 2 to the power 5. So, that way, but again I am just going to check the gates and I will check the interconnects, but now if I inter phase everything I am not going to test that part, but the gain I am achieving is from 2 to the power 25 which is infeasible to 32 into 6 patterns.

So, that is what is the optimisation for the testing, but slight compromising in the quality because statically, it has been found out that even if all the gates are interconnects are separately operating fine interfacing circuits are when even the circuit is interfaced together the circuit operate or so, in a very proper fashion.

(Refer Slide Time: 27:27)

Structural Testing Example

- Number of test patterns required are $6.2^5 (=160)$, which is many fold smaller than those required for functional testing (2^{25}).
- Time required for testing the circuit the using a 1 Mega Hz Tester is 0.000016 seconds and for a million samples is 16 seconds.

Structural testing is highly beneficial over functional testing.

So, this; what is given. So, from this is basically 160 inputs are required which is much much less than 2 to the power 25 and you just required such a less amount of time to test a circuit.

So, very easily you can test 100s and 1000s of circuits using this structural test.

(Refer Slide Time: 27:43)

Structural Testing—Penalties

- Each individual gate is tested; however, the integration is not tested.
- To test the individual gates, controlling and observing values of intermediary nets in a circuit becomes mandatory, which adds to extra pins and hardware
- Circuit with about a million internal lines needs a million 2-1 Multiplexers and same number of extra pins. **This requirement is infeasible.**

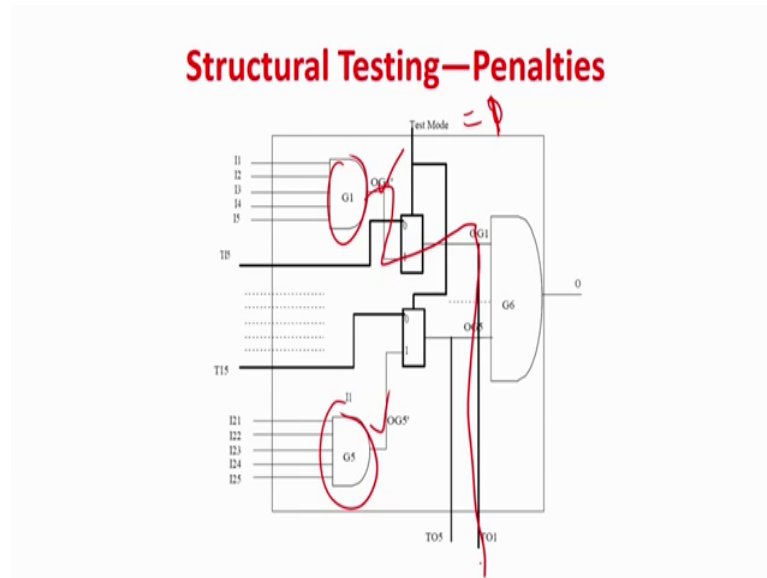
Now what are the penalties each individual gate is tested, but integration is not tested. So, actually another very important thing is that which I will show you a figure and it will be clear these two points then I am telling of course, you are going to test the circuits.

So, but this is a chip this is a black box, you have seen that all the VLSI chips. So, there is a plastic cover and after that there are some pins which are coming out, but if I am applying the inputs at this I need to see what is the output of this pin, but again this pin is inside a circuit. So, how I will observe this pin, similarly this is I have observed very good, but if I want to apply some inputs to the gates, I have apply this inputs directly over there because if I try to controlled this inputs via this NAND gates a in the problem will be blow up. So, I need a direct access to the line to for read and write basically, but how I do it because this is a black box.

So, that is one big problem of structural test that is to test individual gates controlling an observability is very much required because is a circuit only the input and output pins are there, but how do you probe the internal speed. So, there millions of internal lines in a circuit what you can do you bring out everything, it is an impossible, it is a very another more impossible solution because your chip will has some 100s and 1000s of pin outs which is not possible.

Now, what is with the very very complex processor also the number of input output pins are maybe 1024 or it will lower than that that actually will also a ball bead array package. So, this requirement is infeasible. So, how to do that?

(Refer Slide Time: 29:12)



So, this is basically a picture and how to do that even if you are having a pin out. So, if you can see they have put some multiplexing arrangement and these are the pin outs, they have brought. In fact, they can also this also you have to basically I am I am just going to just give the gist because you can easily find that in any test book. So, this is actually a multiplexer.

So, what happens basically? So, whenever you want to sorry. So, whenever you want apply something directly to this gate, you will actually cut this out to this multiplexer and directly give the input from the outside similarly. So, you have to again give some inputs to the gate also, then I have to actually cut this out and I am having a multiplexer which is having 2 mode, 1 is called the normal mode and the test mode the test mode basically the input to this from G 5 to G 6 is cut and there is an external pin which will directly give input to this right and then basically these actually solve the problem of directly applying something to G 6, but when the I will be operating in the normal mode you make the test mode equal to 0.

So, sorry the test mode equal to 1, so, in this case, external lines will be cut and the output of this gate will be fetch to this gate and your job is done that is you can apply some patterns directly to this inputs again when I as I already told you that I I have to also explicitly observed this lines when I am have to test this 2 gates, but how can we observe again you can change the configuration mode the outputs from this can be observed this line.

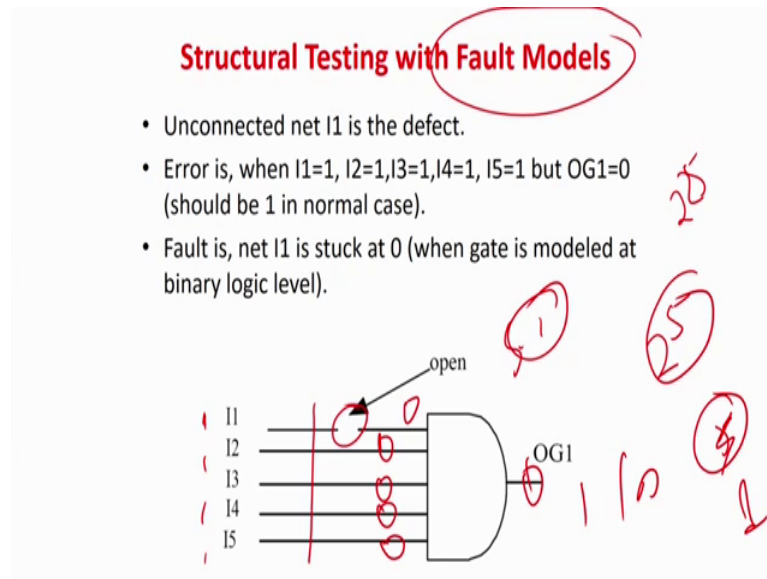
So, by actually having some multiplexing arrangements, we can also go for observation of the internal lines because as I told you cannot bring probing out, sorry, you cannot have infinite number of probings and control, but still you can see I require many probe bounds, there is I have an extra pin for these 2 lines have to observe. So, I have to bring two exiles of the chip and again two lines for the inputs of this gate.

So, basically two line means there are the 5 gates which I have not shown basically. In fact, there will be 5 lines, we shall have to take care of output, they will be 5, again 5 inputs for this control pin which will be again taking as the output. So, some extra that is in the G stuff is that you can go for structural testing, but the problem here is that you can very easily go for structural testing because the number is this everything is very feasible.

But I have to sometimes observe this lines directly and sometimes I have to also control this line directly to test these gates at the individual level, how to do this you have to actually make some pin outs for all these parts and also you have to some multiplexing arrangements so that I can apply directly something to this. So, you can understand a huge number of extra multiplexers will be required and huge number of pin outs has to be made corresponding to all the points which are internal to the circuit chip which have to observe by control.

So, structural testing is a good idea, but I cannot make out so many number of pins.

(Refer Slide Time: 31:56)



So, now how to solve this problem there is something then came up which is called a fault model. So, these are the very interesting discovery.

But if you ask me that how it is mathematically correlated, it is very difficult to answer at this point of time, but it is statically found out that it is a very successful model. So, what you mean by statistical testing very simply let me tell you that initially we told that how do a test that we go for functional testing, then you say the functional testing is very difficult 2^n to the power n complexity, etcetera then what we do? We go for structural testing.

But in structural testing only, you are testing the gates and mainly the inter connects, then if I integrate how do guarantee the functionality is proper, it is founds statistically that if you go for functional testing sorry, if you go for structural testing then for 99.9 percent plus accuracy you can predict that the structural test is if the structural test is proper the functional test is also proper.

So, with that correlation from statistical finding of testing of physical chips in the lab, we have found out the structural testing takes lower time at the same time the correlation is the very very nice with the functional test. So, that is how it happens, but again the structural testing is having a big problem, what is the problem we have seen? The problem is that you have bring lot of pin outs to see the internal pins of the circuit and

also control the internal lines of the circuit that is the big problem that is happen there, something came with something called a fault model.

So, what is the fault model? In fault model, they say that we are not even concerned with it functional way of the gates within first we are not concerned with this first, we are concerned with the functionality of the circuit, then we went to structural testing where we are not concerned about the functionality of the circuits rather we are concerned only about the function of the gates.

But now in the third level which is in fact, the de facto standard which is now used everywhere is the fault model, in this case, we are not even concerned about the functionality of the gates level whether what we do we say that is the fault model. So, what is the fault model fault model means they assume that there will be certain kind of faults in the circuits which by statistics, they have found out is the most likely to be occurring and then you have to verify that those faults are not there. So, we are not even concerned about functionality of the gates, but what we assume that there can be certain faults which are very predominant and you have to verify that those faults are not there if those faults are not there, we will assumed that the circuits operating finally, proper or not again this is an optimization to save time, but slight compromise in the quality of test.

But again as I told you by statistics people have found out that if you are having a proper fault model the accuracy with functional testing is 99.9 plus. So, again reiterating optimization is all happening in the terms of test time test pattern generation algorithms, etcetera, but slightly compromise with the quality of test.

So, all the optimizations are in the terms of algorithms. So, again one it is very simple fault model is called a open fault model; that means, whatever lines you have fabricated that line will be basically open because of some kind of less amount of doping or some kind of issues with the fabrication technology some lines will be opened. So, this is a open fault. So, what you have to verify that none of the lines are basically open. So, how what you can do simply you can apply all 1 1 1 1 1 and you have to wait a 1 over here, of course, if you are if you apply all once and you get a 1 over here no line is open.

But in fact, you see I am not even verifying the functionality of the AND gate. So, what I am just verifying that no line is open because if the line would have been open, this would have been 0 and I would get the value 0, I would say there is some open faults in the circuit as simple as that. So, basically open is also called stuck at 0 and I will come to that what is the name of fault model this is a fault model.

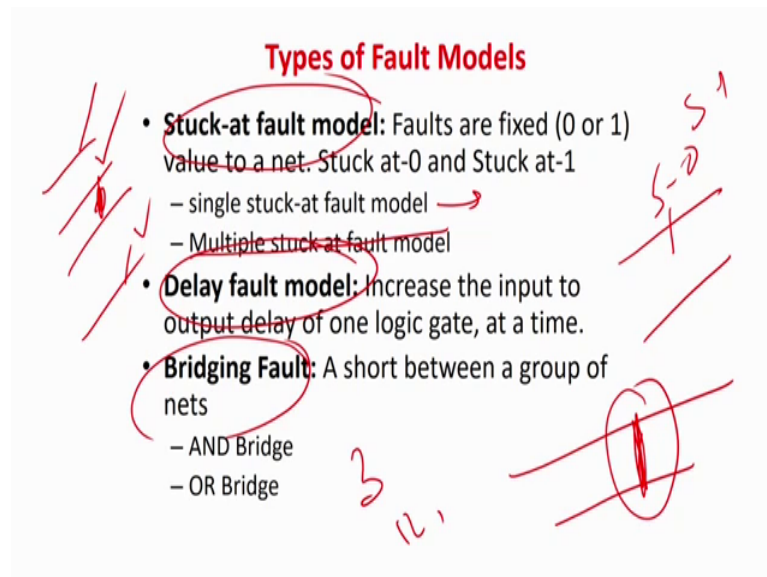
So, typically say that these are some faults, this has to be tested and do not even bothered the functionality of the gates. So, now, now in the previous case, for the 5 input gate we have to apply 2 to the power 5 inputs, but now in this case, I will only apply may be 5 inputs from 2 to the power 25, I have come 2 to the power 5. Now I am coming to 5. So, what are the 5, 5 or in fact, you can call 6, then we will assume that all these lines may have stuck open faults that is the lines are cut.

So, you have to just apply 5 patterns, if in fact, I am sorry, you have to just apply one pattern i[f]- there is 1 1 1 1 1 and the output is 1, then you can be verified that none of the line is stuck open that none of the line is cut.

So, you are fine so. In fact, from 2 to the power 25 and coming to 2 to the power 5 and in fact, I am coming to one pattern, one pattern actually test all these stuck open faults, in fact I mean you assume that I am applying 1 1 1 1; 5 time, but it actually not required I am applying in means mathematically speaking all the ones are have been applied for each individual faults like this can be open this can be open this can be open this can be open and so forth, but in fact technically only one all ones if you apply you can verify that none of the line is stuck open, then your job is done. So, with one pattern, I can verify the there is no stuck open faults.

Now, from 2 to the power 25, I have composed single pattern to do the test huge save in what do I say huge save in the number of test patterns and time, but where I where I am loosing, I am not now even testing the quality of that is the functionality of the circuit, I am also not testing the functionality of the gates even just I am finding out that there is no stuck open fault because somebody told me from statistics findings that such type of stuck is mainly are having stuck open faults just you verify, there is no stuck open faults, then you can easily shift the chips in the circuit and the accuracy is 99.9 percent plus correlation with the functional test.

(Refer Slide Time: 37:18)



So, on that belief only, I am doing it. So, these actually called fault model. So, now, what are the basic type of fault models which are in practice in fact, this what I have said is this stuck open is the very straight example, but there are some real life fault models which people have found out statistically and now are been applied and this one.

So, that is the stuck at fault model which is very very popular. So, it is says that none of the line should be stuck at 0 or stuck at 1 that is means no line should be permanently 0, no line should be permanently 1, in fact whatever logic I am giving to a line, it should be ported to the other part of the circuit, it was found statistically till last 10 years or 15 years that if a circuit do not have stuck at faults the circuit is 99 percent probable to be working as per the functional its functional specification so that much guarantee was given.

So, stuck at fault model even the very powerful fault model and in fact even it is also the defacto standard and everybody starts with the stuck at faults, then there is something called a single stuck at fault model which was the most popular one that is at a time you assume that only one line is having a fault. First, you state this line is stuck at fault or not, then you go for another line and so forth, you assume at a time, only 1 line will be stuck at fault or not, then there is something called multiple stuck at fault model because in that case; any 1, 2, 3, any number of lines can have stuck at faults.

So, you have to take all individual lines then you have to take pairs and then take 3 lines and then quads and so forth, these slightly complex model, nobody even actually applied this people have found, surprisingly, it is a boon where if you stuck; verify the there is no structural stuck at faults in the circuit 99.9 percent provable that it is the functionally proper because multiple stuck at fault also the fault numbers are high, then as I told you nowadays, we are also operating at higher levels of speed. So, there is something called delay fault model that you have to find out whether the input output the delay is maintain and there is something also called the bridging fault model the bridging fault model is that you have to verify that do no two lines are very short.

So, more or less these fault models are still use in even in very very advance circuits. So, stuck at fault model was applied in the traditional level. Now, what is delay faults and bridging faults are also been incorporated. So, what they verify that no line should have a stuck at 0 or stuck at 1. Similarly, the delay you measure that if I am giving 1 or 0, the output and input should take place within an given amount of time and also to verify that no two lines are stuck and very one important point, I want to emphasize people go for single stuck at model or fault model.

So, what is the single stuck at fault model single stuck at fault model means at a time, you assumed that only one fault is there, it is tested not to be there, then you go for the next fault and so forth like in this case, I will see if there 10 lines, first I will see that first line is a stuck at 0 or not, then I will go for the second line and third line, I will not considered pairs and the triplets in this case because the numbers will be very high. Similarly for the bridging pair bridging fault, generally, we take two lines at a time and some modern cases they take 3 lines at a time which are in close proximity and then you take all pairs and you see whether there is a bridging fault or not; that means, you take if there are 3 lines say. So, I will take one two then I will take a all the combinations, I will take and whether I will see whether one at a time say for example, if I have say if I have say for example, 4 lines; 1, 2, 3, 4 say.

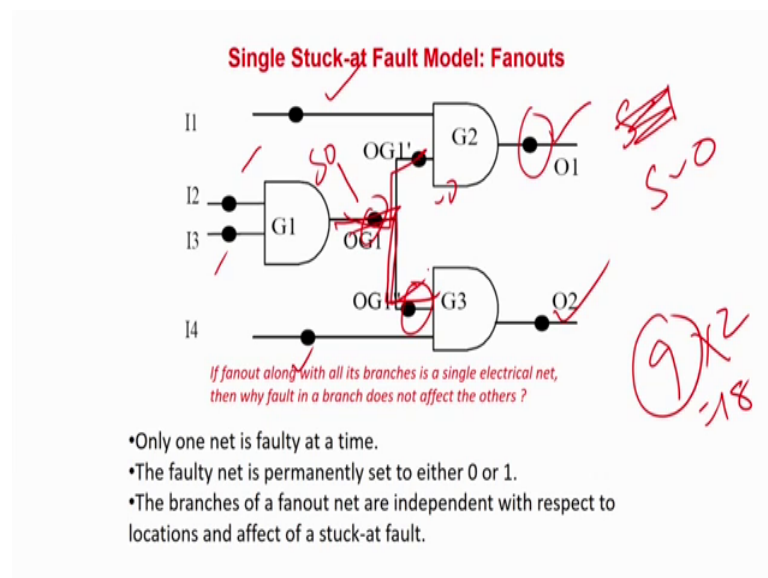
So, basically I will see whether these two lines are stuck at bridging fault or not, but I will not checks simultaneously that whether I will not try to simultaneously assume that these two bridging faults are there single stuck at fault model means, first I will see whether this is there or not if I verify this is not there, then I will go for this then maybe I will basically take the other combinations like may be after that I will take this

combination and so forth, I will drop this and this combination because they are quite far and short probability will be very less.

So, these how basically it goes. So, what is the idea of fault model basically you do not even test the functionality of the gate just verify the faults which are given in a list are not present in a circuit if you are able to do that; then your testing is done this is a way this was actually a path breaking optimization in the area of test. So, whenever we moved from low scale integration to and from medium scale integration to high level integration that is VLSI we all move to fault models nobody could do functional or structural test at that level because of the complexity.

So, fault model was a phenomenal discovery that statistically it is shown that if you verify on basis of a fault model mainly the stuck at fault model very higher probability that 99.9 percent that the circuit does not have any functional defects.

(Refer Slide Time: 41:40)



But there was a slightly some people have modified the fault model for a fanout, I am just showing you the stuck at fault model. So, for a circuit like this, they are actually you have to point out all the nets of the circuit and you have to verify that none of these lines have stuck at 0 or stuck at 1, 4 at a time, for example, if I want to test this, first we will assume there is a stuck at 1 fault here and apply certain inputs and see that is stuck at fault is not there, then we will erase it as take stuck at 0 fault, then we will apply a input

pattern as see it is stuck at 0 fault is not there, you will reiterate for all the lines of the circuits and then your testing is done.

So, in this case there is 1, 2, 3, 4, 5, 6, 7, 8, 9. So, 9 lines are there. So, 9 into 2 will be equal to 18. So, 18 faults will be there. So, you have to at max apply 18 test patterns, but actually it is shown in theory and with examples it can be as I will show you that we will not even require 18 patterns for one pattern more than multiple number of faults get detected.

So, may be only for 3 or 4 patterns will be required to testing. So, one bi-product that has been found out for fault test at fault model is that if there is say 10 faults if there n lines of course, there two n faults two into n because stuck at 0 and stuck at 1, but 2 n number of patterns are not required the patterns are much lower than that because one fault pattern can test multiple faults basically.

So, in this case, we will not require 18 will require much less than that, but one important point, I just want to emphasize that this is a fanout branch from here, I am going to all these parts 3 lines I am going, but I do not considered this as a electrically even line same line what I do is that I have I can have faults here I can have faults here as well as I can fault have fault here.

So, we these a fault model. So, the fault model assumes that the fanout branches are independent if I actually if I will consider stuck at 0 fault here, I will assume this line to be fault free, but when I will be having a, but when I will of course, when I will be having a stuck at 0 fault here of course, this line whole line will be effected, but when I will having a stuck at fault at this zone at this line. So, this line will be only effected and the other branch of the fan out will be fault free.

So, this is basically the stuck at fault model which is the defacto standard and was applied these are very long time. So, in this preliminary lecture I will be mainly covering the stuck at faults and then when I will be going for advance lectures maybe will try to see something about bridging faults and delay faults that will try to give you a coverage.

(Refer Slide Time: 44:01)

Single Stuck-at Fault Model

- Several stuck-at faults can be simultaneously present in the circuit.
- A circuit with n lines can have $3^n - 1$ possible stuck line combinations; each net can be: s-a-1, s-a-0, or fault-free.
- Handling multiple stuck-at faults in a typical circuit with some hundreds of thousands of nets is infeasible.

Single stuck-at fault model is manageable in number and also provides acceptable quality of test solution, it is the most accepted fault model.

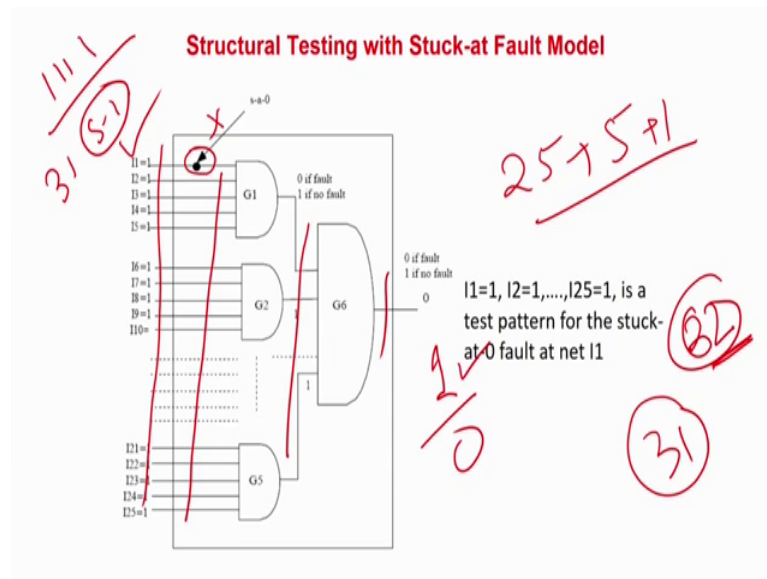
2^n

So, as I told you. So, if you have, then what is the question of single stuck at and multiple stuck at faults?

So, single stuck at fault means as I told you if there are n lines the two n number of faults can be there stuck at 0 stuck at 1, but why people will have not considered multiple stuck at faults because if there n lines, you want to consider multiple stuck at faults then the line can be either stuck at 0 or stuck at 1 and fault free. So, 3 to the power n minus number 1 minus 1 is for all normal case can be having a faults. So, 2 lines, 3 lines, 4 lines, all possible combination, if you want to take the number is very high and people have found out that it will take more time to go for multiple stuck at faults, but the accuracy is only slightly improve considering a single stuck at fault.

So, by statistically people have found out that single stuck at fault is manageable in number provides acceptable quality of solution and it is that is why it is the most acceptable fault model that people has found out by real experimentation.

(Refer Slide Time: 44:56)



So, that is basically de facto standard now again I am going to same take same circuit for example, and I assumed that there is a stuck at 0, fault at this line basically that is the idea, then what I have to do in this case, there are 25 lines, these 25 plus 5 internal lines is 5 plus 1 that is 25 30 31. So, there are 31 lines here. So, worst case there will be 62 number of faults and 62 number of patterns may be in the worst case require to test it.

But again what are the benefits, I will show you that you do not require any number of pin outs from the internal branches that is that is how if the most gain of the stuck at fault model or any fault model that I do not require to probe rather I require to save anything as an input, I will just use the input and output lines to do the testing that is what will be the most amount of gain compared to simple structural testing, but along with that we will be very happy to see that one pattern will detect multiple stuck at faults. So, even I do not require 62 number of test pattern that number of patterns will be very very low.

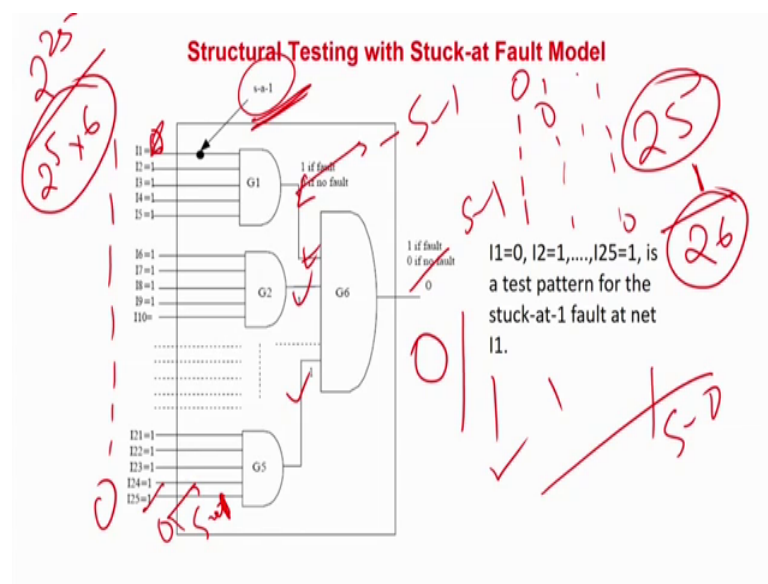
So, let us take assume that this line is having a stuck at 0, I have to verify that none of the line has stuck at 0 and also have to verify that none of the line has stuck at 1. So, stuck at 0 here, say I apply the pattern all once. So, what is the case the output will be a correct. So, now, if this line is stuck at 0, what I am going to get the answer I am going to get the answer as 0. So, by applying all once, I can verify that these line does not have a stuck at 0, similarly, it is very easy to observe that all once and the output is also a one basically guarantee is that none of these lines has a stuck at 0. So, all one pattern basically verifies

this 25 lines plus 5; 30 and 31 lines these all one patterns basically verifies 31 stuck at 1 faults is not present.

So, one pattern that is all once verifies that none of the line is having a stuck at 0 that is one pattern 31 faults gone. So, that is the beauty of structural testing and also if you see I do not have to prove anything in between what I am doing is I am applying the inputs and just I am looking at the sorry I am looking at the output and my 31 faults are gone or miss cover that is actually cover fault coverage is all once and able to test 31 faults again as I told you, you have to verify also that none of the lines has stuck at 0.

So, I am using a stuck at assuming that these was a stuck at 0 fault. Now I am assuming a stuck at 1 fault over here. So, now, what is the pattern? So, of course, I am app say I am applying a 0, it is very obvious that if the line has a stuck at 0 fault you have to apply a 1 and these line and vice versa.

(Refer Slide Time: 47:16)



So, is a stuck at 1. So, I am applying a 0 over here and all lines I have to keep it 1 to keep the controllability of the n bit. So, if the circuit is normal, I will get a value of 0 because I am applying a 0 at the input, but if this line is stuck at 1 then what is going to happen you are going to get the answer as 1.

So, if the answer is one you know that this line is having a stuck at 1 fault else the circuit is normal. So, this pattern actually tests stuck at 1 over here and also you can easily

verify that it will also test a stuck at 1 over here because even if this line is not stuck at 1, but this stuck line is stuck at 1, you can have a fault like this and of course, this line stuck at 1 we are going to get it. So, one pattern in this case is testing 3 faults again if I assume that say this line is having a stuck at 0, this line is having a stuck at 0 then what I have to do I have to apply 1 1 1 1 1, sorry, I am saying was stuck at 1.

So, if I have assume that the another line is stuck at 1. So, you have to apply all once in all the inputs accepting for this line and the same philosophy will fall. So, how many patterns will be required, as you can see, you will be requiring 25 patterns because each pattern will verify the input line does not have a stuck at 1 fault. So, to add along with each pattern not only it is the input faults, but also will test this fault as I have shown you and also the stuck at 1 fault at the output.

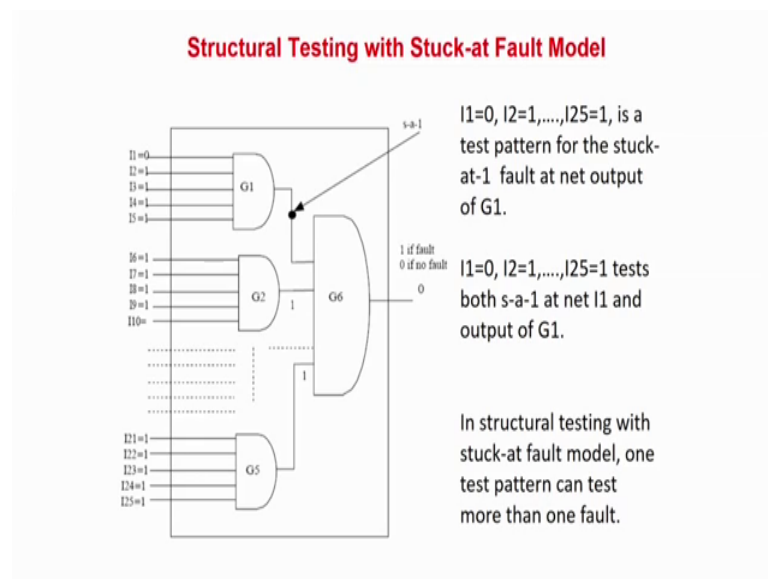
So, 25 plus 1. So, 26 faults are patterns are actually solving the problem for you and you have to understand that we do not at all required to any probing inside the chip that is what is the foremost advantage of stuck at fault model structural fault testing was very good, but the problem is that we have to actually probe out the value. So, now, what we have got from 2 to the power 25. So, we have got into 2 to the power 5 into 6, still this is ok.

