

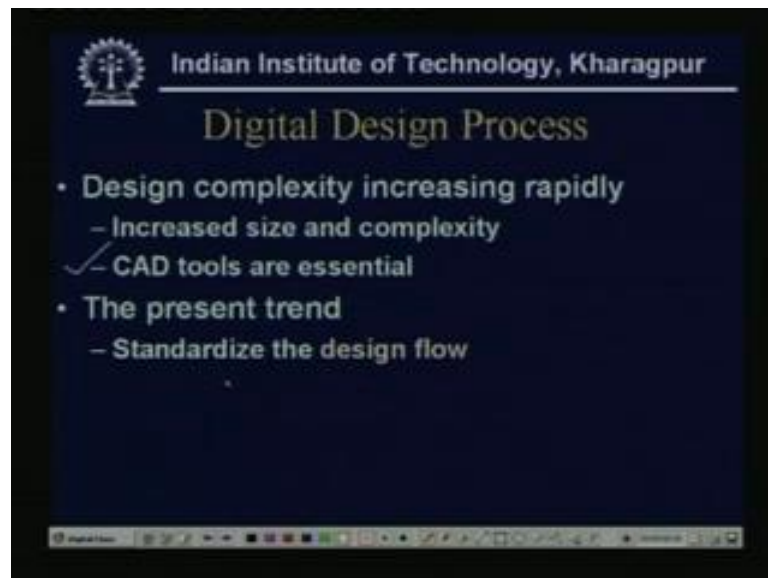
Digital System Design
Prof. D. Roychoudhury
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 40
Digital System Design Current State of the Art

In the last class, we have read how one micro-programmed CPU can be designed. Already we have read that how the combinational modules, the sequential modules can be designed and the one CPU and the part of the CPU that arithmetic logical unit, the control unit all these things can be designed. Now, with the advancement of technology, the digital systems has it is growing enormously and the complexity of the system is also large.

So, now all the design procedure is automated. So, today we will discuss what so far, we have read, how this can be automated using the CAD tools and what is the current state of the art of designing the digital systems.

(Refer Slide Time: 01:51)

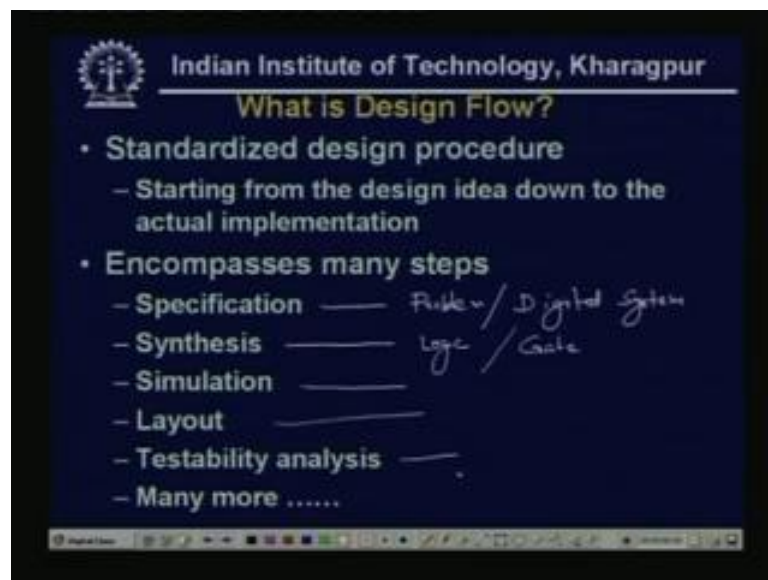


Now, design complexity is increasing rapidly with the increased size and complexity. Now so far in this class, we discussed the different design procedure, when the system is in moderate size then, we can utilize this all these things or manually we can apply this techniques. But, now with the advancement of technology, the digital system grows enormously and so, for human being it is not at all possible to do the design manually.

So, for this the CAD tools are essential but, one thing to be remembered that all these CAD tools, they utilize this techniques or actually they implements the design procedure we have read. The design of ALU, the design of the control unit for the CPU design that actually the CAD tools are designed, but as of now, that another thing is that IC for the digital IC, that Integrated Circuit.

Actually thus within a silicon, the within a small area millions of gates are fabricated, so though the design procedure is same but, we need a design flow for these CAD tools. Now, the present trend is this standardized the design flow, so first we see, what do we mean by this design flow of digital ICs or a complex digital circuits.

(Refer Slide Time: 03:45)



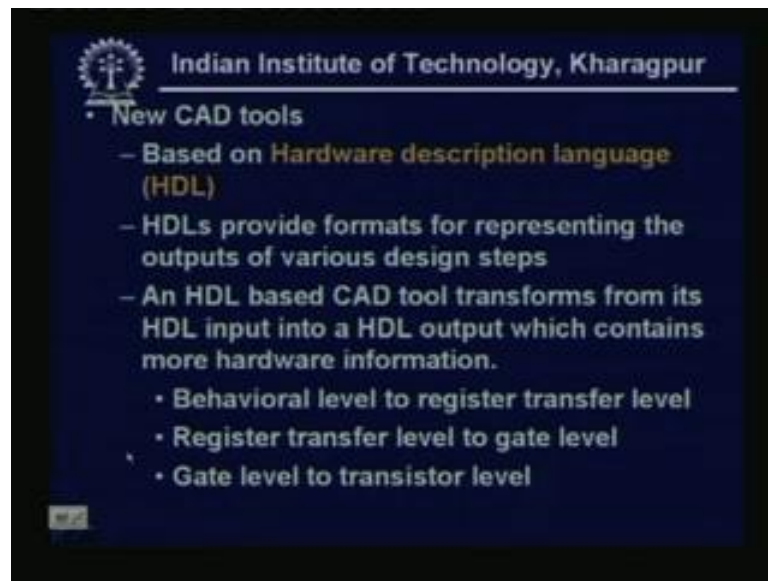
So, starting from the design idea down to actual implementation or in the one, we can tell that given a problem which we want to design a digital system or given a digital system to be implemented on a silicon, the step by step procedure until we get the whole design on the silicon is called the design flow. Now, they encompasses many steps first is the problem specification this is actually the problem or we can tell the, the specification of the digital system which is to be implemented the digital system.

Then, we have to synthesis it, so given the problem how we can get the logic synthesis part that means, the gate using the gate or there can be different equipments. The simulation then, whether it is behaving correctly or not whether the system is behaving

correctly or not, so that is for the simulation, then the layout. So, actually on the silicon how it can, how we can get the implementation.

Then, the testability analysis that, when whether it is a running accurately or working correctly that for that purposes, that we have to do the testing and many other things to be done. So, mainly these are the procedures and this step by step procedure is called the design flow.

(Refer Slide Time: 06:06)



Now, some new CAD tools are available for this design flow, actually not only one CAD tool is sufficient for this whole design procedure. The steps even for each step, a different CAD tool is needed, now first thing is that, given a system to be implemented a digital system or digital design. First I have to specify or I the, a system means this is a hardware, so when I and CAD tool is a software.

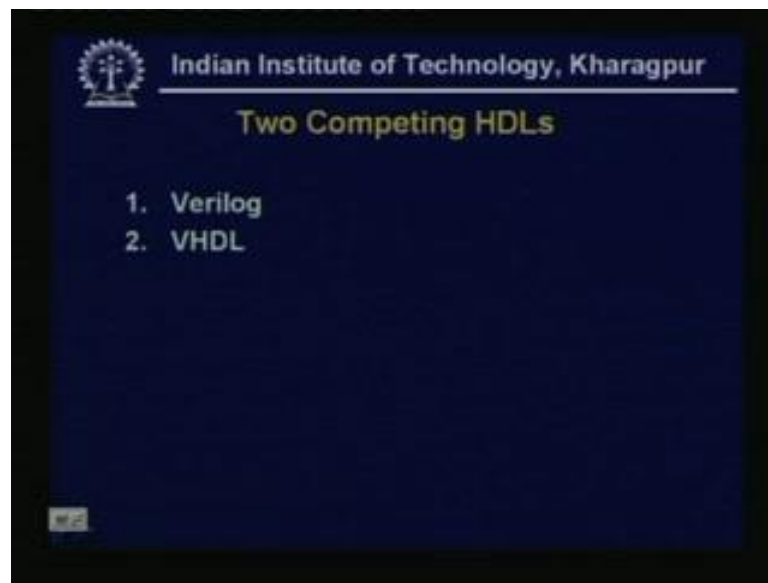
So, I want to interface between this hardware and software and for these purposes one Hardware Description Language called the HDL has been formed. So, HDLs provide formats for representing the outputs of various design steps, and HDL based CAD tool transforms from its HDL input into HDL output which contains more hardware information.

Why more hardware information because, my problem specification or the specification of the digital system is given. From there, we are creating or we are developing a

program or the how using hardware description language which is software, but my ultimate aim is to get a hardware on silicon. Now, using this design flow or these steps, step by step procedure as we go from the, or a as we are running the steps, then more and more we are going nearer to the hardware.

