

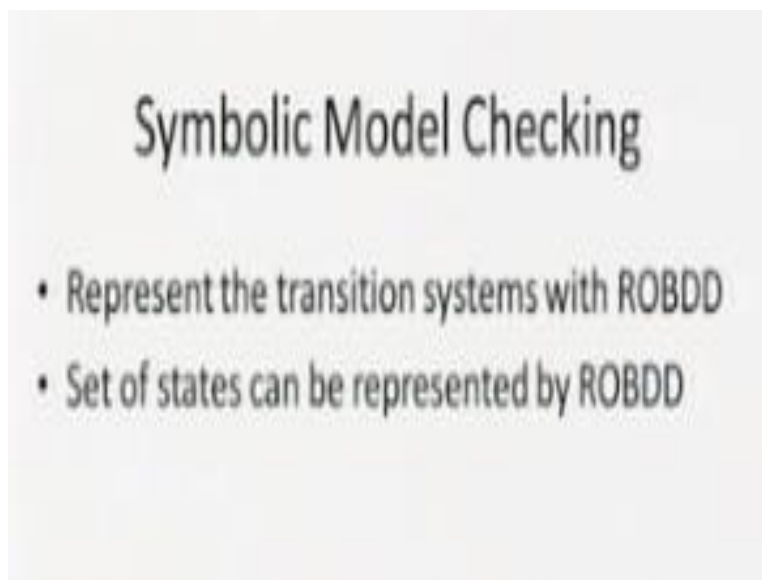
**INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI**  
**NPTEL ONLINE CERTIFICATION COURSE**  
An Initiative of MHRD

**VLSI DESIGN, VERIFICATION & TEST**  
**Prof. Jatindra Kr. Deka**  
**Department of CSE**  
**IIT Guwahati**

**Module VI: BINARY DECISION DIAGRAM**  
**Lecture V: Symbolic Model Checking**

Okay so, what we are discussing still now we are discussing about a about a structure called ROBDD Reduced Ordered Binary decision Diagram and we have seen that with the help of this data structure we can represent the most all Boolean function ant it always gives a complex representation of all Boolean function. Also in the last class we have introduced, how we are going to represent transition system with the help of the ROBDD.

(Refer Slide Time: 00:58)

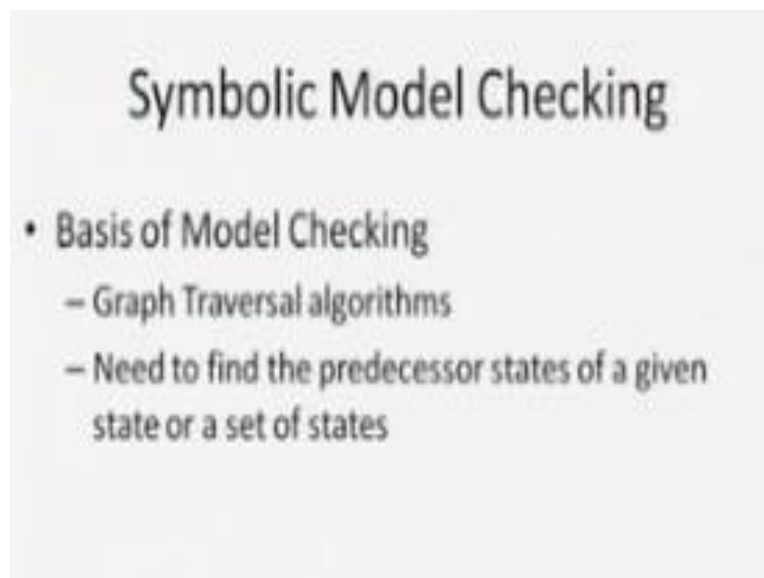


Also we have seen that, if we are going to take some sort of test ok. We are having transition system we are looking for a particular set of test and these particular set of test can also be

represent by ROBDD. So we can represent the entire test of this diagram with ROBDDs and the state of test also represented by with the help of ROBDD.

And I have slightly introduced that these particular set of test can be used to implement the module setting algorithm and one you use ROBDD in particular OBDD to represent the transition system and we are going to implant the manual algorithm. Then we call this particular model as symbolic models because we are symbolically representing the entire system.

((Refer Slide Time: 01:55))

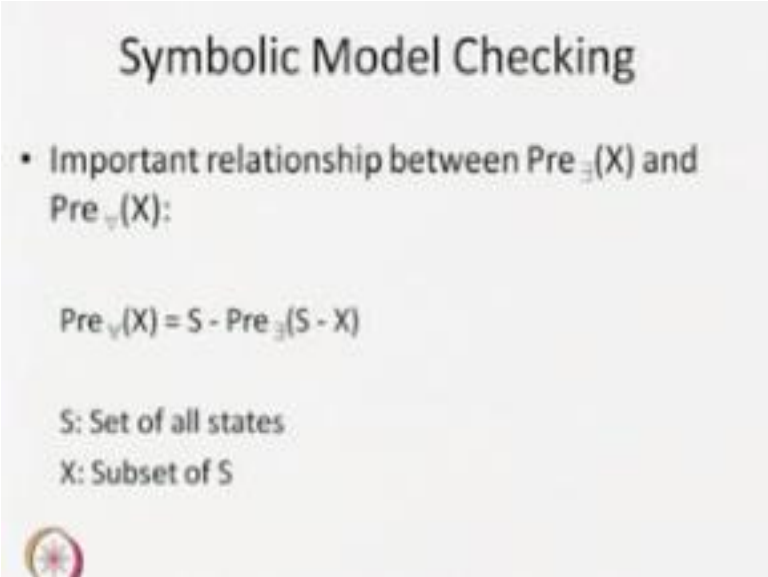


So basically what happen we know the models what is basics of symbolic model checking? If you look by you will find it is nothing but graph traversal algorithms because the biggest structure them to model our systems can be treated as a graph and in the particular we are using graph traversal algorithms to look for the particular state or set of state where the given property is two.

And the model checking algorithm is particular set of states. So in this particular graph represent the algorithm, if you look minutely you will find it all basic this is the set of given set of states are a given set okay.

So this is the basis thing that we are going to need to find out the predecessor of the given set or set of state.