But structural testing means you have do lot of probing not at all feasible then we have come for technology which is called test with the fault model were you are only using 26 patterns to test the faults because here, I have shown you all 1s will verify that none of the nets has a stuck at 0 fault and then with a test patterns of working 0s like 0 1 1 1 1, then 0, sorry 1 0 0 0 0 sorry means that is a series of working 0s like 0 1 1 1 1 in 1 0 1 0 1 1 1 1 dot dot dot finally, again 1 1 1 1 0.

So, if you have apply this 25 patterns all the stuck at 1 faults will be covered is very obvious a in the last case one pattern was covering 25 1 pattern was covering 25, 30 plus 31 faults, but in this case actually the number of faults covered per pattern is for the stuck at 1 is less, it is only covering 3 patterns, 3 faults, 1 pattern is covering 3 fault, but any way. So, this shows that structural testing is actually brings down the number of test patterns to a huge level not only that not only that it also basically has no requirement of any probing in the circuit.

So, this is actually called structural testing with fault model. So, in case of all VLSI circuits even with the recent once people have the basic fault model as the people do testing on the basis of a fault model because nobody can go beyond this because that is an optimized level in terms of test time, but again as I told you, what is the compromise the compromise is even not testing the functionality of even the gates, but statistically found the correlation is very good with functional test.

(Refer Slide Time: 50:43)



So, this is I am just reviewing the feature. So, 1 1 fault model one pattern. So, all the good things about the structural and fault model is test over here.

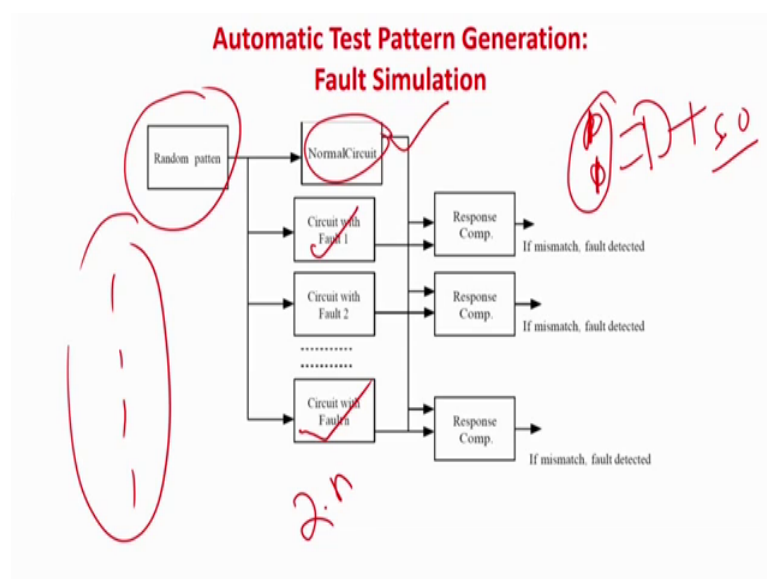
(Refer Slide Time: 50:48)

Pros and cons for structural testing with stuck-at fault model

- Pros
 - No extra pin outs or DFT circuitry like 2-1 Multiplexers and shift registers for controlling and observing internal nets
 - Low test time as one test pattern can test multiple stuck-at faults
- Cons
 - Functionality is not tested, even for the units (gates and Flip-flops). However, testing history reveals that even with this price paid, quality of test solution is maintained.

So, pros no DFT circuit; DFT mean design for testing circuit like multiplexers which are additionally, you have to put in the circuits low test time because one test pattern is covering multiple faults, the only disadvantage is that functionality is not tested even for the gates and fops, but history says that there is a huge correlation. So, not much is compromised, but the optimization is at the level of algorithms or techniques to do test faster, but slight compromise is on the quality of test.

(Refer Slide Time: 51:19)



Now, so, this was a simple circuit. So, I have give the obtain the test patterns manually now the question comes if you have a complex circuit how do generate the test patterns of course, given a circuit I cannot do it manually. So, this is actually call automatic test pattern generation or ATPG algorithms which is the heart of test. So, given circuit and a fault model you have to find out what is the minimum number of test patterns required to do the testing. So, in VLSI era our target of coverage was 100 percent that given a fault model you have to cover all faults that is what about the a de facto standard. So, whatever you do you try to receive 100 percent of coverage.

So, you require some 100s of test pattern because I have shown you that is not very high for the 25 input circuit, you require only around say 26 circuits to 26 in[put]- patterns to do it. So, if it for a large processor also the number of test patterns by this technology was not very high, but now when you are going to NOC and SOC era, the circuits also becoming so large and so complex that even in this level of abstraction or optimization that we are not even testing the functionality of the circuits of the gates and nets, we are just going for a fault model like a stuck at fault model still the circuits are. So, large that even 2 into any. So, large or the number of faults even in a by the stuck at fault model is. So, large in NOC and SOC systems that still these things will not be that much applicable.

So, when we be looking at optimization of ATPG from the context of SOC and NOC or or very large scale circuits we will discuss all those paradigms. Now basically we are looking at how basically automatic test pattern generation actually has solved say over the typical medium scale or certain large scale circuits not to the level of systems. So, one was actually called the fault stimulation. So, it is a very simple idea which I have already done, but we will have to develop software codes to do it automatic. So, you take a normal circuits then you take circuit with fault 1, 2, 3, 4 and so forth faults means this stuck at fault. So, if there a n nets in the worst case you will have two n such circuits then you take a random pattern you can assume that all once is a random pattern.

Some random pattern you take then you stimulate this normal circuits with all ones and all the faulty circuit is all ones then you will find out that there is a miss match in the result say may be normal circuits fault one fault two up to fault all these a miss match; that means, this one pattern is actually covering all these ones because if you apply ones all ones the normal circuit and the faulty circuit will have a difference.

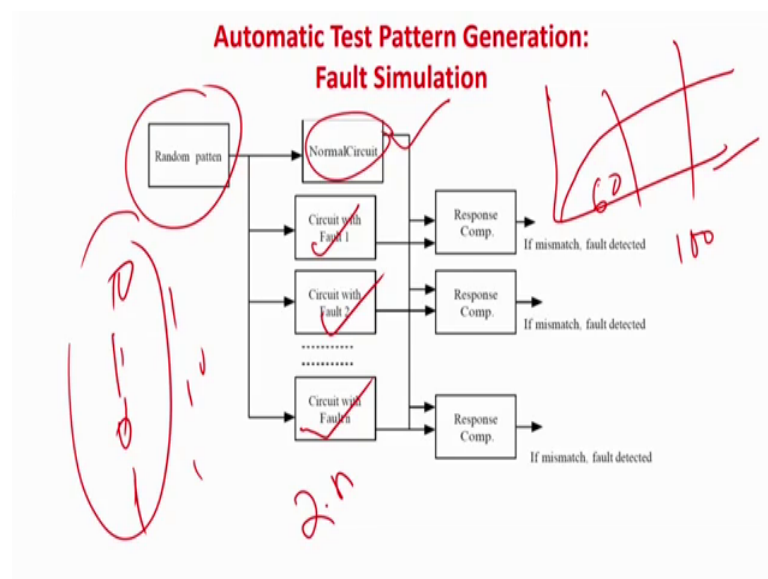
But say for examples with this one only circuit fault circuits with fault one and fault n are different in results correspond to the normal circuit; that means, this all one patterns actually can differentiate fault one and fault two correspond to them again like for example, if is AND gate and you are having a stuck at 0 fault, then if I apply 0 0 over here then the output line is having stuck at 0.