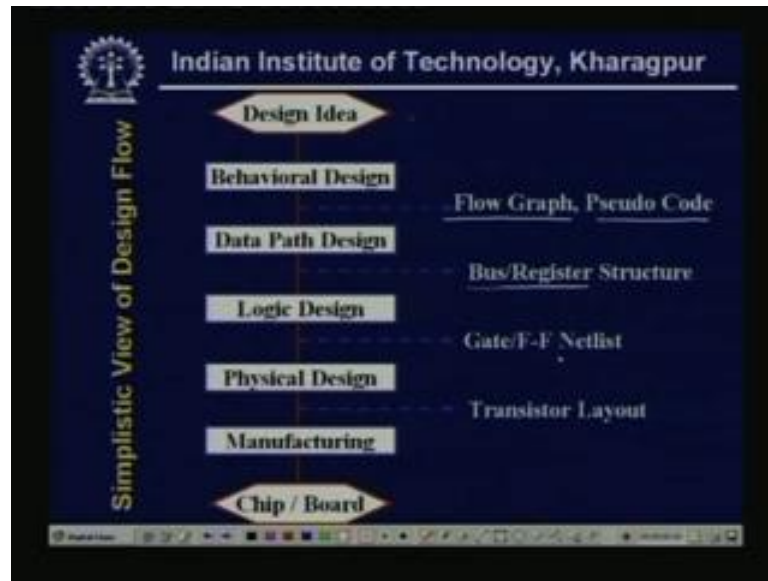
So, this hardware information is in the form of behavioral level to register transfer level, register transfer level to gate level and gate level to transistor level, ultimately on the silicon, the transistors are fabricated.

(Refer Slide Time: 08:42)



Now, the two competing HDLs are in market now, though earlier even 10 years before the different organization according to their requirement, they developed their own languages, but now as the thing has been standardized. So, only the two standard HDLs people are using, one is called the verilog and other is the VHDL.

(Refer Slide Time: 09:16)



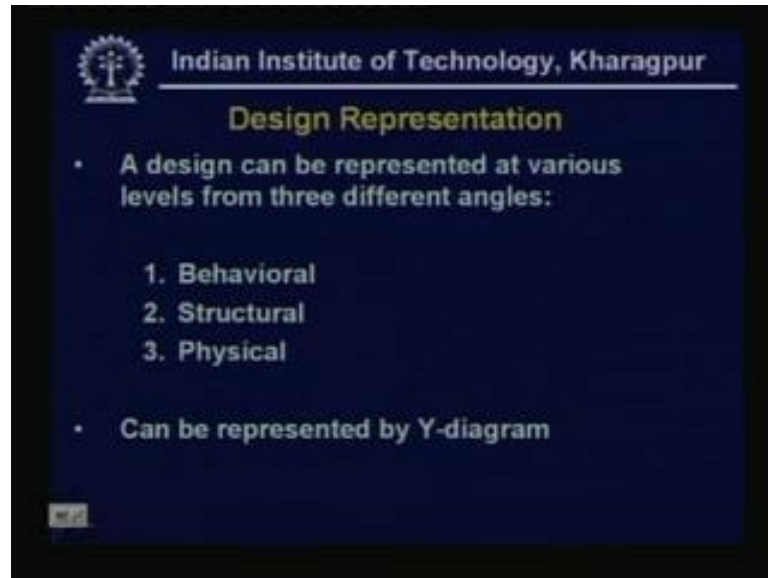
Now, although we give a very simple view of design flow, see this is a this is design idea means, the given the system to the implemented that the designer must have the idea that or a some abstract level design idea that how it can be implemented in hardware. So, that we are calling the design idea then, this whole design its behavior, the behavioral design is represented, this can be a how this can be represented. So this can be a flow graph or this can be a pseudo code.

Now, from this behavioral design the data path design has been evaluated, so these the output of data path design is actually the whole design realized by the bus transistor, bus and the transistor register. So, already we have read that how the register to register transfer. So, the whole design as if the using a load store type of structure the register to register transfer using the bus, then the data path is designed.

Then, the logic design, so logic now the design is synthesized using the gate flip flop and the different type of logical gates. And, normally they are, it is called the netlist because, this is the network of different gates, flip flops, etc. Now, after this logic design we call this is a physical design, so physical design mean once the logic, correct logic we have synthesized then, now my next step is how we can get this implemented on a silicon, this we are calling the physical design.

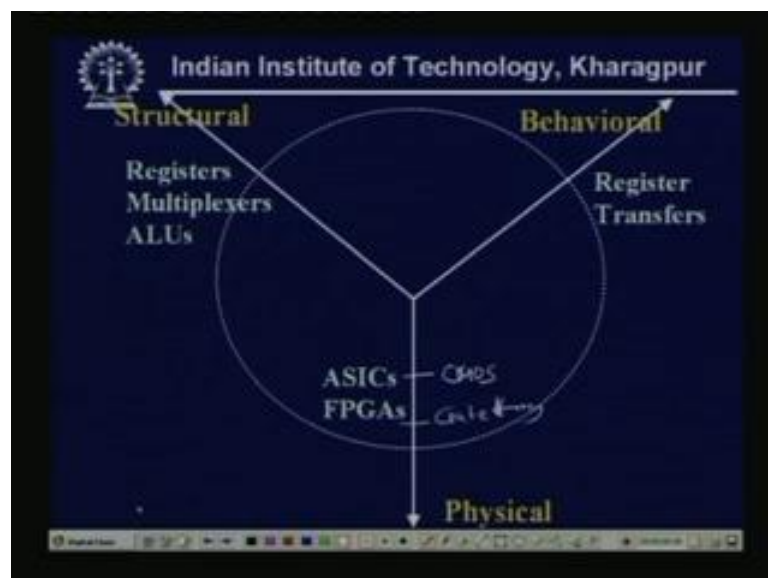
Actually there are several steps later we see in details the, what are the different steps of the physical design, then it is manufactured and ultimately we get the chip or board. Here, chip is the actually integrated circuit chip.

(Refer Slide Time: 12:13)



Now, first thing we see the design idea, how that can be represented we call the design representation, so design can be represented at various levels from three different angles. It can be behavioral, it can be structural, it can be physical, now this three can be represented by the very popular Y diagram.

(Refer Slide Time: 12:40)

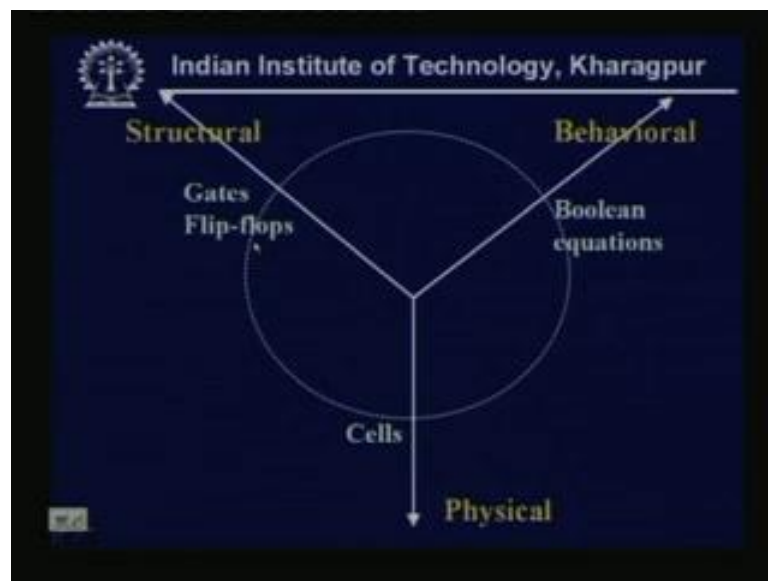


See, this is a Y diagram and the three arms that we are calling that, one is called as if one is structural, another is behavioral and one is physical. So, actually we want the physical representation, see structural is as if the, the whole design I have user have enough idea of the design and then the design, how they can be represented or realized by using processor memory. Obviously, the interconnection is the bus, so this is nothing but, some network of the memory processor of these things, so these are called the structural.

Behavioral means, how the design behaves; that means, how the representation of its behavior by flowchart, what if this input is given, what will be the output and how this output can be generated. So, this is the step by step procedure is called algorithm or flowchart, now ultimately we want the physical thing and this is the PCBs or MCMs this is called the physical thing. Now, the same thing can represent that structural can be registers, multiplexers, ALUs because are design structure or the hardware design that can be synthesized using register multiplexers ALUs.

Behavioral that can be register transfers because, it is the algorithm or the flow chart, so that we also, we can represent by using register transfers. And we get the physical as ASICs, so all FPGAs; that means, either by some CMOS technology or the gate array technology the field programmable gate arrays.

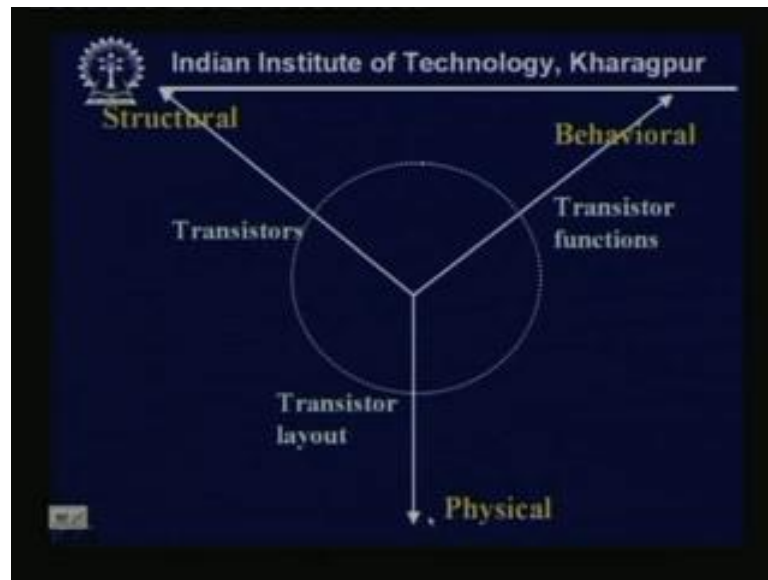
(Refer Slide Time: 14:57)



So, structures even the structures can be gates or flip flops ultimately, it should be gate and then that gate can be realized by transistors. And behavioral can be Boolean by

Boolean equations because the set of Boolean equations can also represents the behavior of the digital system that already we have read. So, now it can be map to the cells ultimately, when it is physically designed.

(Refer Slide Time: 15:27)



Now, the structural thing can be realized by transistors, again the behavioral things can be the transistor functions and this can be transistor layout. So, ultimately we want the implementation of transistors on silicon and that is my physical thing.

(Refer Slide Time: 15:52)

Indian Institute of Technology, Kharagpur

Behavioral Representation

- Specifies how a particular design should respond to a given set of inputs.
- May be specified by
 - Boolean equations ✓
 - Tables of input and output values ✓
 - Algorithms written in standard HLL like C ✓
 - Algorithms written in special HDL like Verilog ✓

Now, see how this behavioral representation can be done, so behavioral is representation it specifies how a particular design should respond with given set of inputs, its behavior of a system. So, this may be specified by Boolean equations, already we have seen that given Boolean equations, how Karnaugh map can be kernel map synthesize the actual and or design and different type of other techniques we have seen that from Boolean equations to realize the actual hardware circuit.

Then tables of input and output values this can be simply represented by a table, if these are the inputs this can be the output just like the truth table of the circuits. Algorithms written in standard high level language like C or some pseudo code, algorithms written in special hardware description language like verilog or VHDL. So, in this different ways, I can represent my behavior of the system.

(Refer Slide Time: 17:19)

Indian Institute of Technology, Kharagpur

Behavioral representation :: example

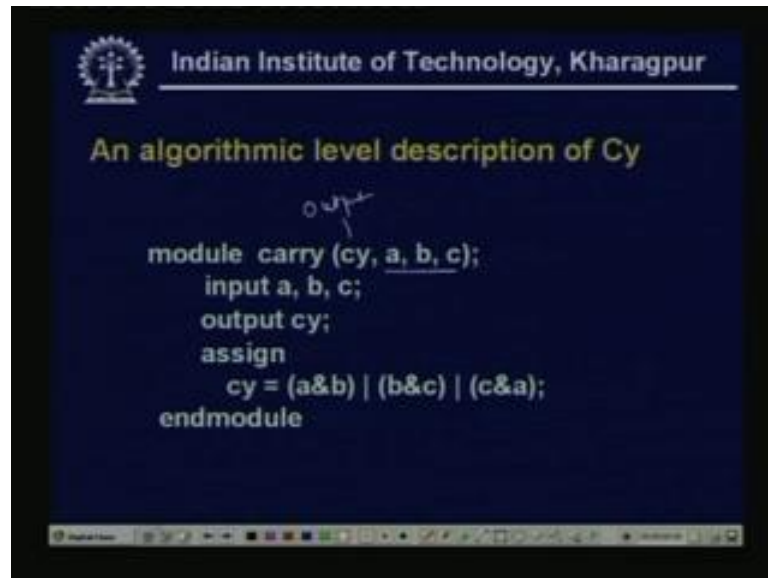
- A n-bit adder is constructed by cascading n 1-bit adders.
 - A 1-bit adder has
 - two operand inputs A and B
 - a carry input C
 - a carry output Cy
 - a sum output S

$$S = A.B'.C' + A'.B'.C + A'.B.C' + A.B.C$$

$$Cy = A.B + A.C + B.C$$

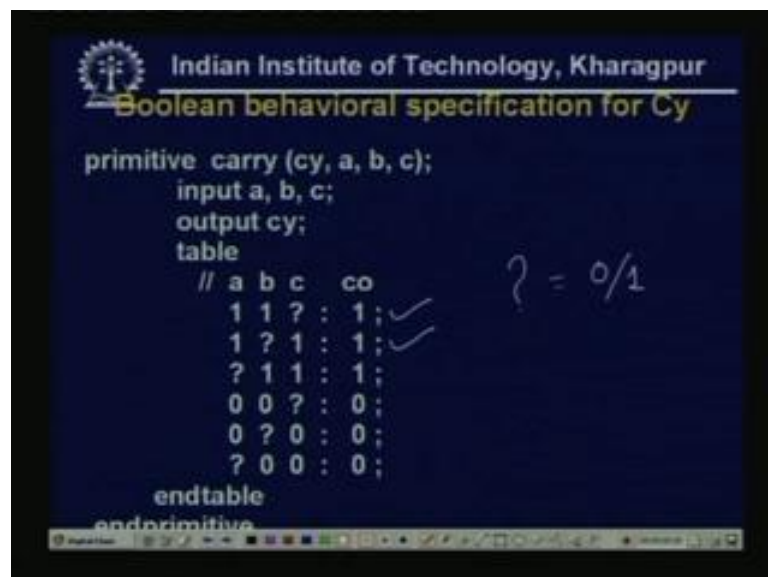
Now, behavioral representation we give one small example, we take one n bit adder is constructed by cascading one bit adders. Now, a 1 bit adder has two operand inputs A and B a carry input C a carry output C y a sum output S, these are the two outputs S and C y and three inputs A B C. Now, already we know that some expression is A B dash C dash, A dash B dash C, plus A dash B C dash, plus A B C and carry output is A B plus A C plus B C. So, these are the two Boolean equations can represent the behavior of adder, 1 bit adder.

(Refer Slide Time: 18:09)



Now, if it is a algorithmic level description then, we can write in this way, say I am writing a carry module, say module carry the three inputs a b c and one output, one output carry, so input a b c output carry C y. Now, assign C y is a and b ab plus bc plus ca, so a and b, b and c, c and a, they are OR, so this is a end module. So, this is an pseudo code or algorithmic level description.

(Refer Slide Time: 18:53)

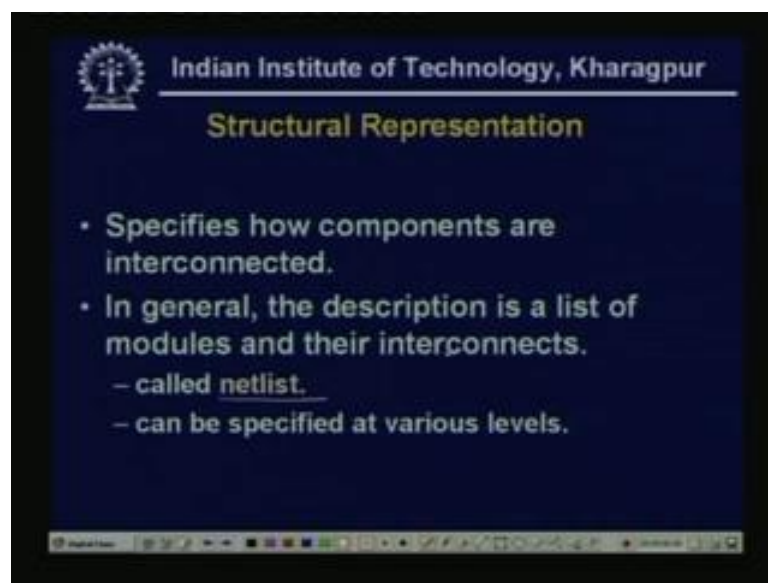


Now, if it is a table form, we can give the just simple input output in a table, so primitive carry again, carry y is the c y is the carry output a b c are the three inputs and that are

declared now, this can be in table form a b c, c 0. So, the truth table; that means, all possible values of a b c are given, here question mark means, that can be question mark means that can be 0 that can 0 or 1 anything. So, if it is 1 1 0 or 1 1 1, the carry output is 1, if it is 1 1 1 or 1 0 1 carry output is 1.

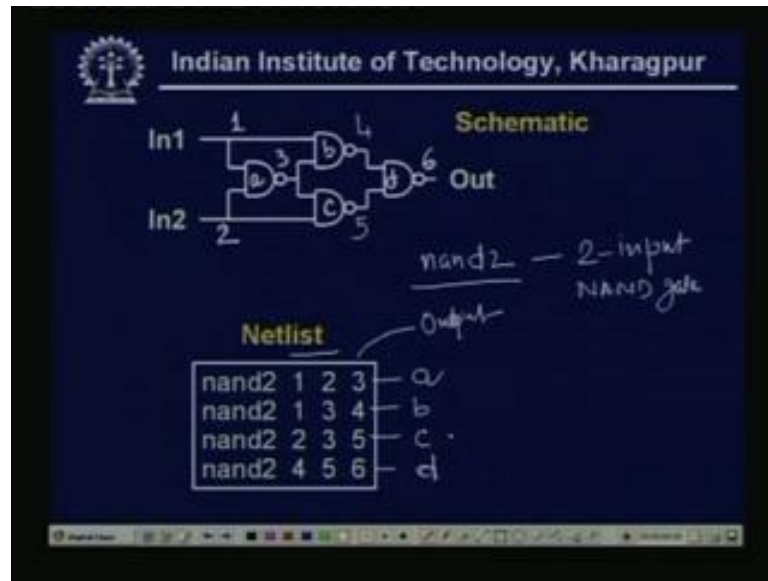
So, actually for all possible values of the inputs a b c what will be the carry output that is given in the table. So, it is in the, the behavioral specification in table form and that is represented in a algorithmic way.

(Refer Slide Time: 20:01)



Now, is a structural representation, now if the designer have enough idea about the design, he has the idea of the structures then, it specifies how components are interconnected. Note actually, the behavioral level or algorithmic level are again some in abstract level, structures mean we can tell this is the second level of design. And if the designer has is expert, so he has that idea and he can even represent the design in its structural representation. So, in general the description is a list of modules and their interconnects and this is called the netlist, so this can be specified at various levels.

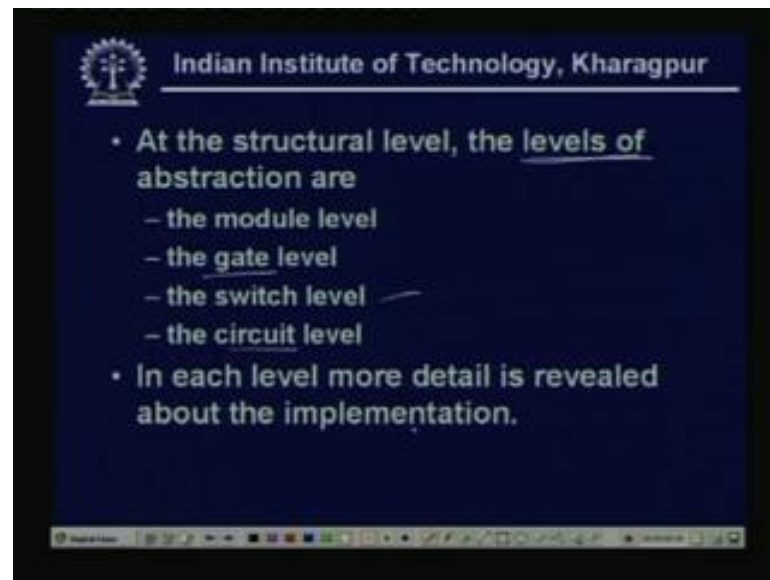
(Refer Slide Time: 20:58)



We take one example, say we take one simple circuit, this is a 4 NAND gate, I give some line number, so this is a input 1 and that line is represented as a 1. Similarly, input 2, input 2 is 2, this intermediate line is 3, some numbering I am giving this is 4 this is 5 and the output line is 6. So, the netlist will be the, how they are interconnected that information should be there, so all are 2 input NAND gate and that can be represented as a NAND 2.

So, NAND 2 means, it is a 2 input NAND gate, now NAND 2, 1 2 3 means, this 1 2 are the inputs and these are the, this is the output line. So, actually this represents, this is the NAND gate a, so this is the NAND gate a, now 1 3 4, so 1 3 4 means, this is the 1 3 and 4; that means, this is my NAND gate b, 2 3 5 means this one, these are the 2 inputs 2 and 3 lines and 5 is the output. So, this is c and 4 5 6, so this is my d, so this, this is the netlist of this small circuits.

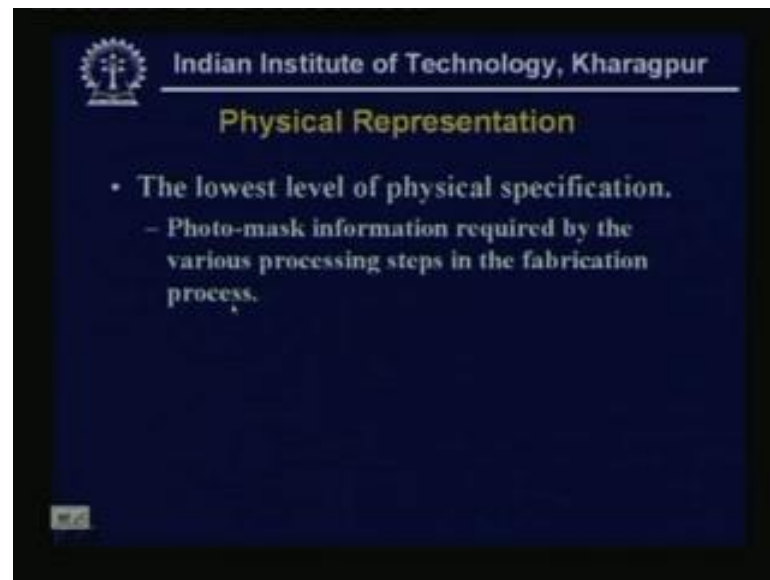
(Refer Slide Time: 23:04)



Now, at the structural level the levels of abstraction are module level, gate level, switch level or circuit level. If it the very large circuit; obviously we cannot represent this thing by a NAND gate or AND gate, so this can be module level. Now, this module is a set of that can be a set of gates or even set of sub modules and this sub modules are actually realized by set of gates. So, that is why the, they are the, there are different levels of abstraction.

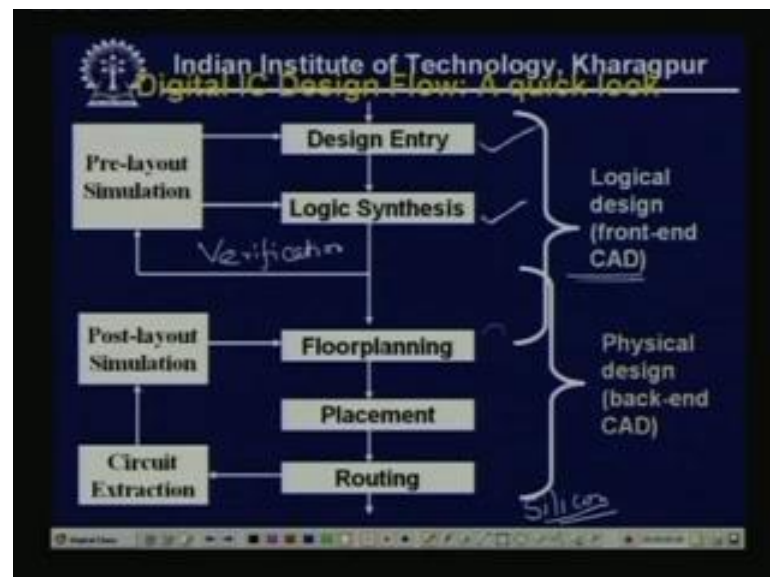
And just now, we have shown the example that is a gate level representation or it can be switch level, it can be circuit level and switch can be realized by transistors or AND gate. Now, in each level more detail is revealed about the implementation.

(Refer Slide Time: 23:59)



Now, the physical representation, so the lowest level of physical specification is photo mask information required by the various processing steps in the fabrication process. So, this is out of scope of this class, so just we mention that again the physical representation can be done.

(Refer Slide Time: 24:20)



Now, we see the, the actual digital IC design flow; that means, when the circuit becomes very large then how they can be automated using the CAD tools. What are the different steps to be followed? So, first is the design entry or just now the thing we have

discussed, that how the behavior of the design of the design specification can be represented. Then the second one is the logic synthesis.

Now, from there how we can get the, the netlist or the, the circuit realized by some gates, these are called the logic synthesis, so one CAD tool is needed for this logic synthesis. Then this is a pre-layout simulation, so mainly or here actually some more steps we can apply, add which is the verification; that means, whether after this logic synthesis, the logic we have got, whether it is actually giving the correct result or not.