(Refer Slide Time: 02:47)



**Symbolic Model Checking**

- Important relationship between  $\text{Pre}_\exists(X)$  and  $\text{Pre}_\forall(X)$ :

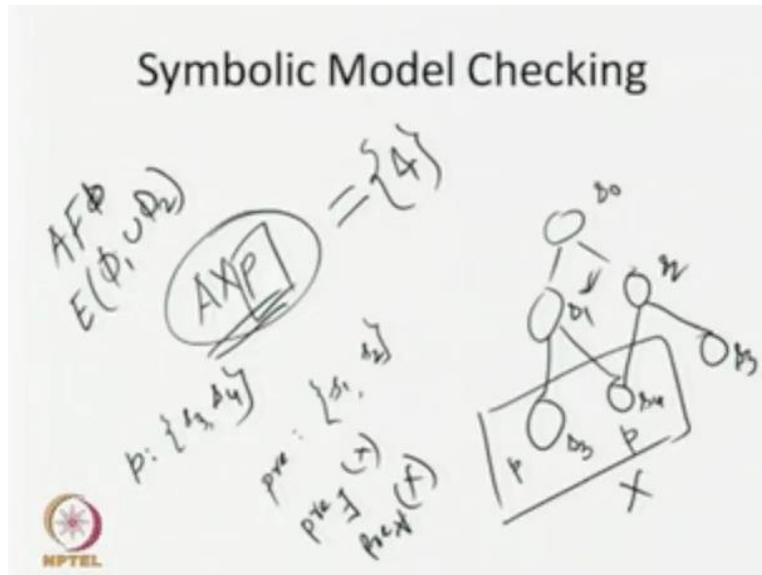
$$\text{Pre}_\forall(X) = S - \text{Pre}_\exists(S - X)$$

S: Set of all states  
X: Subset of S

So for that what happens we have to find out the predecessors? So just like that I am going to consider this particular system this is you  $X0, S1, S2, S3, S4, S5$  and say that now if you look in this particular thing what we say the steps were AXP that's means in all part where XP. Now in this particular case first we are going to see where p is 2.

So in this particular case  $P: \{S3, S4\}$  that means we are going to get this particular set as this is sub set, so in your next step what will happen we are going to get the predecessor as you S1 and S2 ok. S1 and S2 are the predecessor of this particular sub sets. Now over here what happen we are going to pre f(x) pre v(x), that for all x means coming g to this one particular sub set now in this particular AXP all the permission must coming to this sub sets.

(Refer Slide Time: 04:43)



So in this particular AXP all part with X that p is 2 that mean all the predecessor must come to this particular sub set. So in this particular case now the predecessor we are getting s1 and s2. Now we are going to look for this particular criterion where all the predecessor is coming to this particular sub set. We will find that for s1 all the condition are coming to this particular step sub set x but form s2 for one of the predecessor is not coming to this particular sub set x it is going outside of sub set.

So that means for all x will give me this particular set as on that means, I can say that AXP in all part in next to P is 2 is instead S1. So we are going to get this particular S1 and this particular S1. You just see that in the model symbolic of basic we need to find the predecessor state. If similarly I am going to look for the AFY are same EI1 and I2, in all the cases what will happen you are going to start the particular set of state and we are going to traverse this particular graph in backward direction.

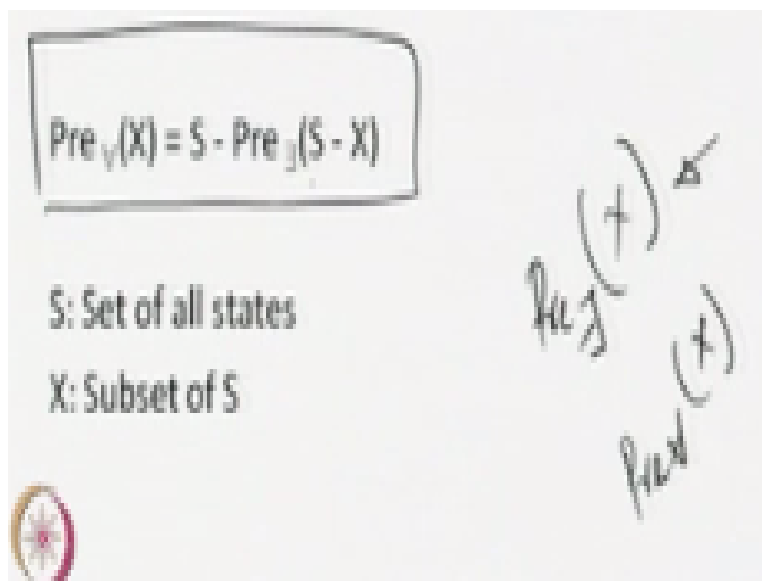
In backward direction, we are going to collect more and more step. Then we are going to find out which are the step that A and P or A as 5, or E as 5 and eventual we get a set of state. And our model algorithm are going to return this set of states and we have seen that, we can use that ROBDD to represent this particular set of states and ultimately our model in algorithms symbolic

models in algorithm is going to return in particular set of states and symbolically which represent with the help of power ROBDD.

So this is the basic things, so when we are going to use ROBDD for model symbolic algorithm to get the symbolic model graph to represent algorithm, you need this two function  $pre_f(x)$  and  $pre_v(x)$  ok. In the last class we have introduced these two.

Now also we have seen one important notes between  $Pre_f(X)$  and,  $pre_v(X) = S - Pre_f(S - X)$ .  $S$ : Set of all States,  $X$ : Subset of  $S$ . This is the relation Ship between  $pre_f(x)$  and  $pre_v(x)$  and if we calculate  $pre_f(X)$  and  $pre_v(X)$  that mean, we need to calculate if we have this particular set. We have to evaluate this particular set of states then by using this particular sets.

(Refer Slide Time: 08:09)



Now we are going to see how to evaluate this particular set of states that means  $pre(X)$  ok. We need a method to calculate, this is the basics of all symbolic module. If I give you a set state only my requirement is to find the set of state and out of that I am going to say what are the step for the  $pre(x)$  or it is satisfy for the  $pre(X)$  ok.

(Refer Slide Time: 08:43)

## Symbolic Model Checking


**Procedure for  $\text{Pre}_3(X)$**

Given,

- $B_x$ : OBDD for set of states  $X$ .
- $B_{\rightarrow}$ : OBDD for transition relations.

Procedure,

- Rename the variables in  $B_x$  to their primed versions; call the resulting OBDD  $B_x'$ .
- Compute the OBDD for  $\text{exists}(x', \text{apply}(\bullet, B_{\rightarrow}, B_x'))$  using the **apply** and **exists** algorithms.



In this particular case now how we are going to look in it, we are going to look for the procedure for  $\text{Pre}_3(x)$ . What is the input? Input given, the input given is your  $B_x$ : OBDD for set of states  $X$  ok. So I am having a set of square, this is the set square  $x$  and we are having some square  $x_1, x_2$  something like that.  $B_x$  we are going to say that this is the BDD representation of this particular sub set  $x$  ok.

And secondly we are going to set a  $B_{\rightarrow}$ ; basically we are having the transverse system. Already we have discussed how to represent the transverse system of OBDDs or may be ROBDDs. So we say that  $B_{\rightarrow}$  is the OBDD representation about transverse system.

So basically we are getting BDD one is your  $B_x$ , this is the OBDD representation of the set of state of  $X$  and  $B_{\rightarrow}$  is the OBDD representation for transverse system. Now one basic requirement is  $B_x$  and  $B_{\rightarrow}$ . What is the basic requirement; already I have mentioned that, since you are going to use some operation on this two particular variable, so they must set compatible variable ordered.

Already I have mentioned what is compatible variable ordering, they must set the variable ordering ok. So we are taking two variables of BDDs and OBDDs as our input and both the OBDDs must set as compatible variable ordering.

Ok now, what is the procedure? What we are going to do? Since, this is the simple one I am going to explain it. This is the procedure we are going to say that within the variable in  $B_x$  to their primed versions and we call it the resulting OBDD  $B_{x'}$ . What is prime version of this variable? Because we have already seen that, if we are going to represent a consensus is the we need to say the variable to represent the particular consensus, say if I am giving a consensus of say  $A$  as  $p$  to say  $x_n$ .

As  $p$  is my present state or current state and  $x_n$  is the next state of this particular consensus say  $p_1$ . So this condition will be represented by the ordered of two states. So basically these states, current state will be represented by some state variable, say you need some state variable then we are going to take the help of that variables say  $X_1, X_2$  to  $X_n$ .

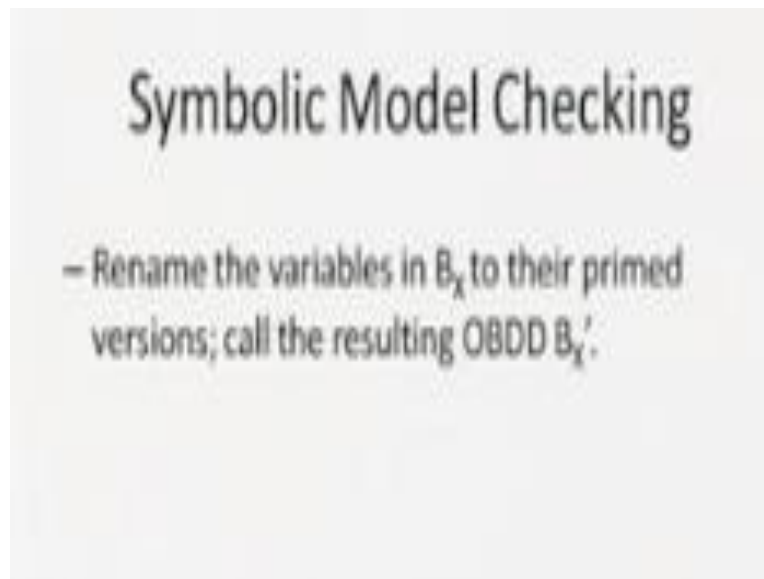
To represent this particular  $X_n$ , we need another set of variable and these sets of variables are basically the prime version of these particular original form variables. Now what we are going to rename these variables in  $B_x$  to their prime version. While explaining what we are going to get then we are going to compute the OBDD for this particular operation ok.

First we will apply, we will use this apply algorithm with this particular dot operator, with this to BDD. OBDD for the transverse system and OBDD for the set of states where the variables are renamed to their prime version and this particular algorithm will be a OBDDs ok. The variable ordering is same with your  $B$  arrow and  $B_x$ . Now after that what we are going to do?

We are going to apply this particular exists algorithm where we are going to make an independent algorithm of this particular prime version of the particular variable. So  $x'$  represents the, I can say that what is your  $x$  set?  $X$  set is nothing but the vector of all variable that we are using  $x_1, x_2$  and  $x_n$ . And similarly  $x'$  is the vector of variable of the prime version of this one  $x_1', x_2'$  like that  $x_n'$  ok.

So this are the two steps we have to apply, one renaming of the variable and second one is we are going to use the particular method for applying the algorithm with dot operator, then we are going to use the exists algorithm ok. So this are the two steps, that we are going to perform and get this particular BDDs exists X ok.

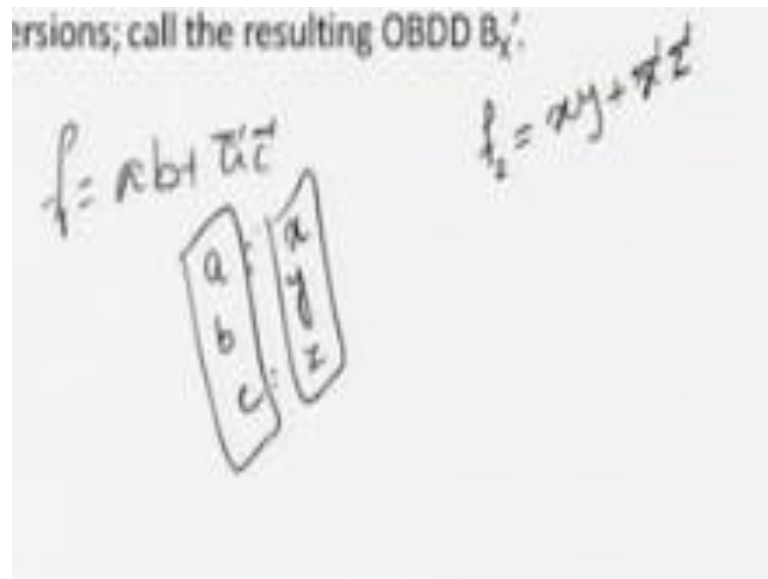
(Refer Slide Time: 13:27)



Now renaming the variable in  $B_x$  to their primed versions, now what happens if I am going to give you a function of  $f=ab+a'c$  and if I am going to write another function  $fz=xy+x'z'$ . Now if you look into this particular two function  $f_1$  and  $f_2$ , you can realize this is the two function because what will happen the variable  $a:x$  and  $b:y$  and  $c:z$  that mean s the name of variable is different. That means we can say that, we are just renaming the particular variable  $a, b, c$  to  $x, y$  and  $z$ .



(Refer Slide Time: 14:45)



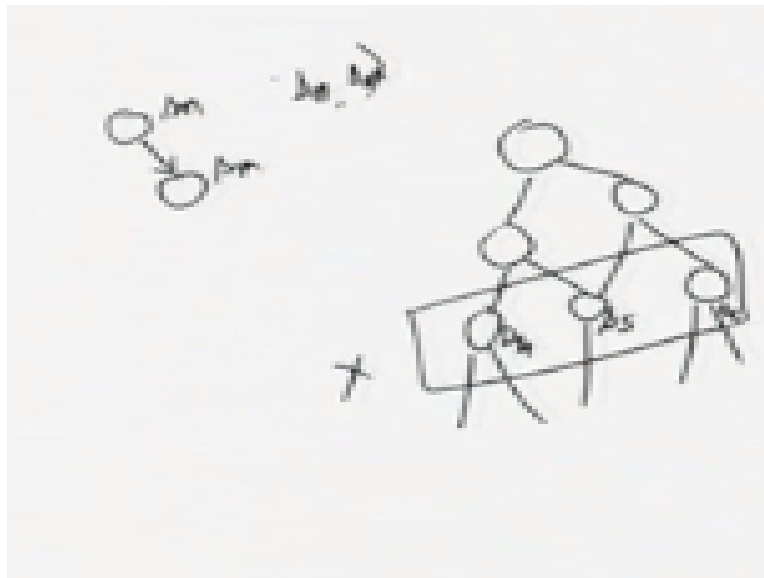
Now if you look at the evaluation, if any evaluation say  $a=1$ ,  $b=1$  and  $c=1$ . Whatever function we are going to get for  $f_1$  if  $x=1$ ,  $y=1$  and  $z=1$ . We will represent the same function or  $f_1$ . For all combination we are going to get the same representation. That is why we are saying  $f_1$  and  $f_2$  are equal or if they are giving the same function.

So these functions are  $f_1$  and  $f_2$  are basically same, they do not have any definite only we are giving the variable  $a, b, c, d$  to  $x, y$  and  $z$ . Now similarly in our  $f_1$ , so now what will happen for  $f_1$  I am going to get a BDD and if you look for the  $f_2$  we are going to get the same BDD only the variable will be renamed now if this is you're  $a$  now you will be rename to  $x$ , similarly if next level is variable  $b$  then that will be  $y$ .

Now similarly what will happen, in  $B_x$  this is OBDD representation of about state of  $x$  with particular variable  $x, y$  and  $z$ . Now  $B_x$  is  $X_1, X_2, X_3, \dots, X_n$ . So  $B_x$  is a representation of sub set of states. And the variables are the  $x_1, x_2, x_3$  and like that  $x_n$ . Now  $B_x$  prime what we are having? We are having the same BDDs but we are renaming this particular variable  $x_1', x_2', x_3'$  and  $x_n'$  ok.

So this is the scenario that we are having. Now see this, if I am having the transverse system. This is the sub set, say  $x$ ,  $s_5$  and  $s_6$ . Now what basically happens, you are having a condition to  $s_n$  to  $s_m$ . Then you are going to represent this condition with this particular order  $s_n$  and  $s_m$  ok.

(Refer Slide Time: 17:30)

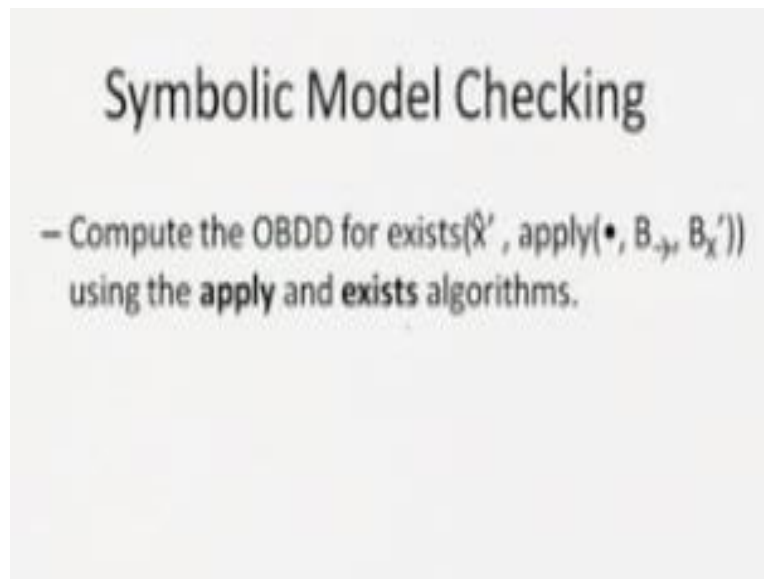


Now when I am having this particular sub set so I am just saying this is a  $B_x$  representing the particular sub set for  $s_4$ ,  $s_5$  and  $s_6$  ok. Now say  $b_x$  they are representing with the state variable. Now I am saying just renaming the particular variable in  $b_x$  to the prime version:  $X_1'$ ,  $X_2'$ ,  $X_3'$ . So what is the basic and basically, if you look in to that particular condition or  $n$  is to find out the predecessor.

That means this are basically next step variable and we are going to find out what are the predecessor, that is why we are renaming this variable to their prime version. Just to get that particular transition ok. This is how basically so after renaming the variable we are getting the same structure as OBDDs. So whatever OBDDs we have that structure will remain same on renaming those particular variables.

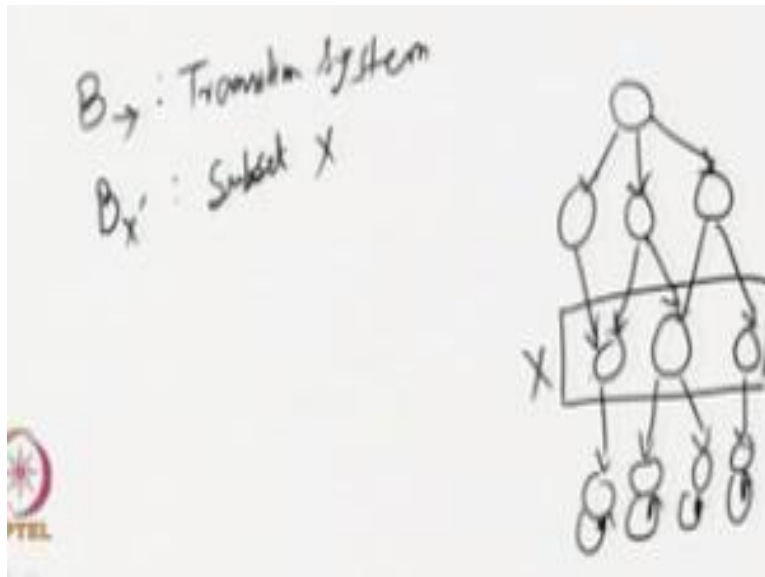
So after renaming the variable we are getting the  $bx$  prime. Now what is our next step, computing the OBDD for  $\exists x$  prime or you can say apply for  $B \rightarrow bx1$  using the apply and exists algorithm.

(Refer Slide Time: 18:52)



Now in this particular case what will happen? You just see, if you consider this part, you see that this is a sub set  $x$ , now you are getting  $B \rightarrow$  is the OBDD representation of my consistent and  $bx$  prime is the representation of the sub set of  $x$ .

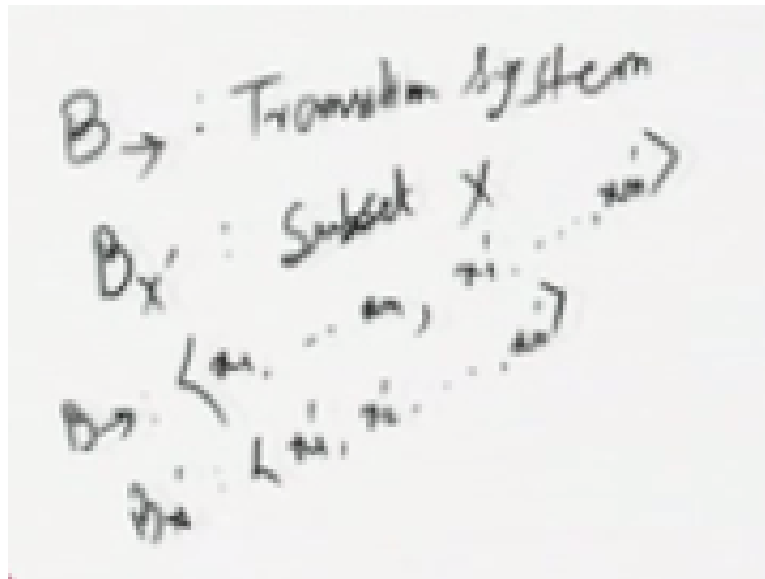
(Refer Slide Time: 19:56)



Now you are having this representation this particular sub set with the prime function of the variable. Now in b condition the BDD of transverse system we are having all the variable  $X_1$  to  $X_n$  and  $X_1'$  to  $X_n'$ . All the variables are available  $B_{x'}$  and your  $b_x$  prime is having the prime function of the variable ok.

Now when I use this particular apply operation, the dot operator it is going to give me the intersection of two sets. Now already we have discussed that we are going to apply this dot operation is going to be the intersection of the two sets. Now I am using two BDD one is going to represent the same sets and second one is representing the prime.

(Refer Slide Time: 20:56)



Now when we apply this particular dot operation is going to give me the common intersection of this particular function. So what basically we are going to get after applying this particular dot operation. We are going to get the in those congestion T1, T2, T3, T4, and T5. So it is going to give me the entire congestion, I have all the congestion. Now I have just intersect this particular transition or congestion with this particular sub set.

Now whatever we are getting, we are getting the congestion that all are coming into this particular x. since other transition which simply goes ahead. So this is a intersection operation we are doing. So basically we are getting this particular operation after applying, what we are getting? We are getting those particular congestion only in t1, t2, t3, t4 and t5 ok. We are getting this five transition because these are five intersections of these two sets.

Now what about the result we are getting about here? We are getting the transition, that is coming into all the transition coming to this particular sub set x. So in this particular BDD, what about result ant BDD we are getting? I am going to set this is your BDD; say only congestion is given, so this particular BDD is having all the variable and x1, x2 to xn and they are prime function x1', x2', xn1 and so we are getting particular congestion which are coming in to this

particular set  $x$  and this particular congestion will be represented by all this particular variables along with state variables next that variable.

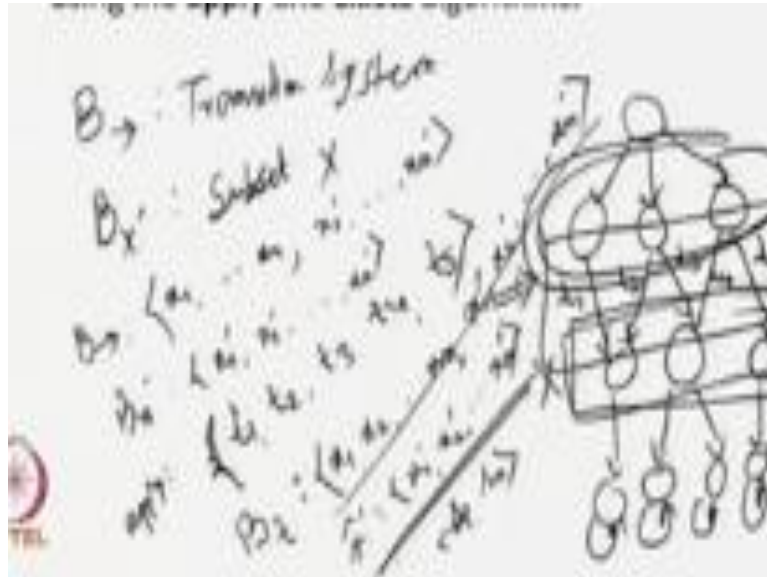
Now whatever BDD we are getting, now we are doing is? Now we are using this particular exists algorithm and what exists we are making  $b$  exists  $x$  that had, that means we are taking this particular factor  $x'$  at  $x_1'$ ,  $x_2'$ , like that  $x_n'$ . So what we are doing basically whatever results in BDD we are getting. We are making  $a$  independent of this particular prime version of the variables.

You just see what we are doing first thing apply a prime function and applying the dot operation and we are getting the intersection, we are getting particular congestion which are all our interest. That means that congestion which are coming to this particular set  $x$ .

Now from that what happens we are making it independent of the next step variable? So when we are making independent of this particular next step variable, so what happens now transition are having all the variables  $X_1$  to  $X_n$  and  $X_1'$  to  $X_n'$ . So this basically represents this particular say from present step to next step alter order.

Now you are making the independent of next step variable, so in this particular case what we are getting in the result and BDD whatever final BDD we are getting. Now it is independent of next step variable so eventually you are getting the representation of those particular steps only.

(Refer Slide Time: 24:46)



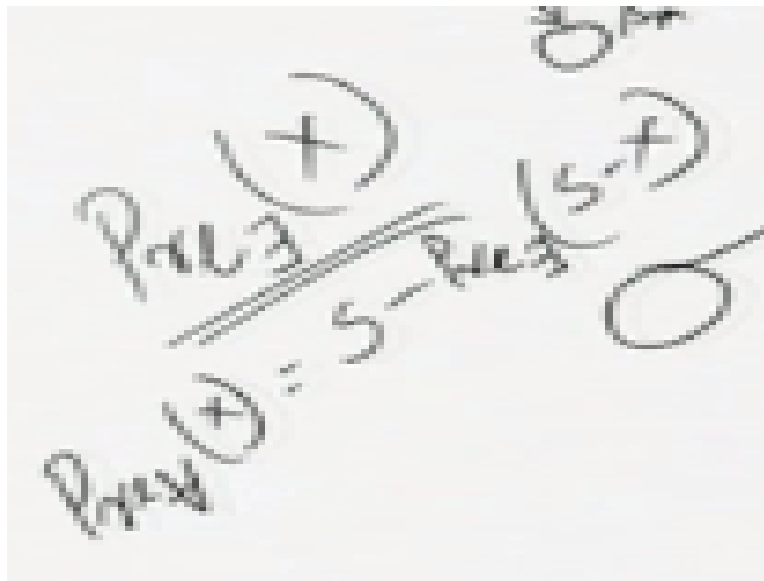
Now try to understand it, that we are getting this particular step only, that this are representing the step and this are represented by the step variable. You just see that whatever the ROBD, OBDD return by this operation. It is going to give us the steps of OBDD representation of the sub set from where all the congestion are coming into this particular sub set  $x$  ok.

So this is the way we are getting, so that means we are getting the result in BDD of where sub set and from those particular all the congestion are coming to this particular sub set. This is the where we are calculating the pre all the predecessor. So if all the predecessor that we are getting in, so predecessor they are exists  $x$  will also be the same because we are getting all the step of where the transition, all transition are coming to this particular sub set  $x$ .

So that means at least there will be one transition which is coming to this particular steps, so it is going to give me the pre exists. So if I am having some more transition something like that see these two steps will come into as a resulting step. Now since at least some of the states are coming in to this particular  $x$ . So now with the help of this simple operation in this particular two steps, first we are renaming the variables of  $b_x$  and then we are applying this particular operation or by using applying operation or exists operation to get this predecessor step.

And whatever predecessor step we are getting all those step will give us the pre exists x and ones we are having the predecessor exists we can very well calculate the pre for x and already we have said this is nothing but x-pre their exists. So now we are having the method to find the pre exists x and ones we can calculate the pre exists, we can calculate pre for x also.

(Refer Slide Time: 27:12)

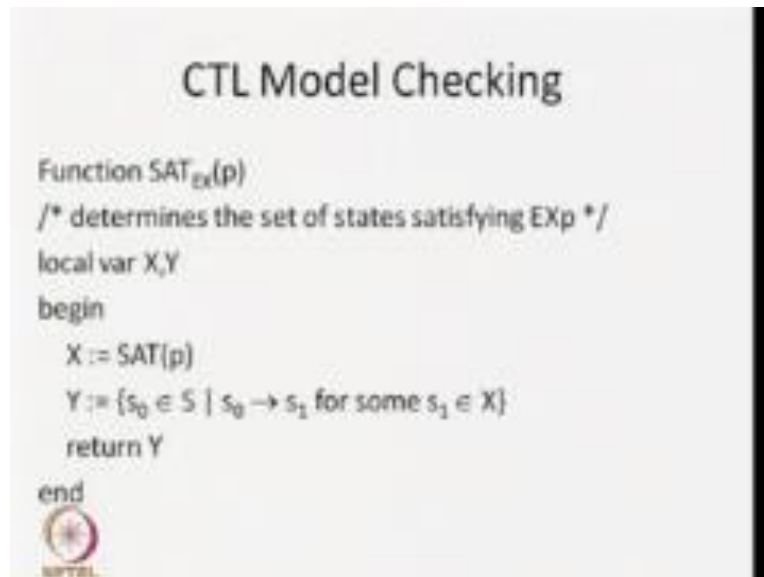


The image shows handwritten mathematical work on a whiteboard. At the top, there is a circled plus sign  $(+)$  above the expression  $P(x)$ . Below this, a horizontal line is drawn. Under the line, the expression  $S(x)$  is written. To the right of  $S(x)$ , there is another circled plus sign  $(+)$  above the expression  $S(x)$ . Below  $S(x)$ , there is a circled plus sign  $(+)$  above the expression  $P(x)$ . The overall structure suggests a derivation or a relationship between  $P(x)$  and  $S(x)$  involving predecessor existence.

Now this the best or this is the basic requirement of the symbolic model checking ok.



(Refer Slide Time: 27:20)



Now we are going to see how we are going to implement those symbolic model algorithms. Already we have seen that we are having, we have discussed with the four temporal operator next state  $X$   $F$   $G$   $U$  and this four temporal operator are predecessor. So always processed by the part A and E.

So we are getting a different combination but already we have seen that, if we get a three operators also, so that will give me a set of operators. So in that particular compose operator I need the next step operator, I need the un till operator or I need either Fuser or global one of this two.

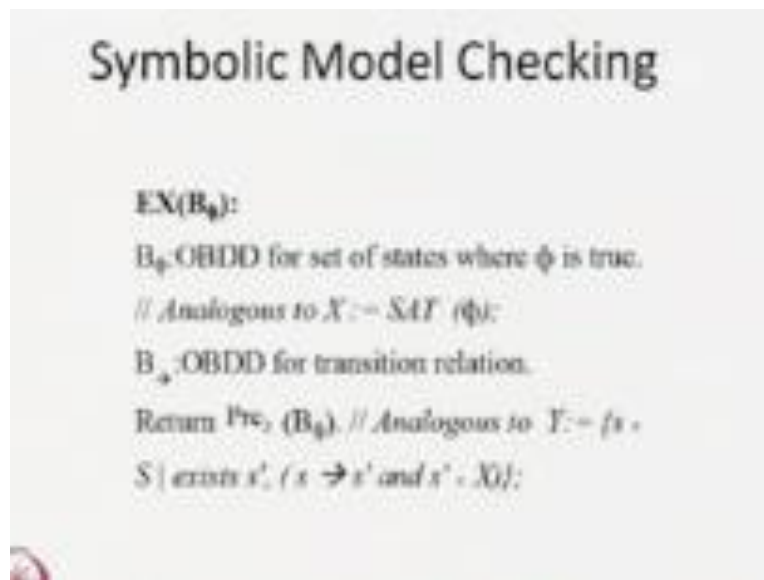
So either all a or ex whatever it may be, so it need three operators and already we have discussed the algorithm for those three operators that composes the operators. Now we are going to see how we are going to implement that things symbolically model checking algorithm.

So first operator we are talking about  $EXp$  that means they exists a part where next step  $p$  is 2. This is the simple procedure. So what we are taking first we are taking all the steps where  $p$  is 2 so satisfied  $p$  is going to define all the steps where  $p$  is 2, so we are going to collect those step in

such a way that from those particular step we are having a congestion to s1 such that s1 belongs to this particular set x ok.

So this is basically the evaluation of ex. This procedure we can implement symbolically. So how symbolically model checking algorithm will look like?

(Refer Slide Time: 20:08)

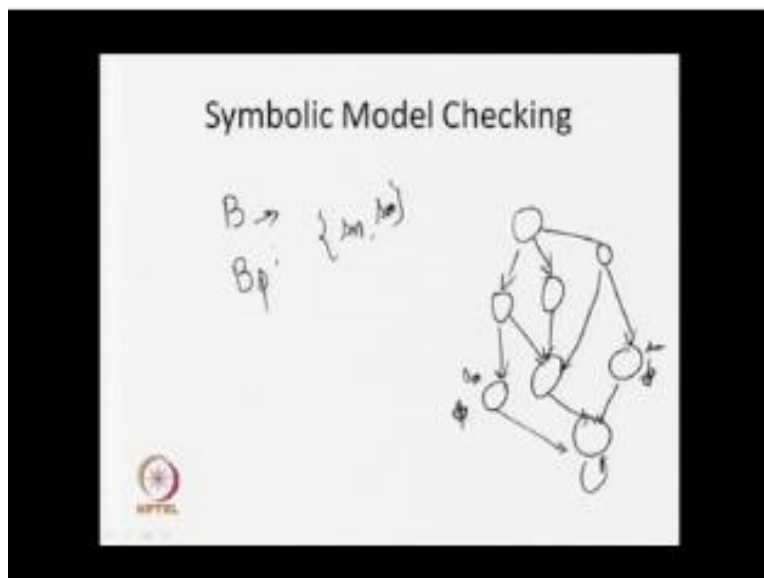


You consider Ex b5, what is b5 it is OBDD representation of the set of state where 5 is 2. Because ex5 we must know the set of state where 5 is 2 and we are going to represent those set of state with the help of BDD and we are going to represent the new file of the BDD is the particular representation of the particular file, this is analog to x send the state of φ is 2. Where analog to x sending the state of φ of particular x.

Now what about the B arrow is the OBDD representation of the state transaction system. Now whatever variable of the ordering we are having for the B arrow that we have to maintain it for B φ also, so from that what we have simply exists B φ. Now we are getting all the set this is analog to another step this particular thing y is equal to we are going to collect all the set from S. So that we are having from the S to S' belongs to this particular x.

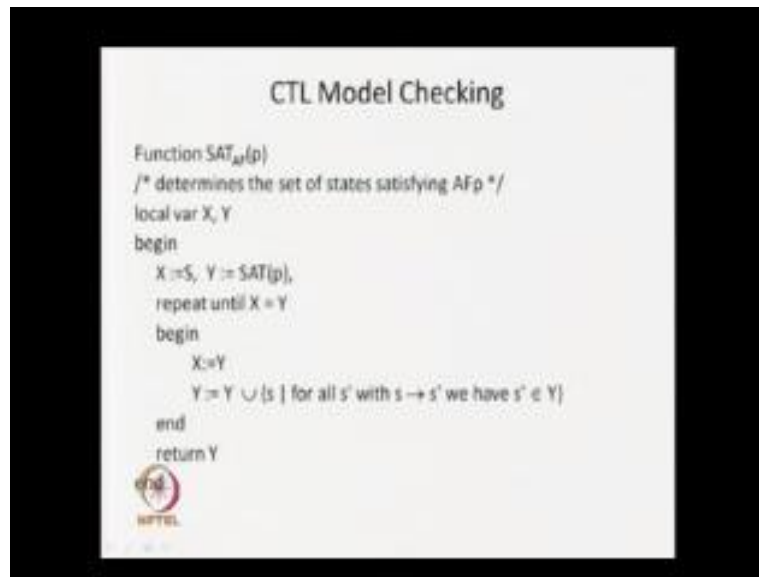
So this is the way that we are going to help so we simply use this particular  $Bx$  is simply  $x$  okay. Now what shall we consider, we can think something about like that, you say if  $p$  is true and then what we are going to get  $B$  arrow of which is the representation of particular whole transaction system, then you are going to get  $B \varphi$  where set of state where  $\varphi$  is true, these are the two steps where  $\varphi$  is true.

(Refer Slide Time: 31:30)



So now I am going to say that  $S_m$  and  $S_m$ , so this  $B \varphi$  is nothing but the BDD representation of your  $S_n$  and  $S_m$ . Now we are going to look for a all predecessor step so if you calculate the predecessor then these two steps will come okay. As to the evaluation and these are the steps where  $E_x \varphi$  is true okay. So we are going to give the input as of  $B\varphi$  for  $B$  transition on that we are going to evaluate this particular thing.

(Refer Slide Time: 32:27)

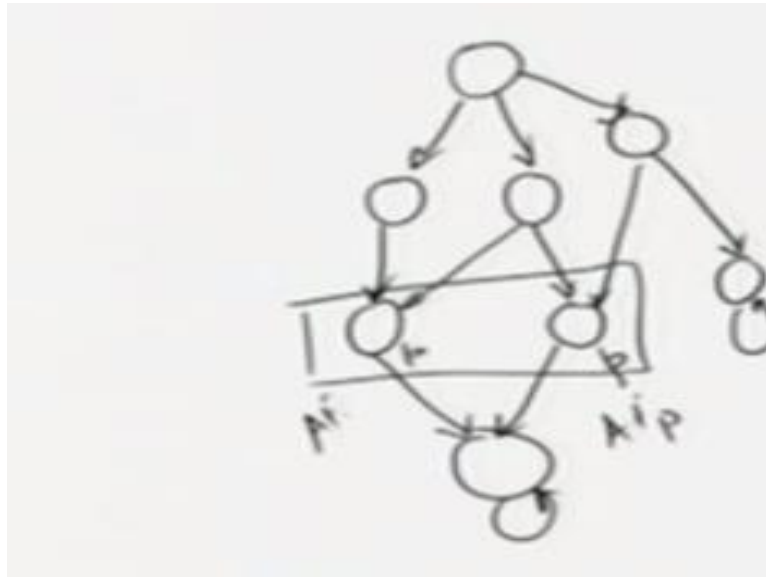


So B predecessor exists x of Bx of  $B\phi$  is going to give me the set of state is true okay. S

o this one algorithm is very simple one, so second one we are going to set up a state AFp is true, already we have discussed the algorithm so we are going to perform for this operation where ever p is true we are going to this is as per the semantics that have discussed says that present includes it. And this semantic presents that includes that which are being able to work for all the steps.

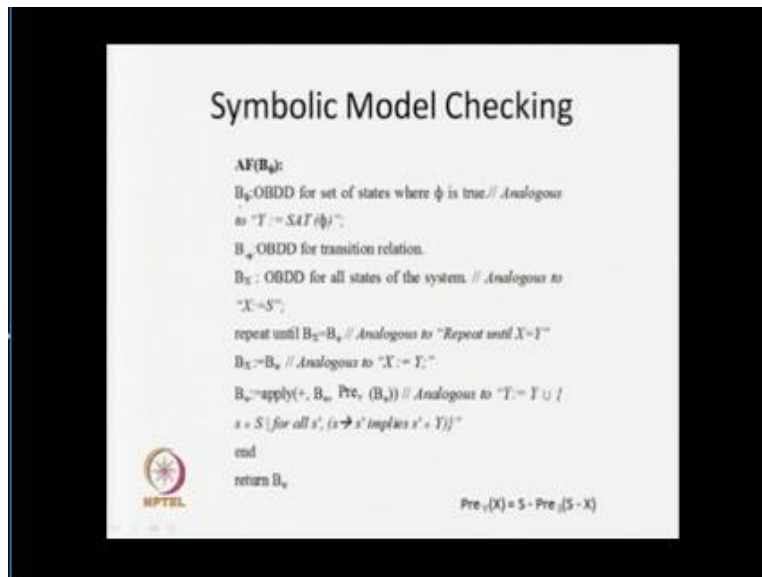
From where we are having a condition for those particular step so we are going to call this step so that we are having s to s' belong to y . So this is basically consists something like that. In all part in user file may be present. So you are having something like that and p is 2 over here when once I am going to take, then also p2 is also over here then I am going to take this particular because of set we are going to calculate with the previous steps all the condition that are coming for the source.

(Refer Slide Time: 33:35)



So in this particular case all this condition coming true in the subset of particular  $x, s$  of  $AFp$  will be 2 over here and  $AFp$  will be 2 over here, then if I am going to look for this on then  $AFp$  is 2 is true for this state but this state is not  $AFp$  is not true so this state will not come. So we are going to repeat this particular predecessor, so that is why you are saying that collect some more steps and then again see and go for an where  $x=y$  or not at some point of time that means you are not adding any more steps then we are going to terminate this step for this particular predecessor.

(Refer Slide Time: 34:30)



So this is the same predecessor we can use now and apply or symbolic model algorithm, so what we are having said  $B_{\phi}$  is the OBDD representation for a set of state, where  $\phi$  is true. First we are going to start from the particular set of state, where the given formula is true like that starting from this particular subscriber and now going to perform a predecessor operator and going to find out what are the steps where  $AF\phi$  is true or  $F\phi$  will be true.

So first, we are going to take the OBDD for set of state where  $\phi$  is true and  $B_{\tau}$  is the OBDD representation for transition system or consider it is OBDD or it may be your ROBDD also. So we are starting with this two OBDD  $B_{\phi}$  and  $B_{\tau}$ , then we are going to take 1 particular this thing  $B_x$  specifically it says that all states of the system which representation analogue this BDD. So we are going to take one BDD where it is going to replace in the another steps, it is not called as a step transition system but this is the all the steps OBDD representation of all the steps.

So this is basically analogue to I am saying that initially taking that  $x=s$  and their steps basically and we are going to repeat this particular procedure now where we are doing in the previous case that repeat until  $x=y$ . So we are going to repeat this thing until that  $B_x = B_y$ ,  $B_{\phi}$  so  $B_{\phi}$  is assigning to  $b_x$  that means you're are keeping the previous into the formation. Now we are going

to use this particular operation, we are going to use this particular apply+ because what we have to do what and all we are doing.

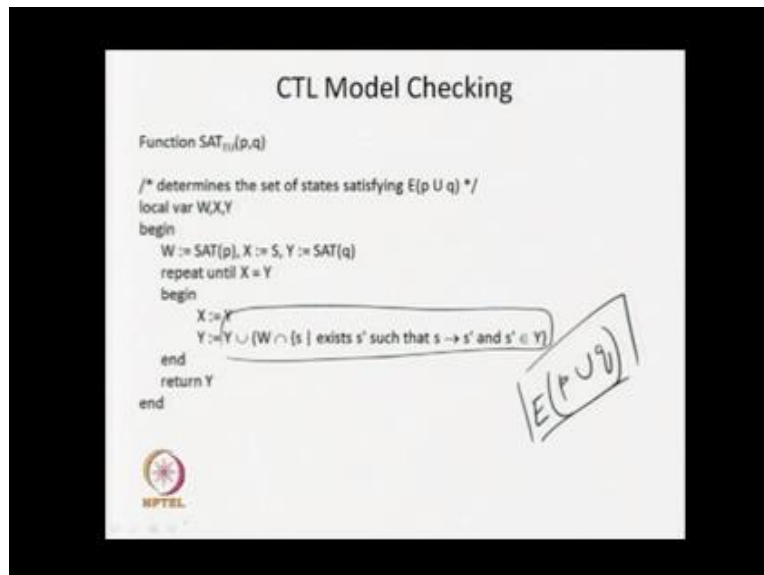
We are going to perform the union with some new more steps okay. Now what are the new steps coming, so this is basically  $U(\text{union})$  and  $B\phi$  is the set of steps which correspond to this particular  $y$  set of state where  $\phi$  is true and now this is your AF in all part in uses that mean we need three for all. So this is basically, we are going to calculate B for all. So this is basically we are going to calculate B for all  $B\phi$ .

And how to calculated already we have seen that we have procedure to evaluate the there exists  $B\phi$  or  $x$ . So use this particular operation to get the B for all  $x$ . So that is why I am just simply writing and not writing the entire expression so we are going to look for all the predecessor state from all the trenches are coming into to this particular subset, where  $\phi$  is true okay.

So now I am getting a new set then again I will open and again I am going to see set of state we have whether it is equal to  $B\phi$  or not so the new  $B\phi$  if they are equal we can terminate if it is not equal you will have some new more steps like this particular case you are having. Now you are going to check predecessor of this particular new step, so we will again enter to this particular process. So see that this is the simple conversion of the model checking algorithm to this symbolic model checking algorithm where we are using OBDD's represent at the system at the set of states where the particular formual is true and applied to it.

So similarly, we need one more procedure where we are having set  $E(PUq)$  so we have a existed part P remains true until q becomes true. So  $E(pUq)$  what is the precedor we have already discussed it. So what we have to do, we have to look for p remains true until q is true. So this is the expression or this is the evaluation that we get actually what is W, W is a set where p is true that means you have to collect those particular states and intersection with this particular new state that we are having the condition particular state and p must be true.

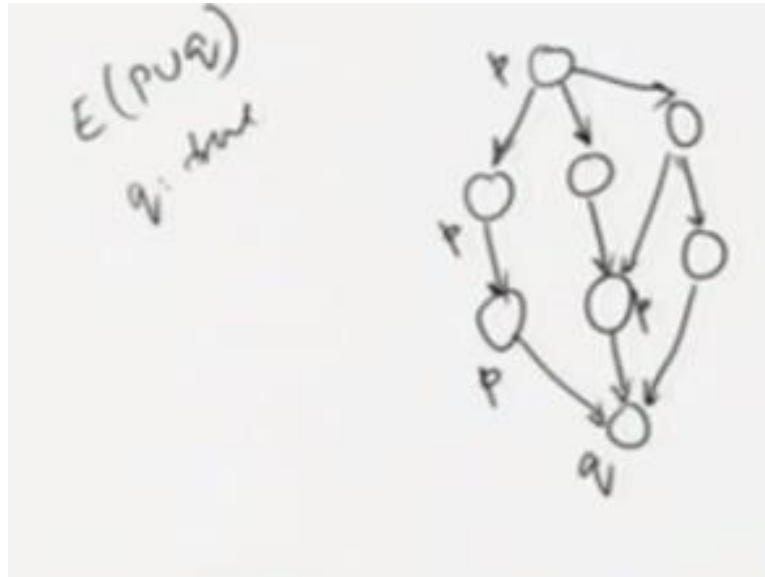
(Refer Slide Time: 38:36)



So basically you just see the condition this I am drawing something like that, see if we are having this thing then what even pass we look for this particular state where we are looking for  $E(pUq)$ , so we are going to look for those particular state where  $q$  is true okay. Since  $q$  is true over here I can say that  $E(pUq)$  because it is a for all semantics that presents in a of, now found this particular state what happens we are going to look for a predecessor state, so we are going to get particular predecessor state.



(Refer Slide Time: 39:41)



So in this particular case,  $P$  must be true and in next step  $E(pUq)$  so this is the scenario that we are having,  $W$  it is where the set of state where  $P$  is true intersection with the particular states where next state  $q$  is true. In this particular state what will happen, in this step what ever we are getting over here then you will find that this particular  $P$  is true and next the  $E P$  is true. So we can say that  $E(pUq)$  is true here also we will get an  $E(pUq)$  is true.

But in this particular state  $P$  and  $Q$  is not true because  $P$  is not forwarded, now I am getting some more steps, we are going to look for the predecessor of those particular new steps. So now I am going to get this two state, again with the same property and same constraint that  $P$  must be true over here that  $W$  intersection of those particular things. We must have a condition to some state of this particular state  $y$ , and  $y$  is the new state we are constructing so again we will find  $E(pUq)$  will be true over here.

But it will not be true at that particular point okay, now when we will come back to this things again  $E(pUq)$  will be true. So this is the way that we are going to evaluating  $P$  and  $Q$ . