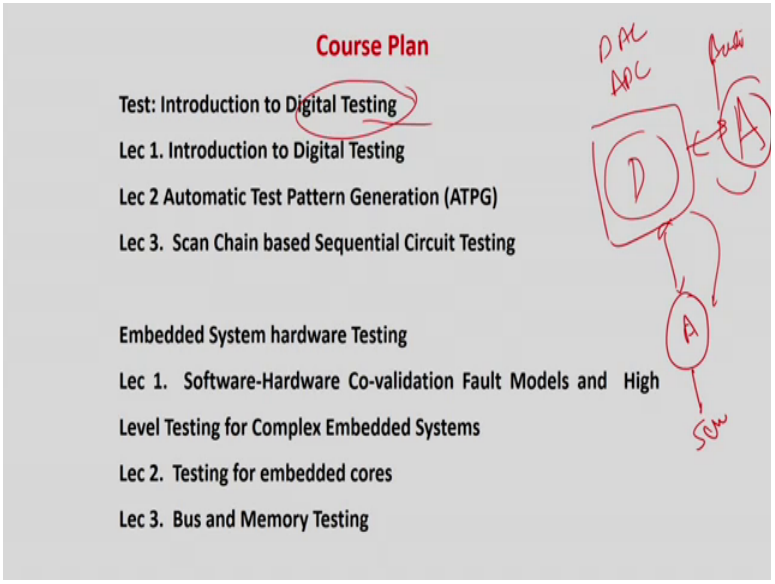
Embeded Systems-Design Verification and Test

Dr. Santosh Biswas
Prof. Jatindra Kumar Deka
Dr. Arnab Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati

Lecture – 26
Introduction to Digital VLSI Testing

Hello. Welcome to the course on Embedded System Design Verification and Test. And, now we are actually in the third part of the course that is on embedded system testing. I am Santosh Biswas from IIT Guwahati. And as we have been told in the introduction introductory part of the lecture I will be telling you on the testing part of embedded systems.

(Refer Slide Time: 00:45)



Course Plan

Test: Introduction to Digital Testing

Lec 1. Introduction to Digital Testing

Lec 2 Automatic Test Pattern Generation (ATPG)

Lec 3. Scan Chain based Sequential Circuit Testing

Embedded System hardware Testing

Lec 1. Software-Hardware Co-validation Fault Models and High Level Testing for Complex Embedded Systems

Lec 2. Testing for embedded cores

Lec 3. Bus and Memory Testing

So, you might have listen in details on design and verification of embedded systems by Professor Arnab Sarkar and Professor Jatindra Deka. Now, after doing this initial part that is how to design an embedded system and how to verify it; finally we have to test it. So, basically what happened in to the last few series of lectures we have seen given a specification what type of models and how they has to be synthesized and how they has to be scheduled. And finally, how design a nice embedded system for a given set of specifications. Then you have seen that as a designs are quite complex in nature so, Professor Deka has taken you through a journey of what is called verification. How do you formally prove that whatever your design or whatever you have intended to design is actually practically correct.

Now, comes the third part which is called testing. Now, my job will be to take you through the journey of testing of embedded systems. So, in this course if you can look the course plan is I will we are assuming that you have very basic prerequisite of knowing about digital design, some amount of automate theory etcetera and all others we are trying to tell you from the beginning. So, that the course become a entirety in itself. So, if you see the first part will give you introduction to digital testing. At the very beginning only on like to tell you about one assumption as you have seen embedded system comprises of some digital hardware, some kind of analog interfaces some kind of sensors and actuators.

But if you start telling in details about what is how is sensors has to be designed, how they have to be verified. Similarly, if you talk in details about the actuators or the analog circuits these codes itself will become actually a 3 semester courses; it will be very elaborate in nature. So, in this course from the beginning we are actually taking an assumption, then we will be mainly concentrating on the digital part of it. Like whenever we are in this part also whenever you will be talking about testing will concentrate our self on the software part and only on the digital processing part. We are not going to cover basically on the analog interfaces or testing on and faults of the sensors and actuators, because those actually again full courses in itself.

So, basically what we will try to do, we will try to tell you about digital testing because our assumption is that for more than 90 percent of your embedded systems are generally process using digital circuits. Only we have some kind of maybe it looks say something like that this will be digital hardware. And there will be some kind of analog interfaces, here analog interfaces will be there and you will have some sensors here and you may have some actuators here.

So, basically this sensors and actuators will interface or we will take real lime readings or and it will translate your competition into actions. So, there will be job of basically sensors and actuators and all general all the processing will happening in digital domain. And, they this analog interfaces basically their analog ADC and DAC and a Digital to

Analog and Analog to Digital Convertors. Mainly those will actually convert your digital signals to analog signal so, there can be properly interfaced.

And, also the others DAC will be there that is an analog to digital and digital to analog. So, such conversions basically will take real world data. So, with the help of sensors AND OR, and we will convert it into in the digital domain so, they can be processed. And, again in another way if you think if there will be something some kind of circuits which will we will take your digital data and convert it into analog that is called DAC. And there will be actually interfacing with the actuators so, that due to the values can be actuated.

So, your ADC or DAC this is analog to digital and digital to analog. So, they are the main analog circuits which are in embedded system. So, I as you can look basic broad discussion you can put have in easily find out that basically most of the processing of embedded software will basically happen in the digital domain. So therefore, when we will be taking about or testing so, we will be mainly concentrating on the digital circuits. And, we will assume that I mean this testing of sensors actuators or testing of the analog circuits mainly the analog interface circuits like DAC and ADC's; we are not going to study in these course.

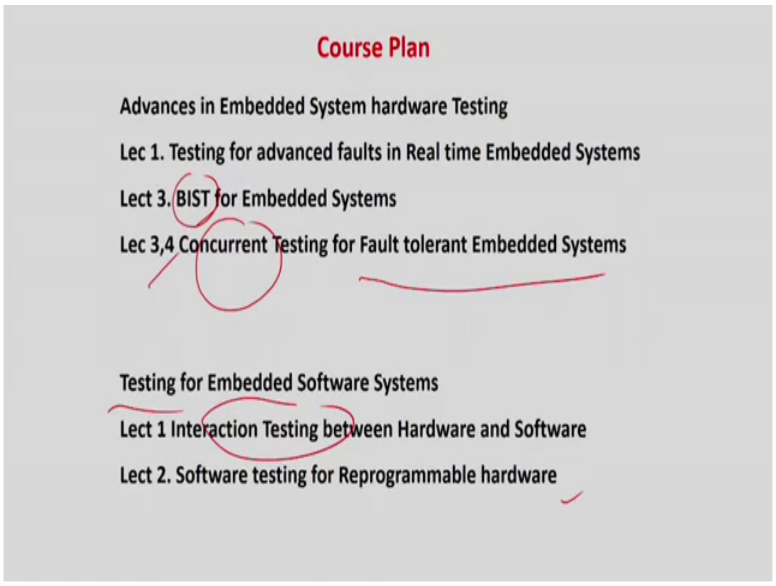
There are full fledged it courses on analog circuit testing; analog verification also is now new subject which is coming up. And in fact, testing of sensors and actuators are very large courses in themselves. So, we will start this course with introduction to digital testing. Then we will tell you how to automatically generate test patterns to test digital circuits. And, basically there will be looking at some interesting techniques to test large sequential circuits. That is that will be the basically interaction part and the prerequisite to know about embedded system testing. Then we will see embedded system hardware testing that is in this case, we will now the diving into depth of testing of digital circuits which will be involved specifically in embedded systems.

Like software hardware co validation fault models and high level testing of complex circuits. So, in this part basically we will try to see if there is a hardware as well as a software and actually as you have already seen that embedded system basically comprises of hardware and software together. Then we will see what are the different type of fault models which will support such kind of a systems where there is a hardware part there is a software part and how different ways you can test those circuits. Basically, they are actually called high level fault models and abstract level of testings which are mainly suited for complex embedded systems.

We will be basically looking at such type of circuits in the second module of this lecture. Then we will also looking at testing of embedded course. So, nowadays if you come consist of if you think of a embedded processor there a lot of course, which are embedded in those say multi core processors system on chip network on chip. So, we will see how basically such embedded cores can be tested. Then we will see that there will, if there is thus number of embedded cores in a chip there will be lot of buses and lot of memory involved inside.

So, we will also have a dedicated lecture to tell you how buses and memories are tested in such kind of system on chips network on chip or NOCs. So, multiple cores how they are tested as well as the interconnectivities, how they are tested and also we look at basically how memory which is also a very very important component of embedded hardware will be tested. So, that will be basically when we will be discussing in depth of embedded system hardware testing, we look into this lectures.

(Refer Slide Time: 06:47)



Course Plan

Advances in Embedded System hardware Testing

Lec 1. Testing for advanced faults in Real time Embedded Systems

Lect 3. BIST for Embedded Systems

Lec 3,4 Concurrent Testing for Fault tolerant Embedded Systems

Testing for Embedded Software Systems

Lect 1 Interaction Testing between Hardware and Software

Lect 2. Software testing for Reprogrammable hardware

Then basically we will also look at then we look for some advanced fault models or whenever your circuit starts becoming complex test model, fault models, testing paradigms everything becomes more and more difficult and complicated. So, in this next module we will see advances in embedded system hardware technique. Now, we will see testing of advanced fault model in real time embedded systems. Like embedded system should also as you have discuss in depth in design and verification the embedded systems not only need to provide correct solutions, but they should also be temporally correct. So, what do you mean by temporally correct? Means, they should be generating the correct data at the correct time. So, there are some real time constraints.

So, we will see: what are the different ways of testing such real time constraints that whether your circuit will be able to means as real time constraints or not. So, that is actually we will be testing for advanced fault models like delay fault models, rise to fault delay at the delay to rise delay to fault such type of test models which will actually determine whether your circuit will be able to give answers in a predefined deadline will be tested. And, how they can be tested we will be covering in advance fault model testing.

Then we will not be discussing something called built in self test that is even just a system starts itself. So, once you test this circuit before after the design and then you sell it to the customer, then who can guarantee that it will also be correct in future nobody can do that. So, there is something called built in self test that is every time you switch on your device now your mobile phone is a handheld device with an embedded system. Your camera is an embedded system, your laptop means sorry the general purpose devices are not embedded systems, but you can just think about your tab. So, it is all kind of an embedded system. So, any kind of handheld devices generally an embedded systems.

So, after a system has been fabricated tester to be properly working I put in your system. Now, how can you guarantee that every time it will be operating and there will be no fault which can occur basically after the system has been sold to the customer. Now say there is something called built in self test, every time a system is started it will test itself and is and only if everything is normal the system will start operating. As it will tell you that this device is having a problem or this circuit is having a problem it needs to be replaced. Finally, nowadays have avionics, cars everything has lot of embedded processes.