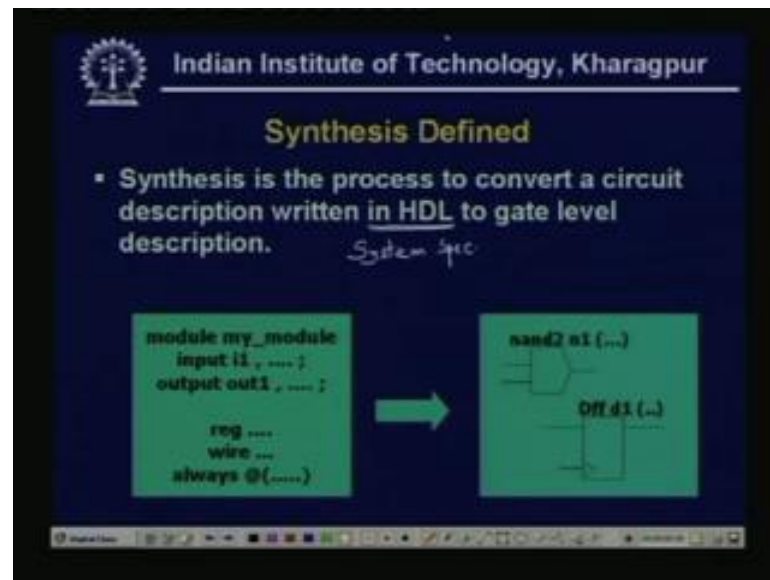
This is called the simulation, means applying some inputs whether I am getting the correct output or not, that I want to simulate. Here in different way we can verify also this is called the verification that also we can do the design verification also. Now, normally these are called the logical design or front end CAD, so some CAD tools are already available, they developed to get the netlist or to get the logic synthesis from the specification of the digital system.

Now, once we have got the logic synthesis of the netlist is available, then we go to the physical design part or normally it is called the back-end CAD. Again for this, the CAD tools are available for this physical design, normally the physical design means, the steps involved to get the logic of the circuitry on a silicon. So, after this actually, my ultimate product is a piece of silicon on which the circuit is fabricated.

For this, we need the floor planning placement, routing, mainly these three steps to be followed and for each step, this is we need a post layout simulation. So, here from here, again after routing we do the circuit extraction, we do the simulation whether the circuit, circuit is actually correctly working or not. Now, from this design entry to get the silicon as we, we are coming down, actually we are, we are coming to more close, closer and closer to the hardware.

Here it was a the design entry is in abstract level then, we can use some algorithm or the hardware description language then, we are getting the logic synthesis which is realized by the some gate or hardware equipments at the represented with the definition of the hardware. Then, it is floor planning how actually on the piece of silicon, here floor is the the silicon area, so how they can be, what are the space for them, then how they can be placed and how they can be routed means, actually interconnection, so we are getting more and more, to more close to the hardware.

(Refer Slide Time: 28:37)

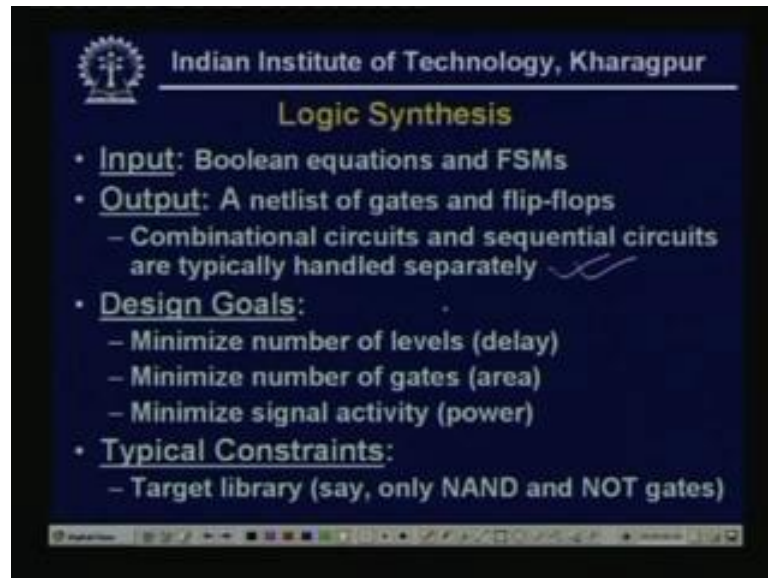


Now, we the first step just after the design entry, already we have discuss the design entry part, the behavioral the structural or the physical representation using the Boolean equations or the hardware description language or some table look up. In table look up format, we can give, now the second step is the logic synthesis, so first we give the definition of synthesis what do we mean by synthesis. So, synthesis is the process to convert a circuit description written in HDL to gate level description.

Here we have assumed that the design entry is given in HDL or the systems specification is in a hardware description language like verilog or HDL. So, system specs is given in HDL, so from HDL to gate level description is called the logic synthesis. We take one small example, see this is a module, my module input I 1 output out 1 some register, where and some HDL logics are given.

Now, given this problem spec this is a small, the specification of a small system, then they can be converted into some NAND gate some D flip flop. So, these are my hardware equipments or the system is now defined with the hardware equipments like, like logic gates, flip flop and their, this is a synthesis, this is a logic synthesis from a HDL portion.

(Refer Slide Time: 30:38)



So, logic synthesis the input can be Boolean equations and FSMs, output a netlist of gates and flip flops, now combinational circuits and sequential circuits are typically handled separately. And in this class already we have seen that from the Boolean equations how they can, the circuit can be synthesized or the combinational circuits or different sequential circuits, how they can be synthesized, that part already we have read.

So, here now for large circuit the, the CAD tools are developed to implement those techniques to automate the procedure, which will be very easier to synthesize the large circuits, that is the current state of the art. Now, what are the design goals, design goals are minimize number of levels the delay, minimize number of gates, again area minimize signal activity that is power.

So, that is why always we say to reduce the delay, area and power, these are the three parameters to be handled for a efficient design. So, these are my design goals to minimize delay, area and power, now the typical constraints are the target library say only NAND and NOT gate; that means, how the circuit is or the logic is synthesized by using what type of gates. So, what gates are available that is called the library and if it is only NAND and NOT. Obviously, that is my constraint, so if all the gates are available, it may happen that we get a very good design structure, some optimized one which reduces all the three.

(Refer Slide Time: 32:47)

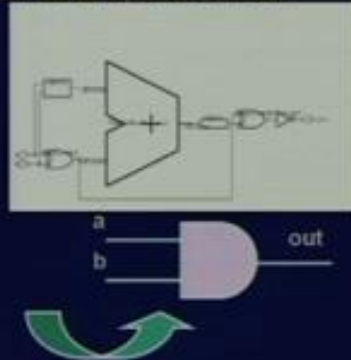
Indian Institute of Technology, Kharagpur

Logic Translation and optimization

Consider the following equations :

$$C = a \text{ xor } b$$
$$C1 = \sim((a \& b) + c)$$
$$d = c \text{ xor } c1$$
$$\text{Out} = \sim d$$

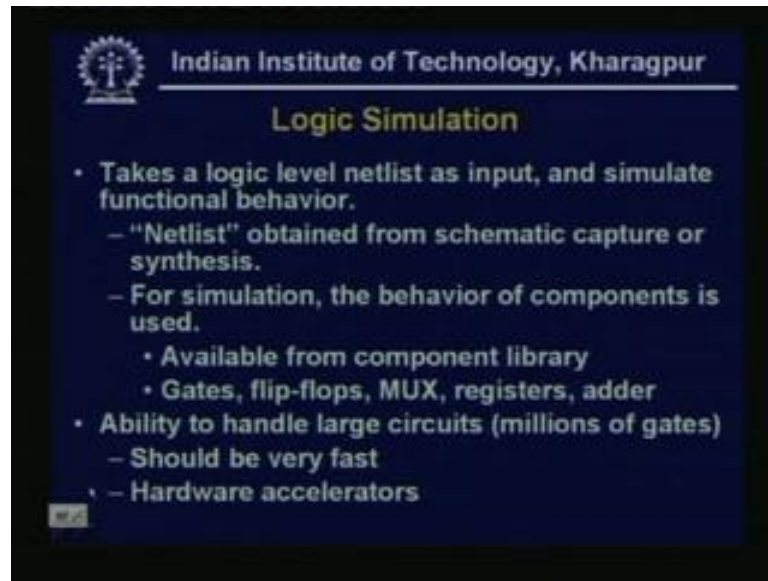
After Optimization the circuit becomes,




Now, another step is actually involved within the synthesis part, that is called the logic translation and optimization because, once the logic has been synthesized it may happen that it is not optimized. So, again we take one example, see consider the following equation that C is a xor b, so this is a, so this is a, a xor, a xor b then C 1 is negation of a and b plus c. Say this is a b and a and b and this is a plus, so this is the thing, now d is C xor C 1, so this is my C, C and this is my C 1, this is my C 1, this is my C.