So, of course, with this input of 0 0 both the normal AND gate and the AND gate is stuck at 0 fault as the output have similar outputs so; that means, this fault that time is not able to cover this fault, but if I apply 1 1 in the normal case the answer will be one and the faulty gate answer will be zero. So, we say that the pattern 1 1 detects this fault or covers this fault.

So, what I do I take some random patterns and I apply to all the circuits normal circuit and the failure circuit where they were is the response miss match; that means, that pattern covers that fault and we eliminate it then again we will say another random pattern will generate say 1 1 0 1, then again I will repeat it maybe, then I find that circuit with fault to has a miss match I will keep on doing it till all the faults get detected, but in fact that does not happen.

We start with a random pattern random pattern say then we find the coverage goes like a curve like this say maybe after have after appearing about say say 100.

(Refer Slide Time: 54:45)



Patterns typical number I am saying we will find out that till about say 60 or 70 patterns new patterns were apply more number of faults are getting detected, but after that this curve will saturated; that means, now what happens even if we are applying new or newer patterns some of the faults are not going to get tested; that means, they are called difficult to test fault some faults are called easy to test faults where what happens basically you take some random pattern, then apply it and we will find out that the faults are getting detected mainly such faults are stuck at faults at the input lines or the output lines of the circuit.

But some faults will be quite difficult which will be say in the middle part of the circuit, etcetera, in such cases if you are applying random patterns you have to apply a huge number of random patterns only a very few of them will be able to detect those faults. So, we call it difficult to test fault. So, idea is that first we go for fault stimulation base automatic test pattern generation you test take all the faults apply some random patterns maybe 100s of them and then see which faults get detected whichever fault will remain we do not, then go for basically random testing in that case it will take a longer time to do it we mark them as difficult to test faults and then basically we go for path sensitized base ATPG.

(Refer Slide Time: 55:52)

Path Sensitization Based ATPG

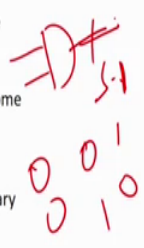
ATPG by path sensitization method is generally applied for "difficult to test faults" and comprises three phases.

Fault sensitization: In this step a stuck-at fault is activated by setting the signal driving the faulty net to an opposite value from the fault value.

Fault propagation: In this step a path is selected from the fault site to some primary output, where the effect of the fault can be observed for its detection.

Line justification: In this step the signals in (internal) nets or some primary inputs, which were assigned for fault sensitization/propagation, are justified by setting (remaining) primary inputs of the circuit.

In the second and third steps, a conflict may occur, where a necessary signal assignment contradicts some previously-made assignment. When conflicts occur we need to take a new alternative path for fault propagation and see if all signals can be justified.



That is actually called deterministic ATPG; that means, we will take a particular fault sensitize it propagate the well through the output and do a line justification I will deal

with I will tell you with an example in this case in fact the idea is that; now it is not a random test basically what we are doing we are applying taking a fault and deterministically we will find out what pattern, we will test because say out of say there were 30 faults, I apply 20 test patterns randomly and say out of 30 28 faults get detected 2 faults remain.

So, detected two faults maybe I have to apply some another 30 random patterns to find whether this that whether when the fault will be detective because they are difficult to test faults and they will be may be detective by only a single test pattern and that pattern has to come in random fashion. So, the probability is less, but the easier fault means they will be covered by multiple number of faults multiple number of patterns say as I told you if you take a AND gate and if you have a stuck at 1 fault over here, then you can detect it by 0 0 input 0 1 input and 1 0 as a input all this will detect the stuck at 1 fault at the output of AND gate.

(Refer Slide Time: 57:04)

Path Sensitization Based ATPG

ATPG by path sensitization method is generally applied for "difficult to test faults" and comprises three phases.

Fault sensitization: In this step a stuck-at fault is activated by setting the signal driving the faulty net to an opposite value from the fault value.

Fault propagation: In this step a path is selected from the fault site to some primary output, where the effect of the fault can be observed for its detection.

Line justification: In this step the signals in (internal) nets or some primary inputs, which were assigned for fault sensitization/propagation, are justified by setting (remaining) primary inputs of the circuit.

In the second and third steps, a conflict may occur, where a necessary signal assignment contradicts some previously-made assignment. When conflicts occur we need to take a new alternative path for fault propagation and see if all signals can be justified.

So, it may be called as a easy to test fault, but if you are taking say I have a stuck at 1 stuck at 0 fault at the output there, if you have to apply only the pattern 1 1 to differentiate it from the normal circuit. So, in that case I can tell that the stuck at 0 fault at the output of the AND gate is a difficult to test fault. So, if a circuit if a fault has multiple test patterns, then if you are going for a random test, it is a highly probable that any one of this patterns will appear from the random set and the fault will be detected,

but is a difficult to test fault have a very limited test pattern then probability of those appearing will be lower. So, basically that you have to go for a fault sensitization propagation and justification based approach in which case you will particularly target that fault and test it.

(Refer Slide Time: 57:47)

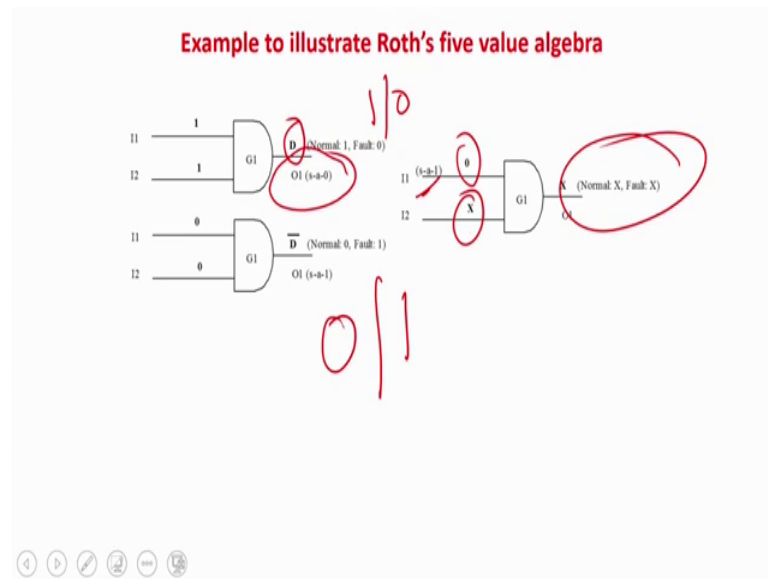
Roth's five value algebra

Symbol	Implication	Normal Circuit	Faulty Circuit
0	(0/0)	0	0
1	(1/1)	1	1
X	(X/X)	X	X
D	(1/0)	1	0
\bar{D}	(0/1)	0	1

1/0
D
↓
0/1

So, these again I will come back to this slide later on, but I will just give you an example the slide there is 5 value algebra basically in Boolean circuits you have two value algebra, but where we are going for test, we go for 5 value algebra not much change one is a D and one is a D prime. So, D actually means basically under normal case the line is one and the faulty the case the line is 0 and D prime means the just the reverse under normal case the circuit is 0 the faulty case the line is one because now as we have fault.

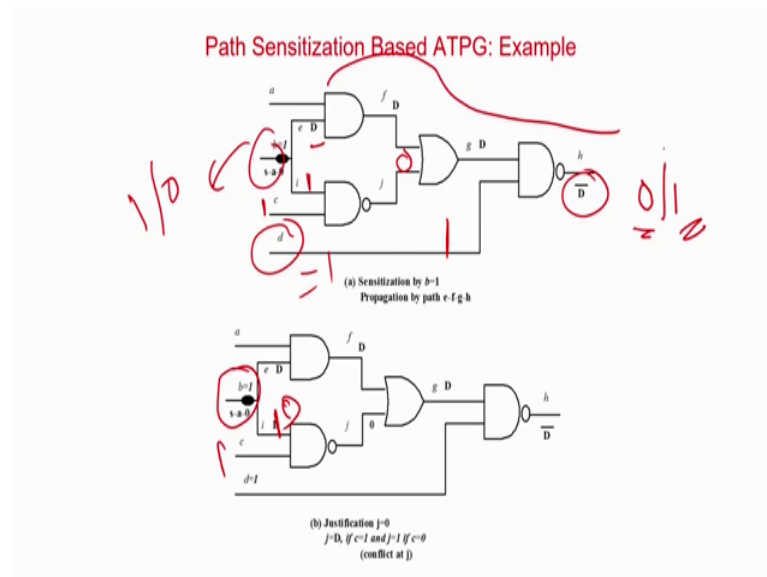
(Refer Slide Time: 58:16)



So, we are extending the two valued algebra to a 5 value algebra just a nomenclature just keep it in mind is what is shown that may be if the output is having a stuck at 0 over here then if I apply a 1 1 the normal case the answer will be a 1 and the faulty case the answer will be a 0 that is the output of the and gate. So, that is we will mark this as D similarly if we are applying 0 0 as if line is stuck at 1. So, the normal case the answer is 0 the faulty case one the answer is the D prime.

So, you can just look at the slides slight this thing. So, may be if you have a AND gate one input is 0 stuck at 1 fault is there another input is a X means I do not know what I have applied. So, in that case the output is X because we know that in AND gate this is a controlling pattern. So, the output will be X this is a roths 5 value algebra.

(Refer Slide Time: 58:54)



Now I am going to take you through an example of path sensitization base ATPG that is not a random ATPG is a deterministic ATPG. So, let me zoom it yeah.

So, let this be a circuit now for example, just see that this is a b is a line stuck at 1 fault I assume that this a defect difficult to test fault just assume that this a difficult test fault; that means, I apply random patterns you could not test it. So, what you are going to do. So, what you mean by sensitization these are logical idea they if this line is stuck at 0 how will you test it you have to apply a one over here obvious if you want to test that whether a bulb is fuse or not.

So, what you will do you will put the light on. So, if the light glows the bulb is or otherwise the bulb is fused, similar logic here stuck at 0 means I have to apply a one pattern here now I have to find out because as I told you that as I have told you that basically there is no extra pin outs. So, only A, B, C, A, B, C and D are the inputs and H is the output. So, only way is that you can apply this patterns over here and you can have see the pattern at the output of this is the only thing you can do nothing extra than this. So, I have to actually bring out the affect. So, what is the affect over here normal case one failure case 0 that is actually D .

So, the D should appear here and here, but what I am trying to do that is I have to propagate the value through the output that is the affect is D . So, the D has to be brought

to output. So, that two paths either I can bring the affect from here or I can bring the affect to this path these are the two paths that is E, F, G, H is one path, I, J, G, h is another path, some path you have to select and you can bring out the output through that path that is actually called propagation then after doing that you have to also justify that whether it is possible by logical signals, but there should not be any clash. So, in this case I am taking a 1. So, in this case, I will get a D because in this case, it is actually normal case one failure case 0. So, it is D somehow I am selecting this path of the propagation path. So, D will be there D will be there this is a inversions. So, it will be D bar. So, basically I should get a normal case 0 and a faulty case one in this case.

So, if I apply if I sensitize this I have to sensitize stuck at 0, I sensitized by 1, the output value I have propagated. So, if the output is 0, then the circuit is normal if the output is one I know that there is a stuck at fault there is going to happen. So, I have selected this path now what I have to do I have to actually justify. So, what do you mean by justify; that means, I should be able to derive this signal.

So, this is a and NAND gate we all know that AND and NAND gate means to propagate the value to the output is a controlling input value is one because if I make this line 0 the output will be default one. So, the fault affect propagation will be lost and you are not going to in and have an difference if I put one here it will be the answer will be 0 and a sorry, if I put a 0 over here is a controlling value, the answer will be a 1 slash 1 at the output, then actually nothing you can observe for both normal and fault case the value will have similarity. So, what I have to do I have to have a controlling value I have to give.

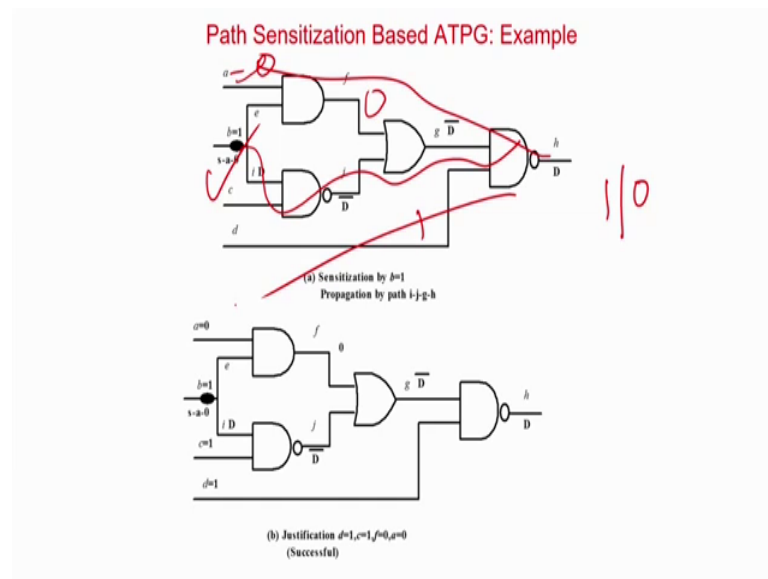
So, actually I will put a 1 1 over here. So, that the D gets propagate here as the D bar. So, this one will have value over here sorry. So, this one D will be settle at 1. So, this actually what I am calling a justification sensitization propagation to the value to the output now I am saying whether it is possible to do it similar, this is the or gate. So, if this is the OR gate what I have to do you know that the controlling value is 0 if I put a 1 over here the output of the OR gate will become a 1.

So, in that case the fault propagation will be loss. So, I have to put a 0 over here. So, if I put a 0 over here of course, you require a basically to have a 0 over here I should have a one over here as well as a I require a one over here that is what is the requirement to

have a 0 over here I should have a one over here one over here. So, 1 and 1 basically is 1 in AND gate and the NAND gate, it is 0.

So, I require a one over here and, but if you see there is a conflict coming over picture. So, what is the conflict that I require a 1 over here? So, that I get a 0 over here and so, forth and C also has to be made one, but this line I am this is a line is having a fault I am putting A B wholes equal to 1. So, this is D and this is D. So, there is a conflict. So, what I want to show here is that automatic test pattern generation also will not very simple that you have to have you have to try multiple iteration to find out which is an exact test pattern and if it is a big circuit you have to apply algorithm to do that. So, there was a problem over here. So, I cannot justify I could sensitize I could propagate, but I could not justify.

(Refer Slide Time: 63:23)



So, what is the option there was another path here, but I could have propagated the fault of it now I will try that. So, let me quickly zoom it and show you. So, in this case I am this was D. So, this is simple that because there is only one with sensitize it stuck 0 where have to apply a 1. So, is one means normal case 1 faulty case 0, now I am taking this path for propagation. So, this path for propagation means D will be inverted D bar or gate D bar and again also set inversion is D. So, in this case what is going to happen normal case the answer is 0 the faulty case the output will be one. So, sensitization is done propagation is done now I have to justify.

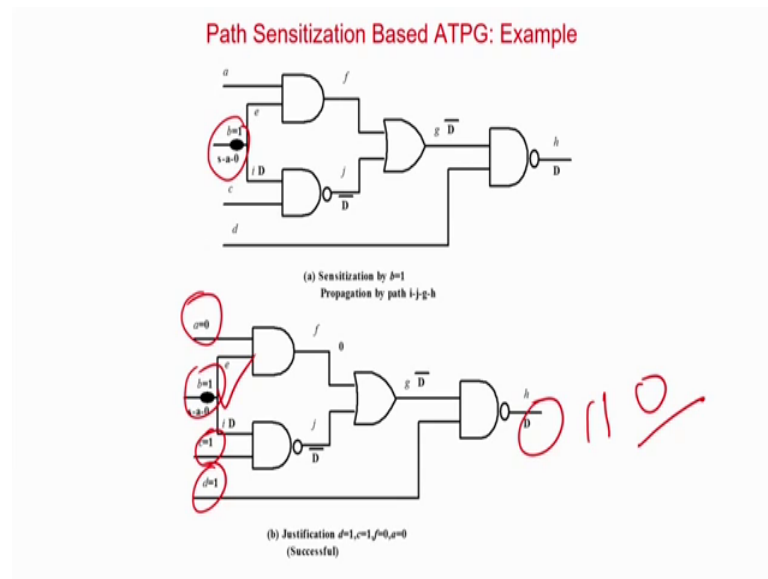
So, again this justification is simple anyway this is D. So, I have to have a one now if you see this line is D bar. So, this line has to be a 0 of course, because always a controlling value will be lost now how do I get a 0 at the output a also equal to 0. Now you have done. So, in this case by selecting the another alternative path basically you could find out the test pattern because fault sensitization propagation and justification are all satisfied. So, what is sensitization take a line which is the fault apply the signal which is opposite to do it justification propagation means you have to value sensitize this is sensitization that is D you have propagate it to some output pin of circuit and justification means you have to track back and settle out the primary inputs. So, all 3 would be done with this new path.

So, in that is what I am saying automatic test pattern generation is not very simple that is why we first go for random test and then we go for this automatic test pattern generation based on algorithm targeting each fault at a time that is sensitized propagated and justify approach we do and, but we want to do it for a very limited number of faults, but again if you take a very large circuit like NOC and SOC, you can find out that this number of difficult to test faults also will be not be less it is also be also be very high the system is large where inside there will be embedded and basically you find out that in one try you may not get a pattern you have to reiterate and find out.

If the circuit is small is a very good standard to follow small in the sense that I am not talking about the NOC SOC even for quite simple level processor also this is well applicable, but if you are going for NOC and SOC level this will take a enormous amount of time even if even for such large systems, but these technology was well accepted in the till about say circuits at the level of processors a mix processor say slightly at the level of medium scale and slightly towards the slightly higher scale it was possible, but with system of chips NOC and SOC even this paradigm is not going to work it will going to enormous amount of time.

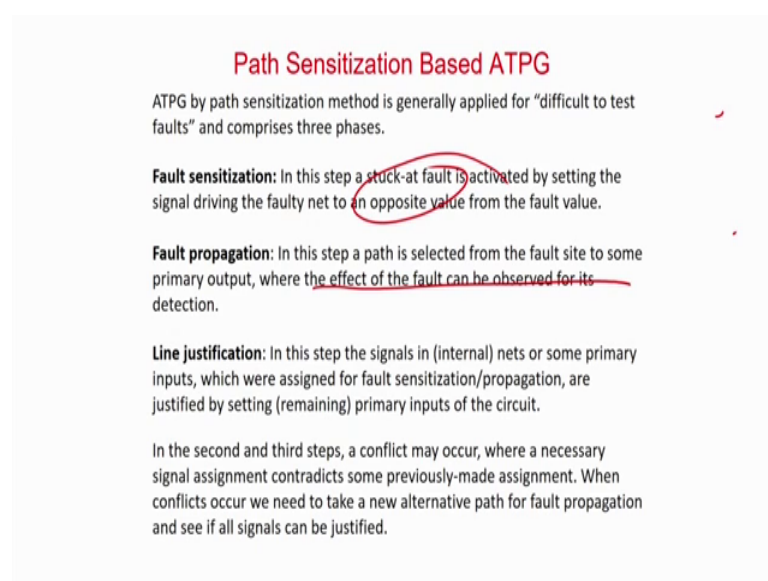
So, when you will be studying optimization will see how to handle this with higher level fault model and higher level of test that you will going to see, but anyway, let us comeback to the basic which we are trying. So, basically now what happens A equal to 0, B equal to 1, C equal to 1 and D equal to 1. So, this becomes your test pattern to detect this fault and the output is D and if the circuit is normal; that means, the stuck at 0 fault is not there.

(Refer Slide Time: 66:15)



You are going to get the answer one if you get the answer 0, then you know that the stuck at fault is there, of course, you will find out that this pattern is not going to test only this fault, they will test several other faults, but I am coming into that in this example I have shown you the deterministic algorithm to test faults which is called ATPG which has to be done for difficult to test faults. So, therefore, this basically called sensitized propagate and justify approach in this case.

(Refer Slide Time: 66:39)



This stuck at fault you are activating the signal to an opposite value propagation means you have to send the fault affect to an output path justification means all the internal signals you have to justify. So, that you get a proper primary inputs set.

So, basically this is the heart of all test that is you have to given a some faults you have to find out what test pattern to apply and as I told you in traditional circuits all job was to get hundred percent fault coverage; that means, some of the faults will be covering by the random manner and whatever will be removing remaining will be going by this ATPG approach that you have to target is individual fault and you find out in this manner.

So, that we get a high amount of coverage, but as you and will be moving to SOC NOC level paradigm even this will be blowed up blow, it will be blowing up because the number of digital faults will be also higher now it is a 3 level circuit in this case it will be a very high level this level means I am taking about this levels this is 1 2 3 4 level circuit, it will be a much more higher level circuits and things will again blow up because the number of stuck at faults will be higher sensitize propagate justification will be difficult because there will be so many alternative path should try and so forth.

So, there be optimization will be at a higher level of test that is not even at the gate level even at the black box level like adder I will consider adder as a block I will consider may be multiplier as a block this is actually called RTL level test and not only and we not even go for stuck at faults we also go for some abstract level of fault models like fault at the adder level fault at the multiplexer level and so forth, mostly they are modified from the software point of view.

So, that will be the idea of how test patterns are optimized to an large circuits and also one important thing now number of test patterns are becoming. So, large that we will have to try to compress them sometimes will be slightly lowering down the coverage even so that we can get a sufficient amount of quality confidence, but still with a very less amount of test time. So, that will be the optimizations we will be talking when will be going for optimization of the combinational circuit ATPG etcetera now the everything was so much about combinational circuits now what about sequential circuits what it will change.

So, in the tomorrow's preliminary lecture, as I told you, the we have to preliminary lectures on the test in tomorrow's preliminary lecture, I will tell you how things will become more complicated or more difficult when you go for sequential circuits after that we will start our optimization.

Thank you.