Like the automatic cruise control, in the car ABS, anti locking brake systems. Every domain of avionics, cars industry have real time or have very critical embedded systems in picture. So, if the system is of mission critical or involves human lives like for example, pacemaker is an embedded system; you think of a diabetic insulin pump is an embedded system. So, in such cases even if the system is being tested to be operating properly, sold to the customer, you have built in self test and before you start your device it is proper. But, still you should always be testing itself that is called concurrent testing, that is online monitoring because you have a pacemaker or they have a insulin pump.

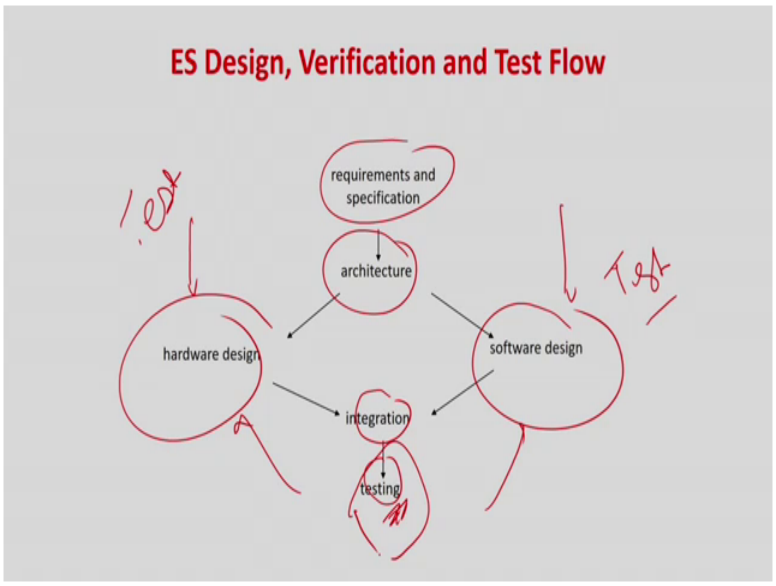
So, everything is properly tested, but after it is delivering insulin say for 20 hours the pump stops or there is some failure. It delivers more insulin or it delivers less amount of insulin, who is going to test that that in a continuous mode that whatever it going to a supposed to do is working. So, in that case there is something called concurrent testing or online testing, that is even after the system has been deployed it is doing its normal job still it is being monitored by a hardware or monitored by a system that throughout this operation it is properly working or not. So, that is called concurrent testing for fault tolerance.

So, we will have two dedicated lectures for that because, it is a requirement for lot of embedded systems because most of the embedded systems are now mission critical or they involve such applications which can have life threatening implications. So, we have put two lectures on designing of fault tolerant embedded system. Then finally, this last module is something interesting because you know that embedded system is there will be a hardware as well as a software which will be running on it. So, we will see how interaction testing happens between hardware and software because, mostly we will see that whenever if you are talking of embedded system testing also mainly we will be looking at the testing of the hardware.

But some because, but some amount of emphasis should also be given to understand that if there is a hardware and if there is a software which is running on it which is flexible so, what type of fault modeling and how such systems could be tested. So, you have to also look at something called interaction testing between hardware and software. Last but not the least FPGAs, CPLDs or reprogrammable devices are now occupying a big hardware platform for embedded systems. Instead of having application specific hardware we are trying to program the FPGAs and we are asking, we are downloading the soft codes on them.

So, how to test reprogrammable hardware like FPGAs, CPLDs will also be will also dedicate one lecture for that because, as I told you nowadays lot of embedded applications are designed and they are dumped into an programmable device. So, that you do not have the headache of making an application specific IC every now and then. So, that is also a new the new paradigm shift is going from ASIC to FPGA based implementation for embedded systems. So, we will also try to see basically how such systems can be tested. So, in broad sense it is what is our course plan.

(Refer Slide Time: 11:50)



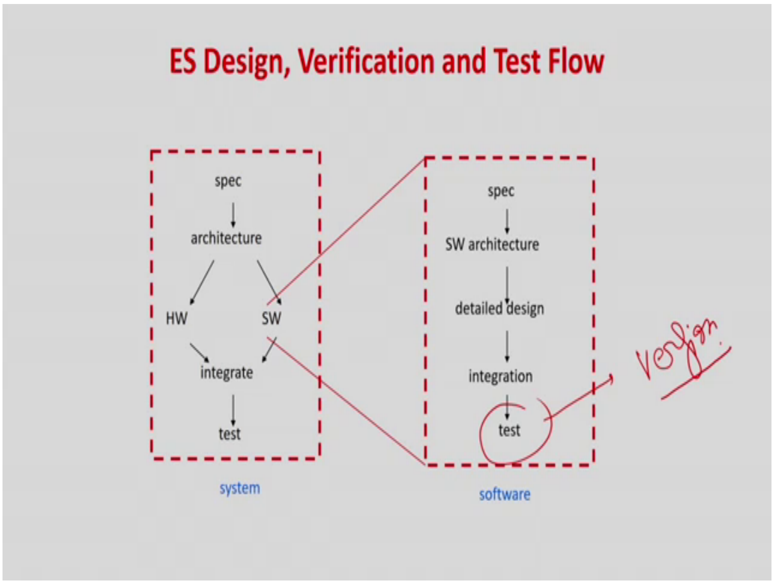
ES Design, Verification and Test Flow

Now, we will start today's lecture which is basically on introduction to digital testing. And with also motive it why digital testing is so, important and why you should have a prerequisite of digital testing when you are know, when you studying basically embedded system testing. So, if you look at this slide this is the again I think this is a repeat slide we have been looking at this slide many times in this course. So, basically embedded system design, verification and test flow; so, basically we have the functional requirements and we have the architecture.

Then basically as we have a software design part and we have a hardware design part, we bifurcate basically and model architecture also having some kind of internal feedback among them. So, that you can switch some of the hard designs from hardware to software and vice versa, but to keep things simple we are actually not studying it right now. Then finally, there will be an integration and finally, you have to do a test. So, this is called the integrated testing, but if you do testing at the last phase there can be lot of issues. Like a bulk found over here can be buy here and buy here, though then you have to do lot of debugging and lot of hard effort increases. So, basically what people does they will also have a test over here and they will also do a test over here

Once there find that both of them basically are operating properly then basically you can do integration and find and test. Slightly, there is a difference in the terms of testing and verification in embedded in this flow, I will come to that. Because, when I say test and when is say verification they are actually something have certain different meaning when you are studying embedded system design verification and test. So, maybe slightly surrounding means slightly confusing, but after 2 or 3 slides I feel the confusion will be over. So, hardware I say test and also I can say software as test and then we can call it as the integration testing. Hardware test, testing of the software and then final integration and you do the final test.

(Refer Slide Time: 13:55)

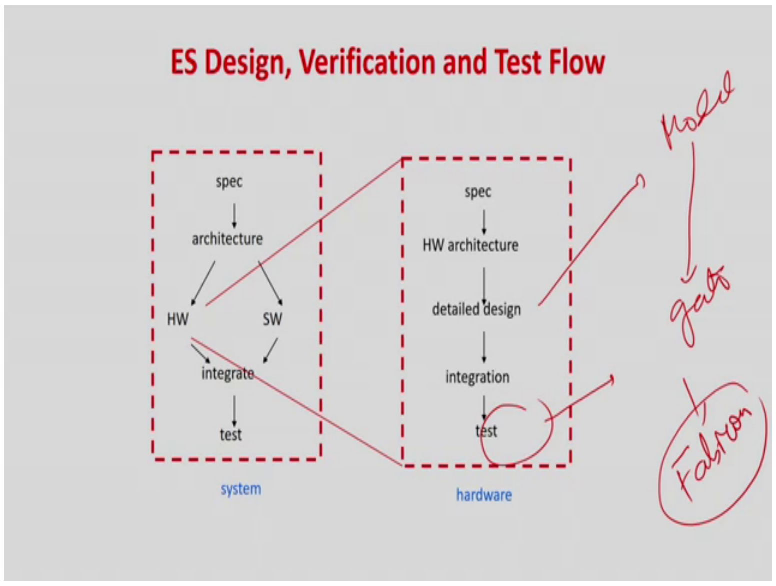


Then what if you just do not put a test module over here and analogs module over here and if you try to do the testing over here, then it will be very difficult to debug that where the exact issues is. So, therefore, what people do there is something like again just say elaborate it here. So, this is your hardware part and basically you try to test the hardware parts in itself that is specification or the hardware design flow is specification hardware architecture, detail design integration and test. This integration is basically only for the hardware components that is your decide what part you design in hardware and what part you will be deciding in software. And, then basically you have a totally separate flow for the hardware part, you do specs, this is design part specification, hardware architecture digital design integration and finally, basically test it.

So, this part actually corresponds to hardware. So, that is you have this one you are doing it separately, designing testing separately; the other part basically is your software part. So, in this case let me just remove you know. So, this is the software specification software architecture, design, integration software part and test. Now, this is a subtle difference between test and verification. So, here actually I should use the word verification.

What is the difference between verification and test? I will take a detail slide only it later, but for the time being verification means you have designed a software or a hardware. Then basically whether your design was proper or not whether you are inserted any or by chance bug is there inside or not you want to verify that. So, that is actually verification which has been taught in details by Professor Deka. So, you might have noted from there that basically testing a software means basically it is a verification because the problem is something which you have done missed by mistake and so I can use the word verification here.

(Refer Slide Time: 15:27)



ES Design, Verification and Test Flow

Then what is test? So, whenever you are talking of hardware, hardware is basically designed by say my modeling and then we are converting into high level synthesis. Then you are making RTL design, and finally you are making a hardware which is called fabrication. So, this hardware if I say the detail design so, you are having a model which is done by a actually Professor Arnab Sir. Then finally, you have your gate level circuit and then you finally, go for something called fabrication; that is your hardware design. So, if I have to verify whether the design of hardware I am doing is fine or not is actually called you verification. That whatever specification I have to taken whether I have made a proper model or not whether I am designing it properly from the from the high level model to an RTL circuit and then I am translating into the gate level, whether such things whatever I am doing is proper or not you want to verify that.

So, that is actually called verification of design. Now, after the verification of design has been completed; that means, you know whatever you wanted to design is proper. However, translating an RTL circuit to a gate level everything is proper, now you want to fabricate. So, in fabrication also lot of problems may happen that is the idea is that maybe when you are trying to fabricate a thin copper line say there is a break in that copper line; you want to fabricate a transistor maybe the gate is slightly disconnected. So, what can happen is that even though your design was absolutely proper because, you have gone for modeling, you have gone for formal verification, you have got some you have given some test benches to verify; every care you have taken before we are doing the design, everything is proper. Now, you have send it for verification.

So, sorry send it for fabrication while fabricating it the manufacturing industry has inserted some kind of faults in terms of defects that is you wanted to fabricate a AND gate, but somehow the output of the AND gate is cut. Because, that was improper doping of this what do I say this is improper doping of proper. Then that is actually a design error or I am say I should say a fault which is not because of you, it is because of the fabrication. So, if I assume that the design is properly verified and correct and some faults are getting inserted at the design at the fabrication part, we want to verify them that is actually called testing.

So, there is a certain difference between testing and verification. So, when you are saying testing that is I have to find out defects which has occurred not again quote unquote not because of the problem with the designer. Designer was absolutely correct he has given me a proper design, but while fabricating it some fault has been inserted because of the mal or malpractice or you can say I mean defects which is coming in because of the fabrication industry; no fault of the designer. So, that if you want to verify if you want to find out such cases that is actually called testing.

So, this I should called actually a test in real sense. And what is verification? Verification is something like this one. So, in this case I should use the word verification, software part is always verification because software you are never going to fabricate. So, whatever you are writing as a code is going to execute. So, there cannot be any defect in circuit because of somebody else is fault, if your software is not giving you a proper value that is the fault of the designer. So, we always verify software. So, Professor Deka's lectures covered verification part of the software.

Now, if I see that the hardware design everything is testing, no in case of hardware a hardware part of the embedded system there is both verification plus test. So, again repeating when you are talking about the software part of embedded system everything is verification. Why? Because, you are writing the software and this is the hardware platform to execute. So, if you there any problem in the software is because, of your mistake or the designers mistake. So, if I want to verify that whether I want to establish whether it will execute properly and implement my logic you have to formally verify it.

So, there is no kind of testing involved over here, but again I am reiterating because this is certainly confusing slightly confusing. But, when you talk about hardware that is both test plus with. So, verification plus test I should rather call it the other way not (Refer Time: 19:25) test verification, it should be verification plus test. Why? Because, if I want to design a processor nobody goes to the industry and put transistors together and make a circuit like that, that nobody it does. If you look at VLSI design verification and test flow we have lot of NPTEL lectures on that. So, in VLSI design verification and test flow what happens, basically you start designing a circuit in terms of RTL or there is a model which Professor Arnab Sarkar was telling you.

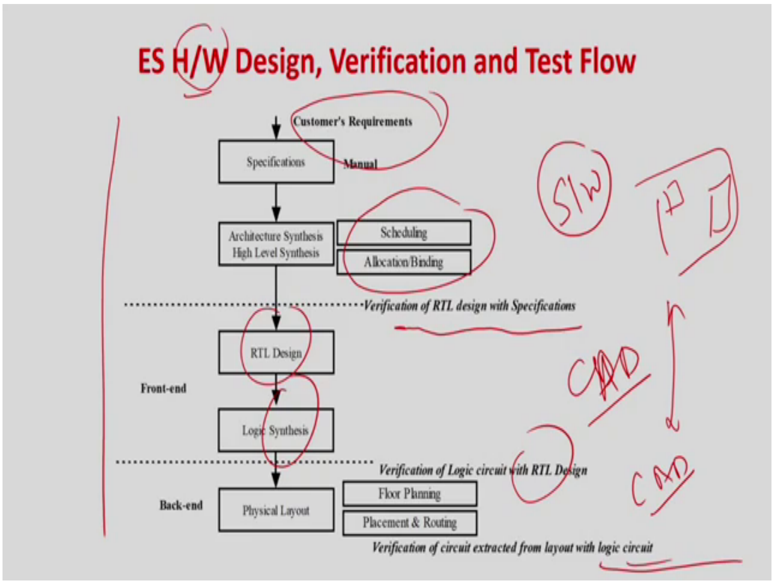
Here we design in terms of specification that is your or different type of models. Then you will make a hardware architecture and then there is a detail design, detail design means you are going for high level synthesis, they are going to make an RTL design and then you are going for a gate level design. You can read any standard VLSI design flow for elaboration on this. All these are basically done with the help of CAD tools, you or in this software paradigm that you write a very long code what do they are called, HDL codes. So, you write a hardware definition code for the design, then you synthesize it, then you design the gate level circuit; all these things are done by CAD tools. Every level you have verification stages.

Sometimes if you find out that there is certain issues basically which is happening because you have made a mistake in the design, you can reiterate and change in software. So, designing a hardware circuit or a VLSI circuit nowadays is not like going to the industry and putting the transistors together, you designing the software level. Finally, when you are satisfied with your design you send it for fabrication. So, before sending it for fabrication whatever steps you are following is basically verification of your hardware design, hardware design. After you have send it for fabrication there are very high chances that many of the devices are going to have defects.

Now, why they are having defects? This is because the fabrication may have problems and there is lot of I think as I as I have given you an example you might have design a nice proper AND gate, but this may be getting a cut because doping of silicon doping of copper was not proper. So, then this fault is happening not because of your error, it is happening because of the problems with the Fab industry. So, that part is actually called testing.

So, there is a certain difference between testing and verification. So, in this unit basically that is unit number 3 that is the last unit will be mainly talking about testing of embedded system. We are assuming design is proper, software design is proper, but some problems happen in the fabrication stage; how you can find them out. So, that is basically the core idea. So, I have given lot of time to discuss on this because, you have to understand the difference between verification and test in terms of embedded system flow.

(Refer Slide Time: 21:45)


ES H/W Design, Verification and Test Flow
Customer's Requirements
Specifications
Manual
Architecture Synthesis High Level Synthesis
Scheduling
Allocation/Binding
Verification of RTL design with Specifications
RTL Design
Front-end
Logic Synthesis
Verification of Logic circuit with RTL Design
Back-end
Physical Layout
Floor Planning
Placement & Routing
Verification of circuit extracted from layout with logic circuit
S/W
H/D
CAD
CAD

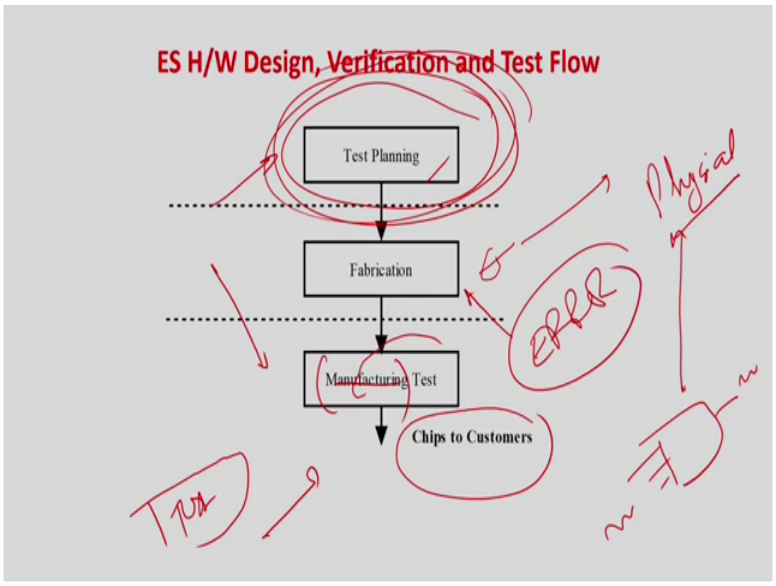
So, this is the embedded system design verification and test flow, mainly the hardware part that is why I am emphasizing on hardware. So, for example, somebody has told you that this part will be hardware this part will be software hard part hard hardware part is given to you. So, that is your customer requirements maybe you want to design a GCD calculator, I want to make a embedded system controller for a microwave movement. This specification has been told to you, that is manual. Then there is something called high level synthesis, you decide that what will be the broad architecture for, how many adders we will put, how many ALU's you will put, how many memory registers will put.

So, basic architecture you have to make it. So, that is actually called high level synthesis then there is something called allocation and binding. You can read any standard course of look at any standard textbook or courses or VLSI design you can find this out. Then basically there is something called and always you see verification of RTL with design specification, design specs are given to you then you are deciding how many adders, how many multipliers, how many ALU block and what will how they will be scheduled. So, whether that one will meet your requirements or not basically, you have to do a verification with it. They have and again all these are done in CAD tools or at the software level; even if you are designing the hardware, but you are still writing it in the level of software.

Then there is something will RTL design that is Register Transfer Level design, this is something called logic synthesis and if you translate into a gates there also again done by CAD software. You are not doing it manually, all CAD tools will be available to do it and they are still at the software level. Again you have to basically go for basically your verification to see that whether the translation happening is equivalent or not. Then there is something called physical layout, floor planning placement and routing. Now, all your gates will be converted into silicon, into transistors because they have to be placed. And finally, routed because at the end whatever you have design in digital level will be finally, fabricated in terms of your transistors.

So, again you make a plan of it that I will put my memory over here I will put my some AND gates over and so forth. So, this part is actually called back end of VLSI design. So, we will make a plan and then you decide how do you interconnect all this because, large number of transistors will be there. So, manual you cannot make interconnection. There are lot of software or algorithms to do that and this is also still in the level of CAD, we call it the back end level. So, still now everything is in the level of your transistors interconnects etcetera, but still they are at the level of software.

(Refer Slide Time: 24:02)


ES H/W Design, Verification and Test Flow

Once this is done basically finally, it goes to the fabrication state where things will be manufactured at the Fab level. That is however, you have told whether the transistors will be placed memorize will be placed the Fab industry will do the silicon mapping. They will do all kinds of doping etcetera, and finally the silicon will be available to you. There is a important part which is called test learning that is now, what is this I have told you the; if there is some error over here. The errors are getting inserted at the manufacturing sorry at the fabrication level not this one sorry. If the errors are actually coming in the fabrication level then you have to do a test not a verification.

Because, whatever you have done up to this is correct every stage you have verified, every test you have done a simulation based test verification everything you have done. And. we are assume that the designer is very knowledgeable. So, everything is proper up to here, but certain errors are happening at the level of fabrication. This is in I will discuss in the next few slide; assume that because of some problem with the Fab industry some kind of lines are disconnected. You wanted to fabricate an AND gate, one leg is missing all these things are happening for which the designer is not the person to be blamed.

So, you have to verify that that is you have to put some inputs, you have to study whether the signals are appearing proper or not. So, you have to do it in the physical level because, where I say that manufacturing so, when at the saying that fabrication chip to customers. So, now it is no longer at the CAD level they are physical chips. So, these are actually your physical devices. So, this is no longer at the CAD level. So, whenever you are talking about manufacturing test it will be no longer of verification, it will be a test. Because, we will put the device physical device you will apply signals over there and at the output also we will have signals and see whether it matches the golden response.

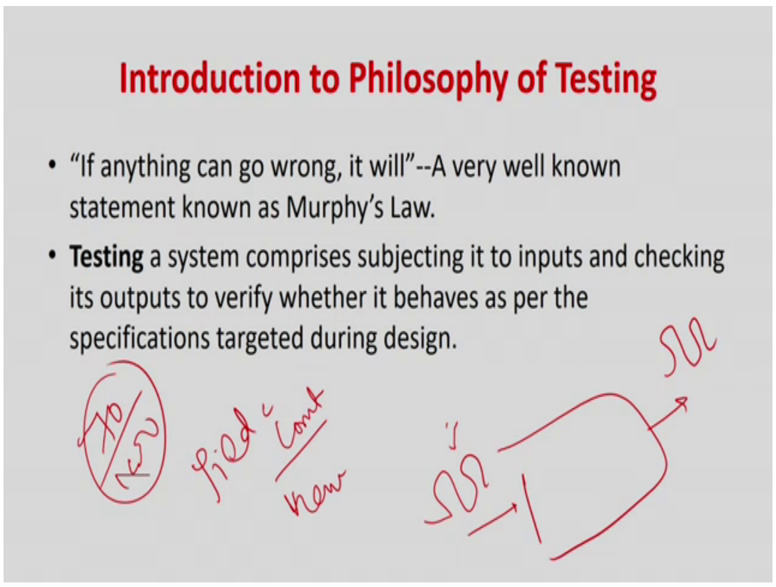
So, this testing is basically that is called manufacturing testing. In fact, we should always use the word manufacturing test because, test cannot be at the software level; software level everything is the verification. But, we generally do not use the word manufacturing test all the time basically we was the use the word test. But, test means the physical devices available to you, you are putting physical signals in you are getting on physical signals and trying to see everything is proper or not. Then there is something called test planning because, you are doing all these things. Finally, you have to apply something over here to see whether the real physical devices which is come out of the Fab is operating properly or not.