So, this gives my d, this output is d and the final output is negation of d, d compliment, so this is the, had some equations of this, the Boolean equations for this circuit. Now, after optimization the circuit becomes, is nothing but a 2 input AND gate, that say it is a a b and out, output is the, this type of thing, that this is a, a AND b. So, this is the optimization because this here, so many circuitry is needed and here only 1 2 input AND gate is sufficient, so this optimization is one part of the synthesis.

(Refer Slide Time: 35:18)



 Indian Institute of Technology, Kharagpur

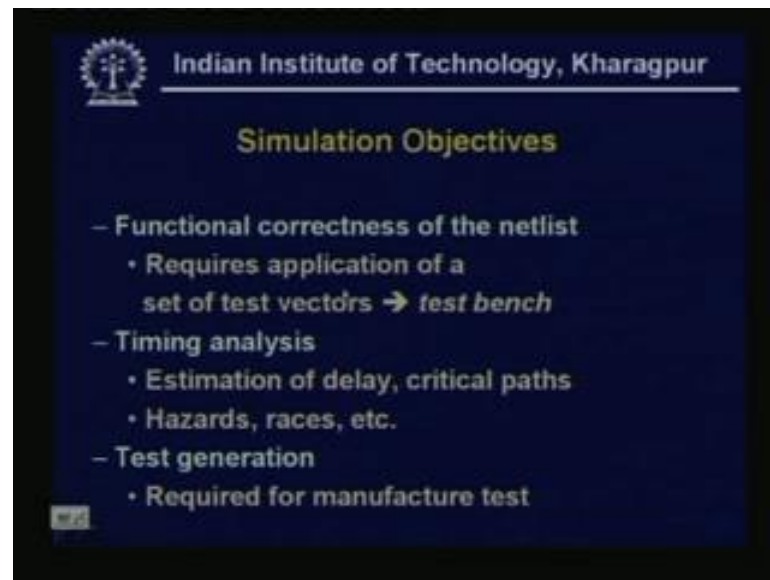
Logic Simulation

- Takes a logic level netlist as input, and simulate functional behavior.
 - “Netlist” obtained from schematic capture or synthesis.
 - For simulation, the behavior of components is used.
 - Available from component library
 - Gates, flip-flops, MUX, registers, adder
- Ability to handle large circuits (millions of gates)
 - Should be very fast
 - – Hardware accelerators

Now, the third step is logic simulation, so this takes a logic level netlist as a input and simulate functional behavior and netlist obtained from schematic capture is the actual circuitry or synthesis, for simulation the behavior of components is used. Now, these are available from component library, gates flip flops MUX registers adders these are my equipments which are necessary for my logic synthesis and we knew, we know the how or the truth table of these gates.

The input output of the flip flops that is the characteristics of all these equipments and then the logic can be simulated. Now, ability to handle large circuits of millions of gates should be very fast or the hardware accelerators because, all these CAD tools are what, today we are discussing the state of the art design procedure. Mainly that is for a millions of gates together or millions of gates fabricated on a small piece of silicon, so mainly it should be very fast and the, it should generate the hardware very fast.

(Refer Slide Time: 36:34)



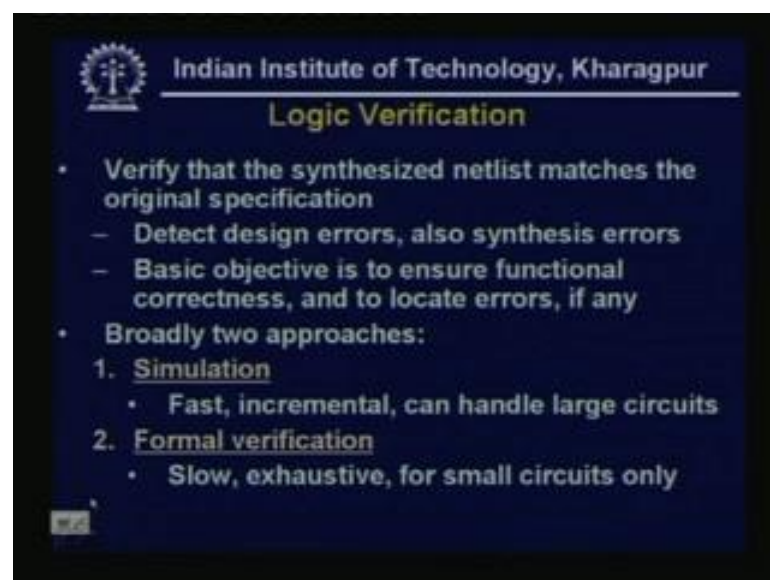
Indian Institute of Technology, Kharagpur

Simulation Objectives

- Functional correctness of the netlist
 - Requires application of a set of test vectors → *test bench*
- Timing analysis
 - Estimation of delay, critical paths
 - Hazards, races, etc.
- Test generation
 - Required for manufacture test

So, simulation objectives are functional correctness of the netlist, this requires application of a test vectors, we apply or test vector to some test bench, we have to generate. Then, timing analysis the estimation of delay critical paths, whether Hazards are there, above added races conditions are in overcome and the test generation this is required for manufacture test. So, these are the simulation objectives.

(Refer Slide Time: 37:07)



Indian Institute of Technology, Kharagpur

Logic Verification

- Verify that the synthesized netlist matches the original specification
 - Detect design errors, also synthesis errors
 - Basic objective is to ensure functional correctness, and to locate errors, if any
- Broadly two approaches:
 1. Simulation
 - Fast, incremental, can handle large circuits
 2. Formal verification
 - Slow, exhaustive, for small circuits only

Then, another is logic verification, so it verify the synthesized netlist matches the original specification, detect the design errors also synthesis errors, basic objective is to

ensure functional correctness and to locate errors if any. So, broadly two approaches are there, first simulation is one because, simulation also we are giving some inputs and what are the outputs we are checking.

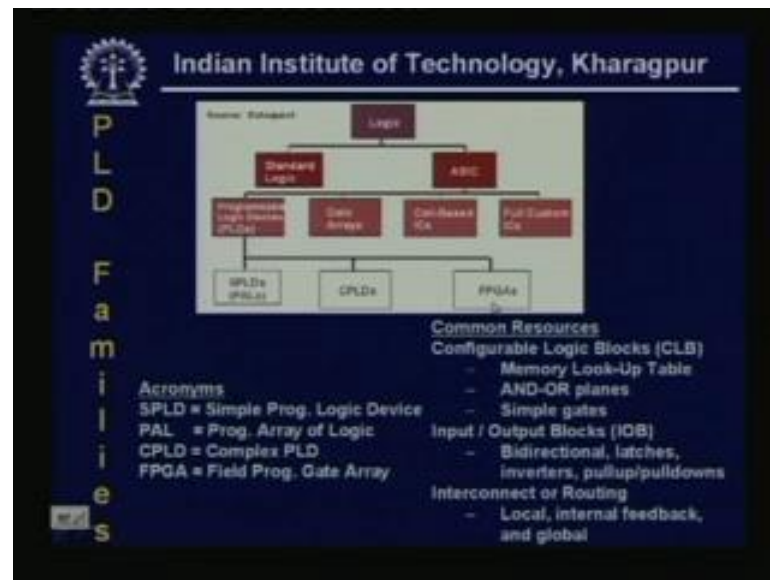
So, these are fast incremental and can handle large circuits this simulation or another can be formal verification, very slow exhausted for and that is mainly for small circuits only, but now a days people are doing for large circuits also. So, these are for my, the design procedure, now quickly we go that, what are the design methodology. We see that what are the different type of design methodology, what is the current state of the R.

(Refer Slide Time: 38:12)



Now, for VLSI that means, a Very Large Scale Integrated Circuit, where the millions of a millions of millions of gates are fabricated on a small piece of silicon, we call that is a very large scale integrated circuits. So, mainly this, that this the design, digital design flow that just now I mention they are followed for this VLSI designs, now what are the technologies or what are the design methodologies. Normally, there are three, the programmable logic devices, standard cell based design and full custom design.

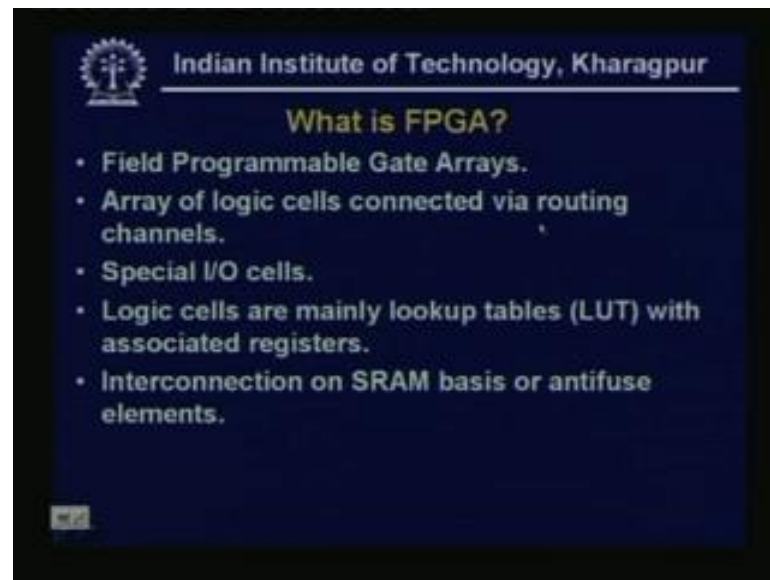
(Refer Slide Time: 38:54)



Now, these are programmable logic devices, already we have read the, how these PLDs means the PAL. PLA they can be designed, they can be, what is the actual circuitry of the PAL or the PLA. Now, using this programmable logic devices, this can be the systems or this VLSI circuits can be implemented. Now, the logics are normally of two types, standard logic and ASIC.

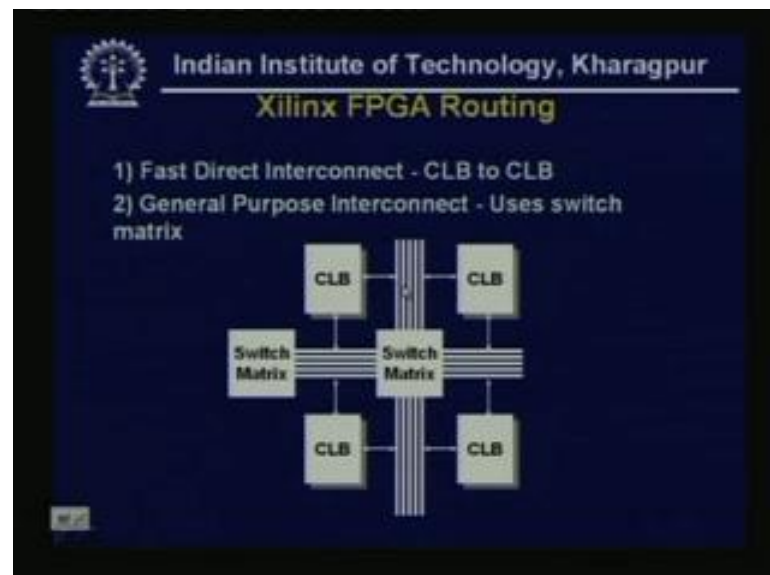
And this ASICs that Application Specific Integrated Circuit, they are programmable logic types, gate arrays, cell based IC and full based full custom IC. Now, this programmable logic devices are of PAL already we have read, the CPLD is Complex Programmable Logic device or Field Programmable Gate Array, that also we have read that actual design structure. And now, using those the large circuits or VLSI circuits are implemented.

(Refer Slide Time: 40:02)



FPGA already the structure we have seen, so this is Field Programmable Gate Arrays, array of logic cells connected via routing channels. There are special I O cells and logic cells and mainly lookup tables with the associated registers and interconnections on SRAM basis or antifuse elements. We have also read this FPGA.

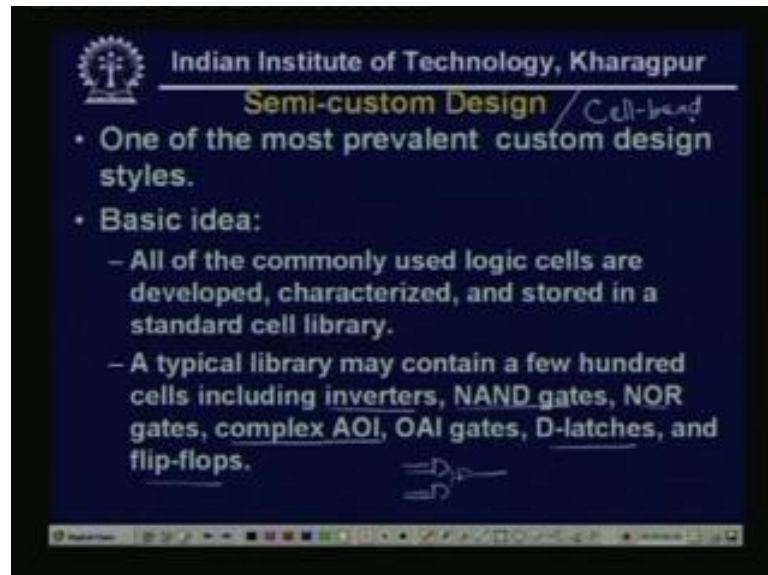
(Refer Slide Time: 40:25)



This is one Xilinx FPGA routing, so this is mainly the, the CLBs are there, the Complex Logic Block and mainly some switch boxes are there which are connected this logic box. So, this is the overall idea of the FPGA how this can be programmable, mainly they are

programmable using this switch box. So, general purpose interconnect uses switch matrix.

(Refer Slide Time: 40:58)

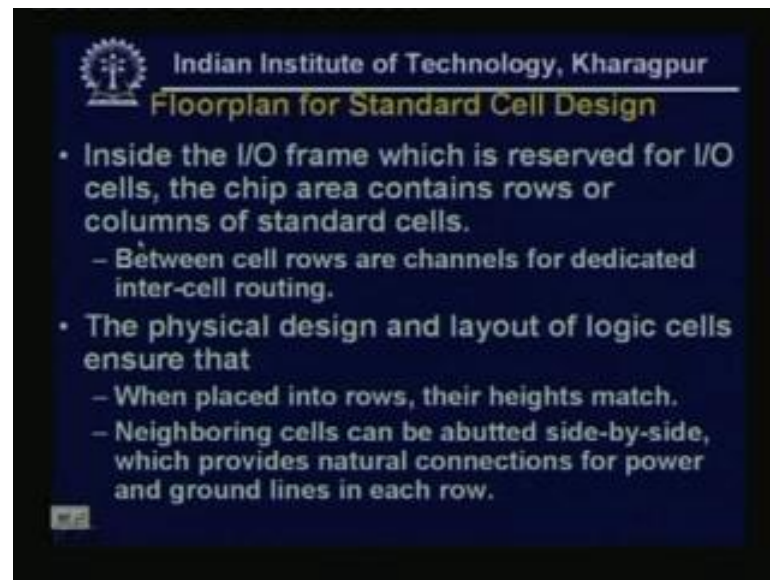


Now, the secondary one is semi-custom design that one of the most prevalent custom design style, the basic idea is all of the commonly used logic cells are developed, characterized and stored in a standard cell library. Nowadays for digital design, the moderately complex or very complex digital systems, this is the state of the art, that the logic cells they are developed, characterized and stored in a standard cell library.

And main concept is reuse, now for a new system to be designed, this already tested already characterized, logic cells kept in the library, that can be used. So, typical library may contain a few hundred cells including inverters, NAND gates, NOR gates, complex AOI, AOI means AND OR say 2 input AND and then it is OR. This is a AOI or OAI OR AND inverter, D-latch, D flip and flip flops.

So, there are several other things or this can be a larger modules that can be used as a cell. So, they are typically kept on a library and they can be reused for a new design. So, that is called the semi custom design or many time it is called a cell based design, this is a cell based.

(Refer Slide Time: 42:54)

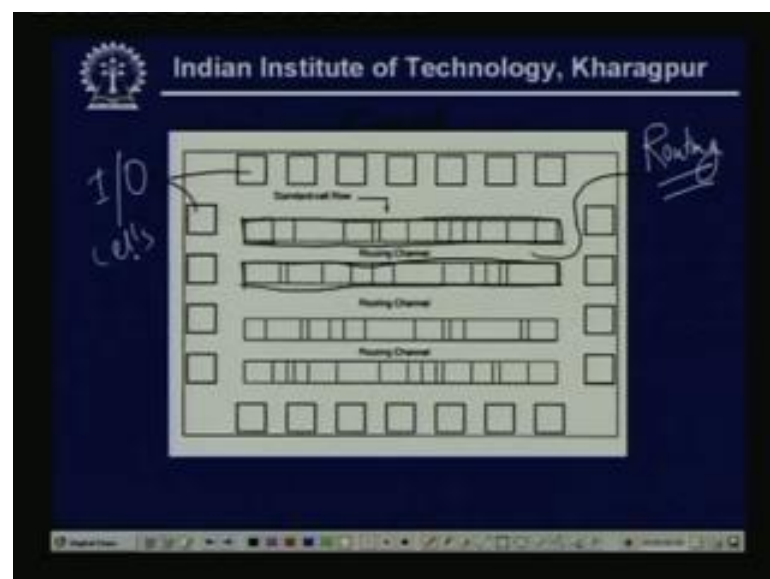


Indian Institute of Technology, Kharagpur
Floorplan for Standard Cell Design

- Inside the I/O frame which is reserved for I/O cells, the chip area contains rows or columns of standard cells.
 - Between cell rows are channels for dedicated inter-cell routing.
- The physical design and layout of logic cells ensure that
 - When placed into rows, their heights match.
 - Neighboring cells can be abutted side-by-side, which provides natural connections for power and ground lines in each row.

Now, floorplan for standard cell design, so inside the I O frame which is reserved for I O cells the chip area contains rows or columns of standard cells. Now, between cell rows are channels for dedicated inter cell routing and the physical design and layout of logic cells ensure, that when placed into rows their heights match, neighboring cells can be abutted side-by-side which provides natural connections for power and ground lines in each row.

(Refer Slide Time: 43:24)



So, if we see the one pictorial this thing, that here actually this is the, this is the rows, where this is the rows, this is the rows where the standard cells can be placed. So, this height should match and this, this is another row where thus again the cells can be placed from taken from the library. Now, between two cells or between two rows this space available this is called the routing channel.

This is called the routing channel; that means, this is the area available for interconnecting the cells and this is called the routing, the interconnection. So, this is called the standard cell row and these are the routing channels. So, over the cell routing is also possible, there are different design styles and these are the mainly for I O cells, these are, these are I O cells.

This is overall, very simple structure of the semi custom design, mainly that of from library taken from the library, the equipments are placed on the cell row, standard cell row and then they are, routed by using this area available between two rows and that is called the routing area or routing channel.

(Refer Slide Time: 45:19)

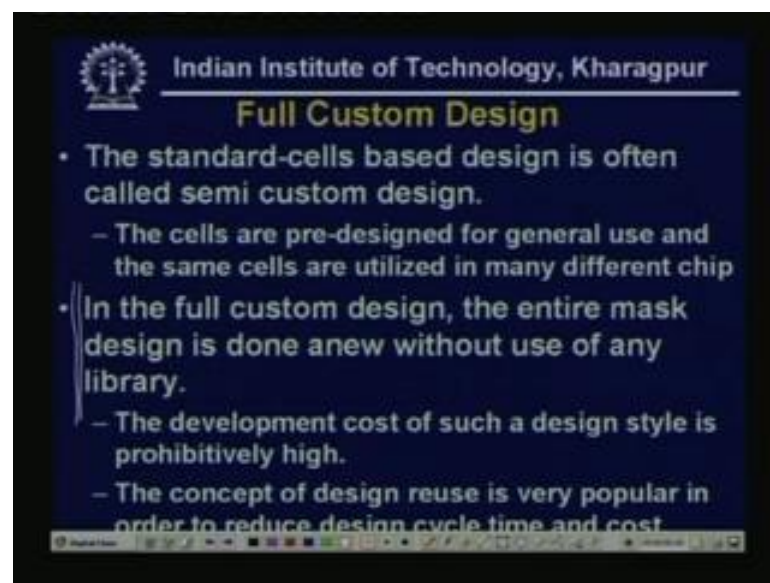


Now, the placement actually now, actually the, the physical design portion, that after chip, logic design is done using standard cells in the library, the most challenging task is to place individual cells into rows where the rows should be placed. For that the interconnection length will be minimum; that means, the routing cost should be minimum and parallelly the delay the, the area, area should be minimum.

So, the interconnect them in a way that meets significant design goals in circuit speed, chip area, speed chip area and power consumption. Mainly these are my three objectives to be made, many advanced CAD tools for place-and-route have been developed and used to achieve the above goals. So, many algorithms are available for this placement and the routing and this CAD tools are nothing, but the implementation of this algorithms to get a efficient design which optimizes speed, chip area and power consumption.

So, placement is the technique that, how and where to place the standard cells within one standard cell row. So, that it can be routed; that means, interconnected with other cells according to the netlist. So that, my objectives can be made, means the, my delay should be less, area should be minimum and power consumption should be low.

(Refer Slide Time: 47:08)



Now, is the full custom design, so the standard cell based design is often called the semi custom design. The cells are pre-designed for general use and the same cells are utilized in many different chip. In the full custom design the entire mask design is done a new, without use of any library, so this is very important that actually from such, we have to design, there is no cell library available.

Now, the development of cost of such a design style is prohibitively high, the concept of design reuse is very popular in order to reduce the design cycle time and cost. So, for digital, this thing is very important, that reuse and that is why the semi custom design is

being used for digital design, but for full custom design normally the, for analog circuits are being designed.

(Refer Slide Time: 48:08)



So, in real or like full custom layout in which the geometry orientation and placement of every transistor is done individually by the designer. So, design productivity is usually very low, typically 10 to 20 transistors per day per designer, so cost is high. Here the cost, design cost is very high, design cost is high.

In digital CMOS VLSI the full custom design is rarely used due to high labor cost, mainly here the cell based design or semi custom design is being used, where the we can reuse the cell library. Now, exceptions to this include the design of high volume product such as memory chips high performance micro processors and FPGA masters.

(Refer Slide Time: 49:11)



	Design Style			
	FPGA	Gate array	Standard cell	Full custom
Cell size	Fixed	Fixed	Fixed height	Variable
Cell type	Programmable	Fixed	Variable	Variable
Cell placement	Fixed	Fixed	In row	Variable
Interconnection	Programmable	Variable	Variable	Variable
Design time	Very fast	Fast	Medium	Slow

Now, comparison among the various design styles. So, so far, we have seen the different type of design methodology, the field programmable gate array, the simple gate array based, standard cell base design means our semi custom design and the full custom design. Now, a cell size is fixed for FPGA, the Field Programmable Gate Array. For gate array also simply gate array also it is fixed, for standard cell based design as already we have seen this is a fixed height.

Means here as already we have seen the, it is a, see here this is the, this is the semi custom design or the standard cell based design. So, actually there are rows and these cells, the cells taken from the or standard cells like, standard cells like NAND gate say NAND gate or NOT or D flip flop or even some larger modules, say AOI type modules. So, first they can be taken from the library.

And they are placed into the row, they are placed into these rows, now here when the one NAND gate is taken and placed on the row and in the same row if a flip flop is placed. So, this is a flip flop and say this is a, this is a AND gate, now the height of this cell that should match, otherwise they cannot be placed in the same group. Similarly, here the another NAND gate or say some AOI is placed and they are their height should match.

So, here that this, this is fixed type means, the height of the standard cell row, where the cells taken from the library are placed they can, they should match. Now, full custom design these are totally variable because we are not using any standard cells or the

already characterized cell, so these are the totally variable. Now, the cell type, the cell type is programmable for field programmable gate array, this is the programmable.

As already we have seen, that this is the programmable by using some switch box. And gate array is fixed, again for standard cell this cell types are variable because, they can be different type of cells are that are available in the standard cell library. The full custom, this is also variable because, this is a totally different and scratch it is being designed. Then the cell placement for FPGA this is a fixed type of array as already we have seen that this is a two dimensional structures and the complex logic blocks are placed on the cells and at the junctions the switch box are there.

So, the cell placement is fixed of the gate array; obviously, it is fixed, but standard cell, it is fixed in row, the standard cell rows are available the standard cells are being placed on the row and the routing is being done by the space available between two rows. For full custom this is totally variable, so full custom design is very flexible actually according to the designers own choice this can be implemented.

Now, the interconnection, so FPGA is programmable, again interconnection is also programmable by using the switch box. For gate array these are variable, for standard cell also this in interconnection also variable and for full custom these are also variable. Now, design time for FPGA this is a very fast, so that is why the current state of the art is the FPGA, particularly for academic purposes this can be these are being used.

Another thing is for FPGA these are very cheap because, already we have that fixed grid size available while the complex logic blocks and the switch everything is there. Only what we are doing that according to our own design the, the complex logic blocks are chosen and the we are doing the programmable I O also to get the actual design, so design time is actually few weeks here.

Now, for gate array this is very fast, for standard cell this is medium because, here though the cells are available the standard cells the different type of gates, the flip flops that are available, but actually the interconnections are being done depending on the current design. So, sometimes not all the cells are available and particularly if memory is there in a design, then this can be a large design and then the simulation time or writing the HDL code that part is also, the time is moderately large.

So, for standard cell this is a medium time gate array but, for full custom this is very slow is actually design time is very high because, as already I mention only per day we can design only 10 to 20 transistors. And then, the interconnections, the simulation and the testing of the circuit also, a consumes a lot of time. So, this full custom design is very slow, but we cannot avoid this full custom design.

Because, for some of the circuits particularly for new approach and particularly the analog circuits always we necessary the full custom for, but for digital design because, we are mainly discussing the digital system design. So, this is the, the current state of the art is the standard cell based design and this is using the, the cells already available in the library and the design time is very well.

So, mainly the CAD tool based design is the current state of the art the, whatever we discussed in this class the implementation etcetera. Now, the CAD tools are nothing, but the software which implemented those designs and now the large circuits are being implemented or realized by this technique. So, this is the end of the course the digital system design.

Thank you.