What we will do, how we will apply test patterns, what we are going to expect. So, all this planning we do beforehand so, it is actually called test planning. We will see why we cannot put test planning over here. We can also put some test planning over here and drop it; we will see what is the issues with it. The idea is something like a parties some groups are designing a circuit and finally, you have to manufacturing. And finally, this circuit will be available to you another group is just monitoring what they are doing and they are trying to plan when the chip will come to you, what do you will do what type of signals physically you will apply. So, that you get you get to find out whether the circuit which has been fabricated or not is free of defects or not.

So, the planning goes beforehand. So, you always try to do this test planning whenever we start the design flow itself. So, these are actually giving you an idea that what do you mean by embedded system hardware design verification and test flow. So, we will be mainly concentrating in this part of the course was the test planning and basically manufacturing test. How do you plan the hardware test and how do you basically physically testing, this is our main goal. Physical testing is you have to put the device in the tester, apply the signals get the signals. So, these some physical activity mainly our job will be to do the test planning. Again I am making around circles over here because, our job will be the plan what will you do. Say for example, somebody gives me 10,000 circuits physical circuits and tell ask me which one is proper and which one is not proper you have to test it.

Then we should have a plan beforehand, then if it comes what I will apply, what I will major and I will tell this is good or this is bad. So, this is actually called test planning. So, I cannot wait till the device fabricated this is coming to me because, fabrication will take some amount of time. So, I should also start my plan beforehand because this part are all at the software level or the at the CAD tool. So, I can have the design on my machine also and I can start doing the plans. So, that is what is actually called the embedded system testing or test planning. Mainly test planning is important because, physical testing means just you put the devices and do the test.

(Refer Slide Time: 28:09)



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.

Field
Const
New

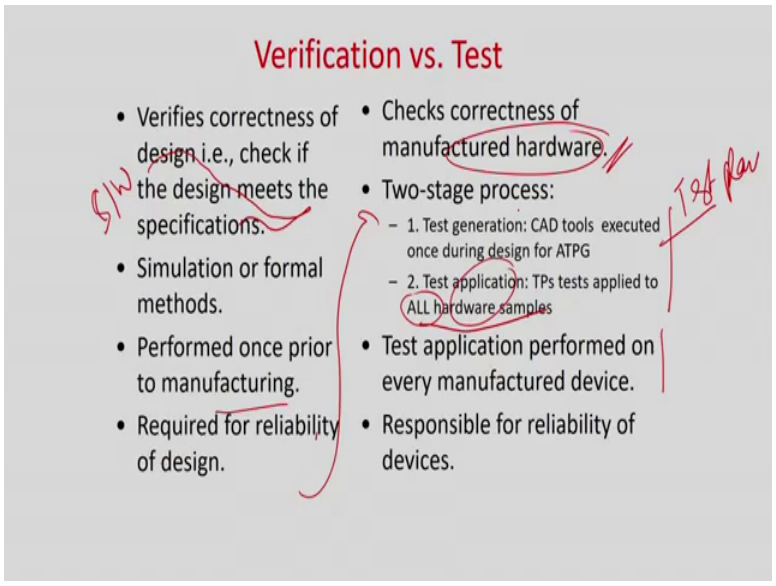
So, now we are going into the introductory part of hardware testing for embedded systems. So, why faults will occur for the first question, your manufacturing devices fabrication industries are very sophisticated industry then ask why a fault will occur. Say for example, I can tell you that some person is repairing or making cycles for me. So, should I ask that out of 100 cycles he will manufacture 10 cycles will have some problems, never it will never happen. So, if I talk about the classical system design like an electrical fan, bicycle, scooter, pen or your lock physical locks. So, if they manufacture 10,000 locks maybe only 1 or 2 will have manufacturing defects which we can throw. But then why it we are shouting so, much about hardware testing; that means, there is really some problem with the Fab.

So, if I say in 10,000 chips will be manufactured out of 10,000 maybe 500 or around say 6 40 percent of the chips may have a problem. 40 maybe slightly higher side, but if I set 1000 chips I have sent to fabricate maybe 2000 will have defects So, there is a good chance that some of the circuits will have defects then why, why such a situation arises. Because, such a thing is not very much practical or it will not see such situation see in something called your what do I say in basically your classical systems like electrical fan, cycles, motors, electrical iron. So, if you manufacture 1000 maybe 1 or even 0 will have defects. So, why is it happening so, much in VLSI. So, you are going to see that one. So, a very important statement which holds in VLSI is called Murphy's law. "If anything can go wrong, it will" a very well known statement; that means, manufacturing of circuits has certain issues which we look at today has something that is not very probable of working properly.

If you manufacture 100 devices 20 or 30 may have some issues with it. So, it goes very well with this VLSI Fab fabrication and this is something can go retrial. So, what is testing? Testing a system comprises sub subjecting it to inputs and checking it output to verify whether it behaves as per the specifications targeted during design. That is for any system whether VLSI whether practical whether physical everything you bring the system, but you device put the inputs get the outputs and check whether whatever you intended on or not is properly functioning or not. In case of digital testing we will put some digital values and output you also see digital outputs and match what is intended.

Now, problem of testing happens because there is something called yield. Yield is nothing, but manufacture devices by say correct. How many devices are properly manufacture and total number of manufacture devices? Say 70 devices come out without any faults out of the 100 devices. So, yield is actually 70 percent you will be very surprised that 70 percent is a very good yield in VLSI. But, in case of if you think about electrical fans motor siren yield is more than 99 percent because, all even higher 19.9999 percent. Because, such cases there are very real you can be it is very very rare to have some kind of defects in the manufacturing; why is a very interesting story as we will see.

(Refer Slide Time: 31:07)



Verification vs. Test

- Verifies correctness of design i.e., check if the design meets the specifications.
- Simulation or formal methods.
- Performed once prior to manufacturing.
- Required for reliability of design.

- Checks correctness of manufactured hardware.
- Two-stage process:
  - 1. Test generation: CAD tools executed once during design for ATPG
  - 2. Test application: TPs tests applied to ALL hardware samples
- Test application performed on every manufactured device.
- Responsible for reliability of devices.

Just before that let me again tell you what do you mean by difference between verification and test. So, what is verification? Verifies the correctness of design, check if the design means this specification; so, I should say that they are at the CAD tool level or at the software level. I should say the design is at the basically a software level, if there design is not any hardware level. How do you do it? Simulation or formal methods described at Professor Deka that is you apply some patterns and see whether it is operating or not, but at the software level. And, formal verification is by the formal methods.

Performed once prior to manufacturing of course, once you verify your software is proper you need not test it you need not verify it again. But, you may have to do it multiple times, but once it is found to be proper you need not do anything else; so, it is done only once prior to manufacturing and this is going for system reliability anyways. So, what is test? Checks the correctness of the manufactured hardware that is very very important is at the hardware level; everything after this will stage will only after this. So, you have formally verified or you have verify that whatever designs you have made at the software level bracket is proper. Then you have manufactured in then you want to test it that is actual called testing. The physical manufacture hardware is there you apply test patterns and see why what happens. So, verification will be software level, testing is all at the hardware level. So, we do not use the word explicitly called manufacturing test. It is a two stage process very important.

Test generation CAD tools executed once during the design for ATPG Automatic Test Pattern Generation. So, as I told you there is something called test plan and second is called test application. Application is very simple you put the hardware apply the signals and see the signals. Test patterns are apply to all hardware samples this is very important of course, if you have 10,000 devices which is manufactured any one of them can have the fault. Because, it is at the problem of the manufacturing stage, but design once you are doing in the software level that is only is manufactured n number of time expected. So, basically verification therefore, they called it done only once, once the system has been established we verified the proper is manufactured.

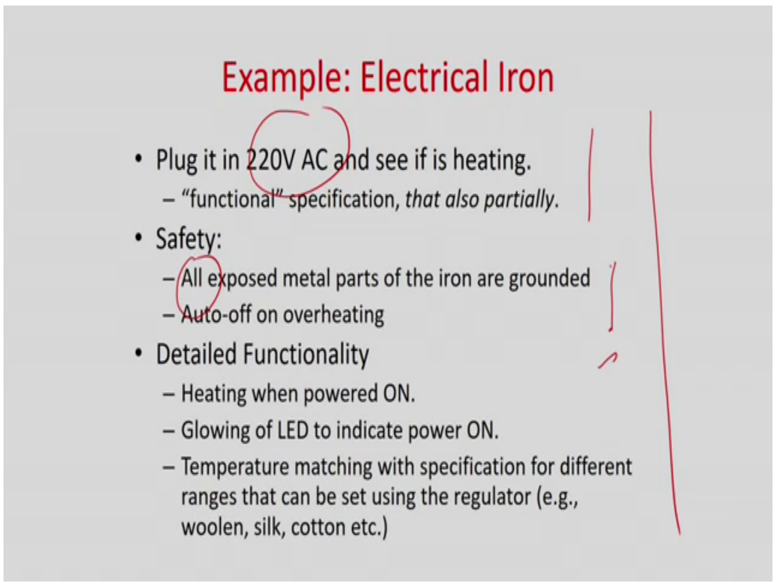
So, the software level is done only once for a design may be we have multiple iterations, but only once it is verified to be proper you will need not verify it again. But, test means 10,000 samples everything you have to test because manufacturing defect can happen in any of them. So, they are called all and all hardware sample. So, there very bold lines all and hardware sample, but again it is a two stage process. Why? Because, testing also have some kind of a software part in it, that is how do I plan a test. I have given your very big circuits of 1000; 10 millions of gates, how do you want to test it, which test patterns you will apply, can we do it manually no.

So, there are lot of CAD tool solutions when you give the circuits they will do test plan with you or help you in making the test plan. So, that is actually called the test pattern generation or ATPG. So, there are tools which will guide you in generating the test patterns at which test pattern you will apply or they will do it in test planning. So, that is actually called test generation or test plan and the second part is called physical testing which is actually the test I was about to tell you. So, testing basically has two parts test planning and one is test application. Application is simple all manufacture device one by one it will come, apply physical signals I get it they need to be done.

But again the generation part is or the test generation of the planning part is a software solution you take the design and start planning what I have to apply. So, a test application is performed on every manufacture device and also it is for the reliability. So, I think I have given able given to able to give you an idea of the difference between verification and test. So, whenever in this third module whenever we will be talking about hardware test so in fact, we are talking about two things: test planning and testing the hardware. So, testing the hardware we will not look at much because, as I told you it is just like taking the device connecting some hardware and measuring the signal.

Mainly, we will be looking at how to plan the test and what we will apply, if I have the plan with me I can easily use a hardware tool or hardware device to apply the test patterns and measure the values; very very important is the test planning. So, if whenever if I still say testing of embedded system, testing of embedded system hardware, testing of embedded system I will be mainly meaning the test planning of the hardware. Or the test plans or how to generate the test plans or the algorithm should generate the test plans that will be my main focus, when I will be calling testing of hardware systems right.

(Refer Slide Time: 35:22)



Example: Electrical Iron

- Plug it in 220V AC and see if 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.)

Now, the question why hardware testing is so, much important? Why we were telling about so much emphasizing. So, now, for the time being let us forget about the VLSI, let us take a simple electrical iron and let us see what it means. So, the basic test of a hardware, basic test if you see an electrical iron how do you test it, it is a classical system. Hardware unit means I am talking about a classical system, you plug it into 220 volts AC and see whether it is operating or not. So, that is actually a test generally we do this, this is called a functionalities. And in fact, it is also a partialities. Why? Because, whatever amount of hardware it is what I tell you whatever how number of irons you bring or the test is done at the sharp level, very preliminary level you put it in the circuit or means which put on is heating it is sold to you.

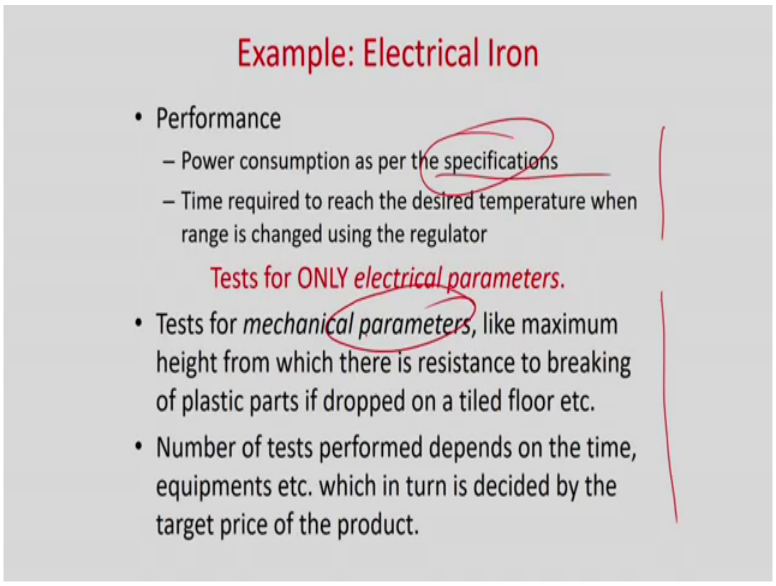
Maybe certain similar test is also done in the industry before after manufacturing, but more or less it is not required. Because, generally what the yield is very very high more than 19.9999 percent because whatever you manufactured generally is proven is always subjected to working. Because, why the question is the classical system they will have some 300 years ago and more or less the technology remain same. So, maybe size sophistication has come you can change the temperature, you can put some well or some water to make a spray. But, otherwise the technique still remain same like if there is a coil it gets its get heated up and you can use the iron. Like a fan the technologies more or less remaining similar for last 200 years, there is no change.

So we have the; or the manufacturing unit is very very well versed how to manufacturer fan, how to manufacturer iron. So, there is a very very less probability then any defect will come in between. So, therefore when you are talking about classical systems like your fan, cycle, motorcycle bike or scooters generally the test part is very very preliminary. Because, the as a manufacturer I know very well what is my job and the functionality more or less remain same over the last 50 or 60 years. So, that is the key role that classical industries testing is important not, but not very very important as in the case of VLSI ok.

So, if the basically if your electrical iron is bit costly so, we will also see the test for safety that all the parts of the irons are grounded, there is auto off facility for over heating. So, you can increase some amount of test. So, you can see testing things whenever things starts becoming complex testing complexity also increases, if it is a simple electrical iron just heating. So, the test is simple put it into the AC on it and your job is done. But if it is a more expensive one; so, you have to go for all exposed parts are grounded, auto overheating off. Even you can go for more detail testing like heat when the power is ON, glow off the LED, then temperature matching is specification like wooden, wood, silk, cotton etcetera.

So, more sophisticated a device become more sophisticated the testing becomes, but still this technology was still no more than 50 years ago. So, and technology more or remain same. So, the yield is very very high. So, testing is not that important in such cases because if I device 10,000 chips, 10,000 irons then the only 1 of 1 or 2 will have a defect and easily it can be very easily verified by the shop keeper.

(Refer Slide Time: 38:30)



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.

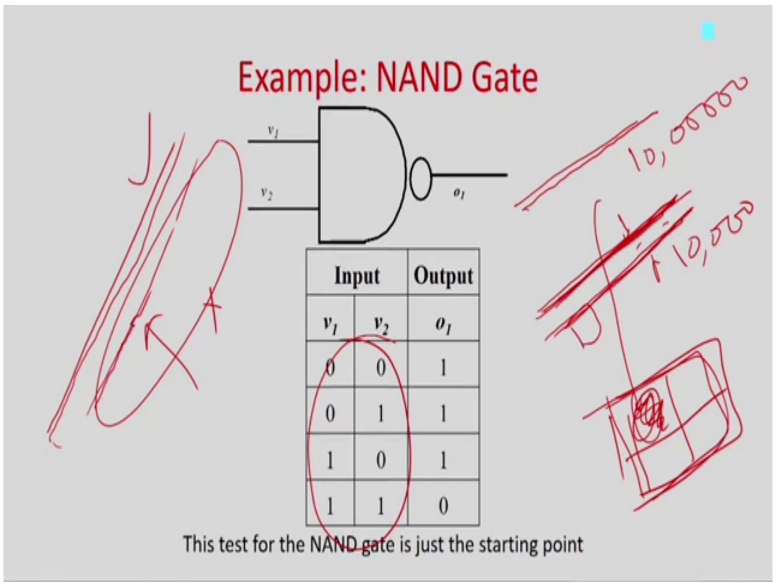
Then if the electrical iron is becoming more and more sophisticated and you need a performance guarantee etcetera, then you have to see for the power consumes consumption like it is a 5 star AC. So, you have to also measure the power requirements. So, you have to also see the power consumption specification. Then time required to reach the desired temperature when the range is changed in the regulator. So, I have to give you a performance, but in the iron such things are never required that I changed from silk to woolen. So, I put to do it so, I will generally give certain amount of time. There is not an even a hard real time, hard real deadline there you should happen by this time.

But assume that such things were there then the performance testing will also come into picture. So in fact, there is something called basic functionality, detail testing like safety basically then performance. So, more a device become sophisticated more amount of testing has to be done. Another philosophy you can see over here is basically, if the device functionally remains primarily which is not changed over years then testing basically remains a very simple phenomena. Why? Because, we know how to manufacture those simple devices over 50 to 60 years or even 100 years so, yield will be very very high in this case.

But, if you want to fabricate or design something very new which is changing everyday then you are proven to make errors even in your manufacturing stage. So therefore, in such cases you have to always test all the devices, again certain stray parameters as I am saying for the mechanical for the iron test. Say for example, I want to test the manufacturing parameters that if I throw the; or if I drop the iron from a certain distance whether it will break or not.

So, I mean nobody will do such a test in a iron, but some I am just trying to give you the example. That testing can be infinite in nature, these tests spaces infinite; it can be a simple function test to a huge test even for a simple iron. It all depends what is the time you have, what is the cost of the device and what is the reliability required for this. It will tell you what to test and what not to test depending on all the parameters. Now, I will basically come to your circuits. Why it is so, much required I want to test this simple NAND gate.

(Refer Slide Time: 40:30)



Example: NAND Gate

Input		Output
v_1	v_2	o_1
0	0	1
0	1	1
1	0	1
1	1	0

This test for the NAND gate is just the starting point

So, what you will you do? You will give me all these 4 combinations and the output of the AND NAND gate, we know that for all combinations except 1 1 the answer will be a 0. So, basically you can say; I want to do the test. Now, compared to iron or a cycle or a or a gadget mechanical gadget whose design is see not change over the last 100 years. NAND gate you can think that it has been designed last 30 years or so in the range of VLSI. But, if you look at the VLSI we first had simple integration, then we had medium scale integration, then large scale integration, ultra large scale integration. And, now we have a very very large scale integration or even now we are not using more terms above

VLSI. But, now we have extremely sophisticated device with millions of transistors are dumped into a single IC like NOC's and SOC's.

So, maybe what I will tell you what happens. So, when we are medium scale integration. So, basically what happens in fabrication general you have to layout make copper wires maybe then you will have a gap, then we will put another copper wire, then maybe we will put a wire over here. So, there is a certain gap in between these two. So, using your mechanic dopers we actually fabricate this lines may be silicon copper wires and we put the transistors and there is a gap. Now, what type is if I make the gap quite large, very good there will be no problems?

And, if I say that I make these lines very thick and in the gaps are high then you can understand that the chances of failures will be less that the manufacturing failures, because I layer 1's then a layer 2. So, there can be any discontinuity and if I make the gap quite large there will be no chances of any short in between. But now if I do this your chip size will become very large. So, if we have a very large IC, if you look at device physics manufacturing a very large chip in a single die lot has lot of physical problems you cannot do that. I do not want to going to depth, but just take for me take quote unquote that are I cannot have a very big single line, it has to be smaller once.

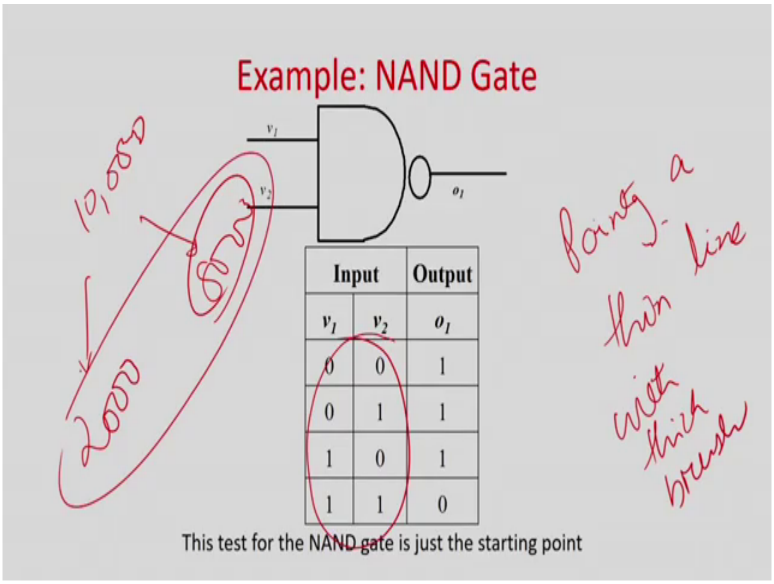
Now, in a smaller die if I think make these line thicks, if I make this gap high then what will happen. There will be restriction in the number of devices going into it. So, maybe I can do 10,000 like this, but if I make these lines very thin and I have a very very middle very thin gap in between these two, then what is going to happen. Then what is going to happen, then maybe I will be able to put more and more devices in this single chip. So, what more chips devices mean more functionality, more functionality means your device will have more kind of functionalities and more wide range of specification. So, your market will be high.

Like for example, you know that Nokia I am just giving an example should not go by render names specifically. But, if you see Nokia had a very good robust mobile phone, but it went down in the market because some other companies like Samsung, Apple and all other industry started giving more functionalities. So, more functionalities means I have put more devices inside a chip. So, the market will go or even you we also try to go and purchase devices which have no more functionalities at lower cost. So, by lower cost means I have to try to put more number of chips oh sorry more number of transistors in a single die. So, that I can give you more functionalities, but again as I told you die size cannot be size cannot be increased because, of some physical limitations.

So, what I will try to do, I will try to make a very thin line and I will try to also make the gap very very thin. So, now a days from 5 micron design today are we are in the nanometer design, 95 65 nanometers. Now, if you try to do it what is the problem, if you make a very thin line doping it may have some kind of gaps in between and if I try to make very thin lines there can be shorts. The idea is that the more clear you want to make a line and more less gaps you want to keep you are going to have defects like shorts and opens, but still why still I have to go by that because, if I give you a very reliable circuit, but if the functionalities very very less nobody will try to purchase it.

Rather what people will like to do is that people will try to take a circuit with more number of functionalities, but can I sell a circuit like this which is having shorts and opens no. So, what I will do? I will manufactured 1,000 devices test all of them, maybe 200 having some defects I will throw them out, 800's I will sell at a higher cost; like this one is correct, this one is wrong. Same thin and thin different I will sell those 800 at a higher price and 200 which are having defects, I will throw them out of the picture. So, in such cases what is going to happen, such cases your market will be rise, because you are having giving more functionalities. So, the economics is driving us. So, the economies is driving you to a situation when we have to or when you have to fabricate more number of device, more number of devices in a single die.

(Refer Slide Time: 45:04)



Example: NAND Gate

Input		Output
v_1	v_2	o_1
0	0	1
0	1	1
1	0	1
1	1	0

This test for the NAND gate is just the starting point

So, I need very very thin lines very thin lines and very thin gaps in between. So, I can put more devices and I can put more functionalities and sell in to the market. More defects; obviously, because making a thin line and little gaps may have to opens and shorts. How will I escape from the market? I will test all of them whatever will have been defects even if this 20 percent, 30 percent no problem I will throw the defective pieces out of the bin. And, I will sell only the good ones to the customer with the increased price; still you are going to make a profit. So, this is market driven not a engineering driven stuff. So, market drives me to make such kind of sophisticated devices.

They have a term called painting, painting a thin line thin line with thick brush; they call it this way painting a thin line with a thick brush. So, of course, you are going to make error. So, what is a brush here? Brush is actually your fabricating equipment. Now, of course, today I manufacturing 65 micron at the some name I have to go out to even lower 35 micron, but maybe I am just having 2 months, 2 years to change this from higher micron to a lower micron; that is the gap will be less lines will be thinner. So, I request some time to a put R and D and do lot of innovations to make such a complicated fabrication machine. But, the time given to me is less 2 years may not be enough time to do it so, my mission is also half completed in design.

So, what happens I will make some kind of errors like open shorts and, but still I want to do go by that because, I want to have my market when I will throw out the elements which are defective and sell the correct ones with a higher price and of the market. So, this is driven test or such kind of manufacturing trained in VLSI is a market driven phenomena or economic given phenomena. For example, say 65 nanometer had lot of defects today, but after 10 years maybe if I am perusing the 65 nanometer technology then I will have a very robust manufacturing unit by that time where there will be very very less defects.

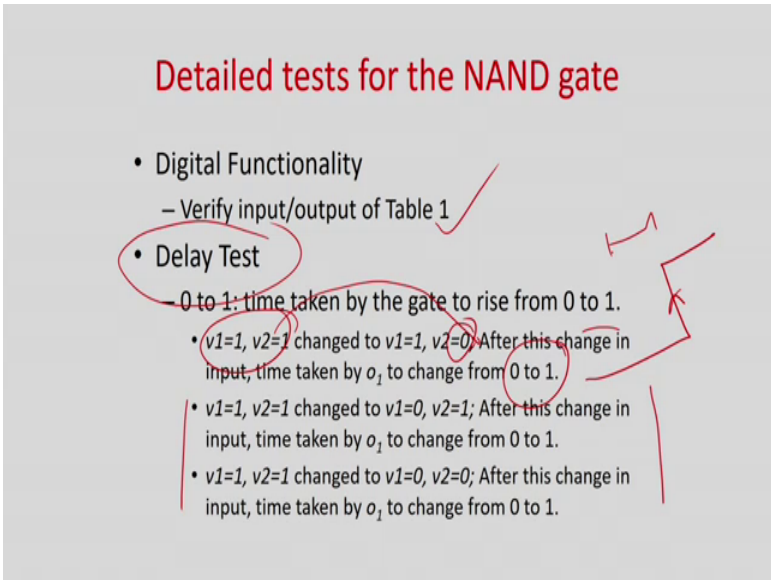
But, 10 years down the line 65 nanometer will nobody will purchase from you; there will be first further lower technologies coming into picture. So, the time you are giving me to make a manufacturing unit is not high so, that I can give you a very reliable designs; so because I am also a designer of the; my manufacturing unit. So, my I will tell you that the time pressure you have giving me I will give a 65 nanometer or 35 nanometer design, but there will be lot of defects. So, defects may be as high as 30 percent or 40 percent, still it is not a problem you are test all the devices whichever having defects you throw them out. And, sell only the proper once to your customers of course, you have to note that this point I cannot sell a defective part to a customer.

So, this will be a gross problem. So, say 10,000 device has been fabricated, I have to test all of them physically. So, may be out of them 200 or 2,000 are having defects, 8,000 are normal. So, these normal 8,000 sell to the customer with a higher price, but I have put lot of functionalities. I will go by this trend rather than not going by a system where the micron is less and the technologies built old, but out of 10,000 9,999 devices are proper. I will not go by this train because that will make me lose out in the market.

As I have story I have been tolding you know that about Nokia and high advanced devices with lot of functionalities like other mobile phones like Samsung, all the other companies like your basically me Jio me and all those phones which are more functionalities. But, somehow if you use have you might have use all of them reliabilities slightly less, but again as I told you so, this part is very very important. You cannot send a defective parts to a customer, you have to test it, throw it and basically sell at a higher price. So, that is and you might have appreciated by now, that I know how to build a cycle over 200 years. I know how to be build a electrical iron over 70 years, technologies not much change.

So, therefore, there is a very very less chance of having physical errors at the manufacturing stage devices in (Refer Time: 48:54) classical literature right. So, that is why testing of hardware is a very very important area in case of any kind of embedded system design or any kind for that natural any kind of VLSI industry; extremely important that you have to always go for you have to think a lot about testing. And, which is once you is not the case in case of classical design ok. With this motivation let us start about your basics of testing. So, in this case you have to give all these 4 variables as inputs and if everything is proper your circuit is operating fine.

(Refer Slide Time: 49:27)

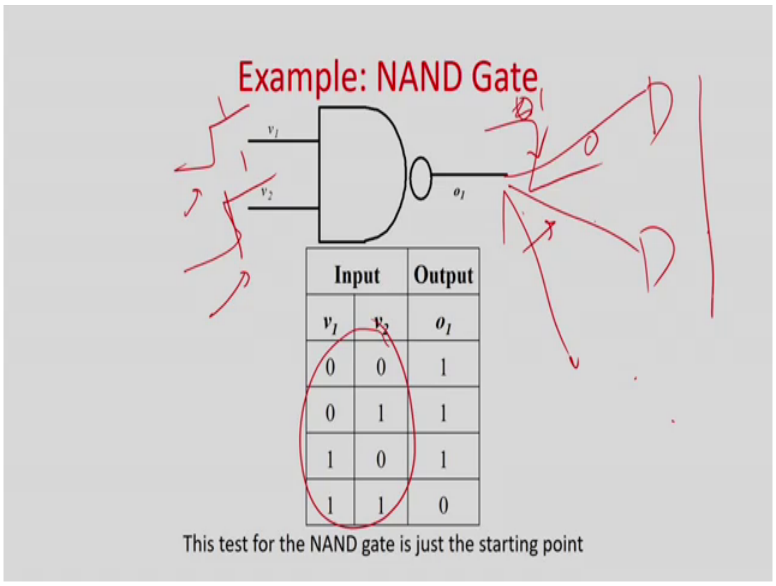


Detailed tests for the NAND gate

- Digital Functionality
  - Verify input/output of Table 1
- Delay Test
  - 0 to 1: time taken by the gate to rise from 0 to 1.
    - *v1=1, v2=1* changed to *v1=1, v2=0*; After this change in input, time taken by o_1 to change from 0 to 1.
    - *v1=1, v2=1* changed to *v1=0, v2=1*; After this change in input, time taken by o_1 to change from 0 to 1.
    - *v1=1, v2=1* changed to *v1=0, v2=0*; After this change in input, time taken by o_1 to change from 0 to 1.

Now, so that is there is actually the basic functionality testing. Now, in case of embedded system a very important part is deadline.

(Refer Slide Time: 49:37)



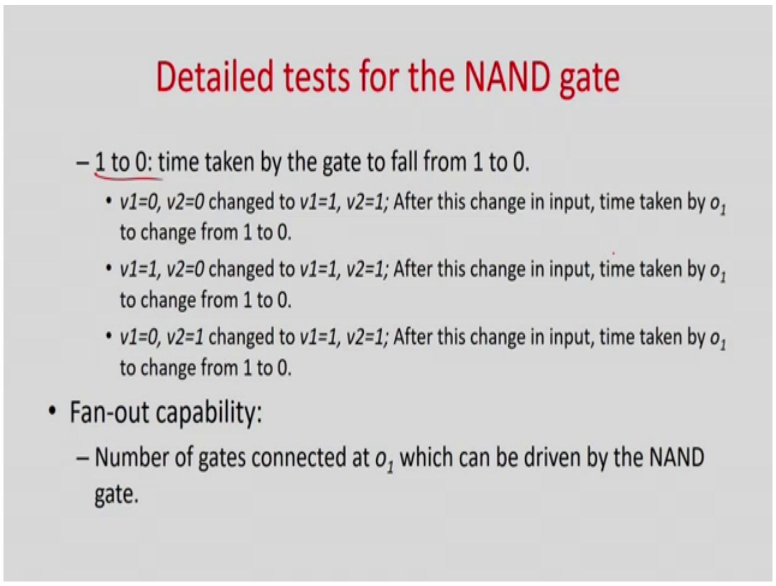
Example: NAND Gate

Input		Output
v_1	v_2	o_1
0	0	1
0	1	1
1	0	1
1	1	0

This test for the NAND gate is just the starting point

That if some example I am giving something like this and then maybe so, basically and then again I have a somewhere circuit like this. So, the output maybe falling like this, both 1 1 the answer will be from 0 to sorry from 1 to 0. So, if after this, after you have applied this both what to the time taken to may for this fault, that is actually called real time. So, you cannot give a very high time for the NAND gate to respond, if the time taken by the NAND gate to respond even if the answer is correct this is a lot of delay; means your embedded system will not be able to give the solution in real time. So, that is actually called the delay time, delay test from 0 to 1 that is time taken by to by the gate to rise from 0 to 1. And, similarly there will be a fall test from 1 to 0.

(Refer Slide Time: 50:15)



Detailed tests for the NAND gate

– 1 to 0: time taken by the gate to fall from 1 to 0.

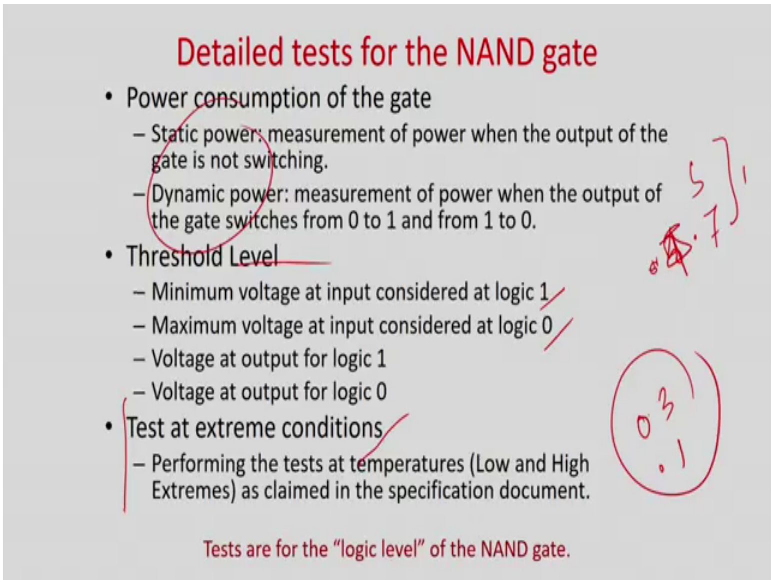
- *v1=0, v2=0* changed to *v1=1, v2=1;* After this change in input, time taken by o_1 to change from 1 to 0.
- *v1=1, v2=0* changed to *v1=1, v2=1;* After this change in input, time taken by o_1 to change from 1 to 0.
- *v1=0, v2=1* changed to *v1=1, v2=1;* After this change in input, time taken by o_1 to change from 1 to 0.

- Fan-out capability:
  – Number of gates connected at o_1 which can be driven by the NAND gate.

So, basically what you can do take these two inputs as one, you change one value to 0. So, output will be changing from 0 to 1 and you can find out what is the rise time. So, the pattern will be we will have a 1 1 over here and then quickly you switch v 2 v 2 from 1 to 0 to 1 to 0 and output will be changing like this and the delay you have to measure. So, there are many other ways of doing the same thing which is written over here. So, any of them you can apply and you want to see whether the riser is proper or not. You can also say and I have to do it exhaustive testing because, I want to answer you that in all the probably in all in all chances of combination of inputs what will be the rise time. And, I have to ensure that we rise in the in the predefined threshold.

So, then my embedded system can give the answers in a real time. So, this is called the delay test rise test and fall test. There is also something called fan out test. How many gates I can connect to this? There is something called a fan out. So, we have to guarantee, but I can give I can drive 10 gates out, I can give 20 gates or I can do only 4 gates as the output of it or I can drive like that. So, all this information I have to provide and how can I do that. So, I may have designed it to drive say 4 AND gates, but it is not driving it properly maybe because off is not able to drive properly. There is something some delay, delay faults are happening or some other types of faults are happening. So, you have to also go for testing of all, this is I called the testing of the fan out capability.

(Refer Slide Time: 51:38)



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.

Then something called power. So, I am not going to the into the details of all these test right now, whenever required in course of time will be giving; I am just this introductory lecture. So, trying to put you in basically what will be basic stuff required for testing and the motivation we are building. So, that is called fan out testing maybe there is something called power. Nowadays all the embedded systems are power or by batteries. So, you have to always determine, but what is the power consumption of this gate. So, basically something called static power and there is something called dynamic power.

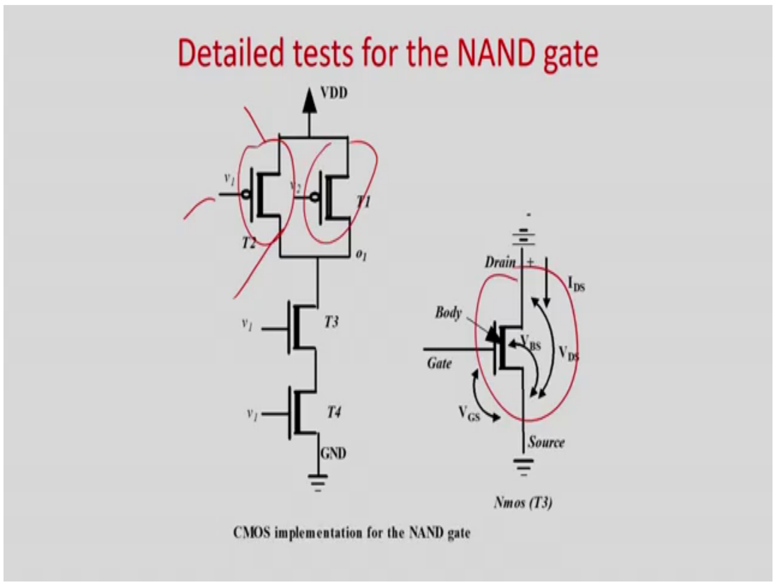
Static power is measuring the power when the output of the circuit is not switching, dynamic power is when the inputs and outputs are switching. So, you have to measure the power and you have to find out whether it is meeting the predefined requirement or not. Otherwise if it is over shooting, then whatever guarantee we have say that this their mobile phone or device will work for 10 hours at the non talk term, 3 hours at the talk time. So, you may not be able to meet these guarantees. So, therefore, you have to also do power consumption test. There is something called threshold test, threshold test means basically like what voltage you are considering as 1 and what voltage are you are considering as 0.

So, maybe your considering is 0.3 to 1 as your 1 and maybe 0.7 to sorry point so, maybe 4.7 to 5 volts have logic 1; so whether it is meeting those thresholds or not. So, those testings also you have to do that, if it is the output should be expected as 1; so, the output voltage is between 4.7 and 5. Similarly, if the output of the gate is 0, it should be within this voltage levels. Then there is something called testing at extreme temperatures, your embedded system unit may be in the car engine. So, the engine temperature may be become high. So, within that high temperature also whether this all the parameters happening properly or not that is actually called testing at extreme condition, temperature is one of them. So, you can also test at that level.

Now, I can again make this is very very large, infinitely large I can make just like the electrical iron; similarly for the AND gate you can do it. Of course, electrical iron the test whatever I have told you nobody will do anything extra then just putting it into the power and see whether operating or not. And, maybe slightly we will check the thermal state, but in case of this gates at least we have to do functionality, we have to do delay, we have to do fan out, we have to do power and threshold. Maybe, if we can we can drop the extreme level these things we can drop because, more test you put more price you have to pay. But the number of tests you have to do for a NAND gate or a electronic circuit will be much higher than a classical system.

And, I can make the list as high as possible if you have time and if you have money because finally, we have also gain profit. Nobody will purchase a mobile phone with 10 lakh rupees which is say that you will always guarantee to operate for next 10 years; nobody likes a device on embedded system like that. So, you have to make a reasonable amount of test. So, we will give a reasonable performance at the reasonable cost. One interesting thing I should tell you is that basically embedded system industry mainly runs by your test economics or the economics the value and money and the market. Rather than giving a very very reliable engineering solution which may not be feasible in the market; in not only about embedded system for most of the engineering disciplines.

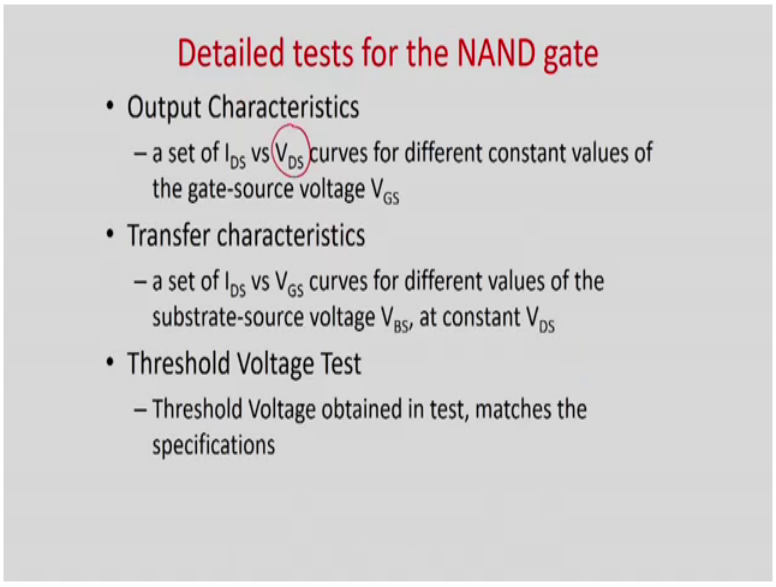
(Refer Slide Time: 54:41)


Detailed tests for the NAND gate

CMOS implementation for the NAND gate

Now, finally, as I told you may be all the silicon all the NAND all the any kind of gates will be finally, manufactured in terms of transistors. So, you can also say that apart from the gate level and also test at the transistor levels. So, this is the CMOS implementation of the NAND gate, I am not going to go to the details. But, the now I can go for the testing and the each of the transistor levels.

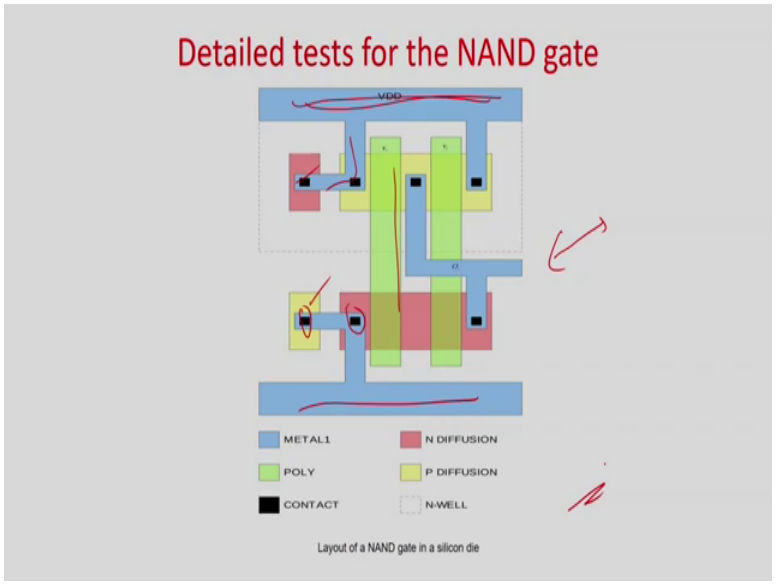
(Refer Slide Time: 55:02)


Detailed tests for the NAND gate

- Output Characteristics
  - a set of I_{DS} vs V_{DS} curves for different constant values of the gate-source voltage V_{GS}
- Transfer characteristics
  - a set of I_{DS} vs V_{GS} curves for different values of the substrate-source voltage V_{BS}, at constant V_{DS}
- Threshold Voltage Test
  - Threshold Voltage obtained in test, matches the specifications

That is output characters, I DS by means drain source current by the volt drain to source, transfer characteristic threshold voltage characteristics. So, all these tests are case cases are basically at the level of transistors. Now, we have to go internally and we have to tape out such points and do test at this level; more reliability will have, but more sophistication it will be.

(Refer Slide Time: 55:24)



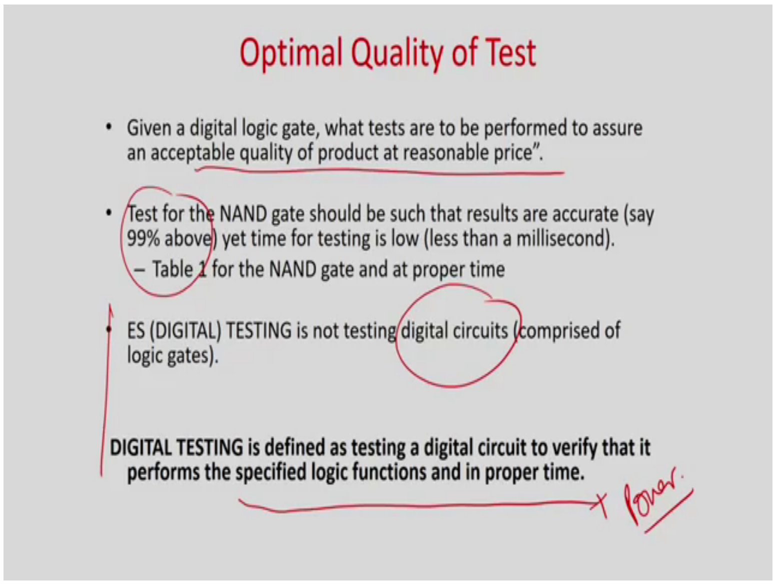
Detailed tests for the NAND gate

Layout of a NAND gate in a silicon die

Finally it is actually the layout of the NAND gate. Now, people will as I was saying say this may be a copper, this is copper this may be a copper this may be silicon, this may be a wire. So, metal all the names you can see this is a poly. So now, I want to say I want to find out whether this metal is properly laid out or not. I can do the testing you require a probe or to do it whether I want to see whether this context are properly there or not I have to do some kind of probe base test. Means, testing a NAND gate itself will take years and years and manufacturing 1 NAND gate cost maybe millions of dollars. So, that is what I am going to tell you is that even for the circuits, even the circuits have lot of issues in case of manufacturing and defects still you cannot go to at this level for each of the gates.

You are you will be actually they will be no monitory benefit to go at this level, then basically at one point we should stop; so, that we can give you a reliable solution. Generally we stop basically as I told you we will go for functionality test, we go for delay test, basically we go for power consumption test. Mainly, these 3 states we do it elaborately; functionality of course, we have to do, delay because embedded system has to delay deadlines has to be made and basically power consumption. Mostly such tests we have to do for all the circuits at a certain level of certain level of regard because, beyond that you may be selling off a defective part to a customer. Other things we should do even at this level we should do, but the problem is that you will not be able to have any monetary gain out of it.

(Refer Slide Time: 56:51)



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
- ES (DIGITAL) TESTING is not testing digital circuits (comprised of logic gates).

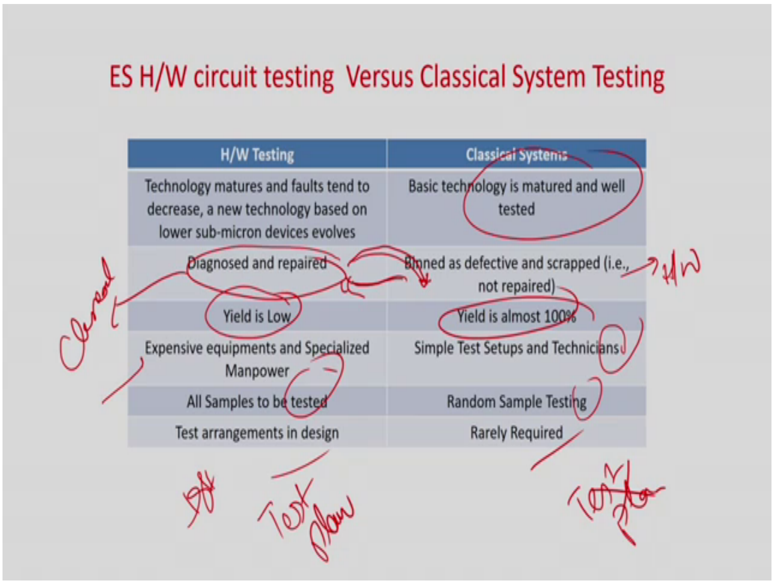
DIGITAL TESTING is defined as testing a digital circuit to verify that it performs the specified logic functions and in proper time. + Power.

So, that is actually call the optimal quality of test. So, given a digital logic gate; what tester to be performed to ensure an acceptable quality of product at a reasonable price. So, as I told you an example functionality delay and power.

So, basically it should be accuracy should be very high, maybe 99 percent above. And, embedded system digital testing is not test the digital circuits comprising of NAND gates. Digital test this is a very interesting statement, digital testing is not testing digital circuits rather digital testing is defined as testing a digital circuit to verify that is performed this specific logic functions and at proper time plus power. I should add the word power. That means if you say that I want to test a digital circuit. So, digital circuit testing means comprising all, this one, this one transistor level and finally, at the fabrication level at the device level.

So, that will actually says that I am testing a digital circuit because digital circuit finally, that the transistor level. But in fact, that as I told you is practically impossible. So, rather what is testing here means you take a gate like this and you say that whether it is giving the proper outputs at the proper time. So, that was actually classical definition of test like digital testing is define at testing a digital circuit to verify that it performs the specific logic functions in proper time. This is temporal correctness plus we are nowadays adding power, because in case of embedded system power is also becoming a huge constraint. So, this is actually I think this line is telling you an idea about what is testing, how rigorous it can be, but for a practical economical feasibility how much you should apply for a embedded system hardware.

(Refer Slide Time: 58:32)



ES H/W circuit testing Versus Classical System Testing

H/W Testing	Classical Systems
Technology matures and faults tend to decrease, a new technology based on lower sub-micron devices evolves	Basic technology is matured and well tested
Diagnosed and repaired	Binned as defective and scrapped (i.e., not repaired)
Yield is Low	Yield is almost 100%
Expensive equipments and Specialized Manpower	Simple Test Setups and Technicians
All Samples to be tested	Random Sample Testing
Test arrangements in design	Rarely Required

H/W

Test plan

So, this slides basically compares the classical system testing versus VLSI testing I have been telling you. So, many time, but just let us look at it. Hardware testing that is VLSI technology matures and faults tend to decrease and a new technology evolves for a lower sub-micron designs. As I told now 65 is a good technology, give me 10 years I will give a very robust manufacturing unit of 65 nanometer. But, by that by that 10 years another must complicated complication technology will be coming into picture. And, there you have lot of defects because I am not ready with that manufacturing unit as it is, but in terms of classical systems basic technology is matured and well tested. So, therefore in classical system faults rarely occur and therefore, testing is not a very major challenge in classical systems.

So, in case of so there is a slightly I am just sorry this should be coming over here and this should be coming over a slightly a mistake. So, basically in case of classical systems there diagnosed then repair; I mean this statement should come over here and this statement should go over here. So, but in case of so, in case of classical system we all know that basically there diagnosed and repair. If there is a problem in the electric error you can repair it in, but in case of hardware they are binned and effective and scrapped and not at all repair. So, this part actually corresponds to hardware, slight mistake in many table and this parts corresponds to classical right.

So, this one is classical so, basically in classical your diagnose and repair, but in case of VLSI we just throw them out because there is no way of repairing it. Hardware testing yield is low the reason already told, here yield is almost 100s 100 percent. Very expensive equipments and specialized manpower is required to the hardware testing because, you have to put voltage, current, you have to open up the chips sometimes put robusts. So, this is a very very expensive procedure. Classical system very simple even the roadside vendor can twit for you, here all samples has to be tested because if you have if somebody tolds tells, if within 100 some 10 chips will be faulty or 20 chips will be faulty you have to test all of them.

But, in case of classical as the reliability is very very high you just take do some random sample testing maybe 10,000 units are going out, you just take 1 and 2. Because, you know that mainly if there is if there is no gross abnormality in the supply chain everything will be proper. So, test managements are required in design here is rarely now one. This is actually called test plan which will be mainly in this class that is test plan sorry this is also the test plan, but not required.

So, test plan means you have to make lot of arrangements to test this circuit because, this is very sophisticated. A circuit we have 10,000 gates from the simple NAND gate example you could have find out the testing is not very simple. But, it is a very used circuit with tens of thousands of gate then you have to plan beforehand what you will apply, how will you apply, what will you measure; so, that you can do the testing in a reasonable amount of time.

But, in case of classical system we really need to do any planning because you know that whatever we are generating without any gross abnormality everything will be proper. So, the test is very very simple. So, we never think about the planning itself, but in case of hardware things is very very different. So, I think I am able to give you an idea and why is VLSI testing or embedded system testing is more involved or more required or more important than compared to any classical system of engineering. And, in this part also I have told you I think I have been able to tell you the difference between verification and test in case of embedded systems.

With this motivation basically next class we will try to give you the algorithm of test planning. Because, whenever you think about testing again I am repeating it in case of VLSI or embedded system testing means putting the hardware connecting and driving the values and getting it. But what really means motivates us as computer engineers or electrical engineers are how will you plan your test that given a big circuit, how will you make the official test patterns; so, that you can do the test efficiently and a very less amount of time. Like I have told you and I want correct functionality correct time and power, other things I am not going to measure. Because, if I do it cost will be very high and reliability will slightly increase. So, how do I have such insights or how test plans that is actually the job of the test engineers.

So, in this module basically we will be learning about test planning of embedded systems. So, with this we come to the end of this introductory lecture. In tomorrow's lecture what we are going to do is that, we are going to take real examples of circuits and we will find out algorithms how to test the circuits.

Thank you.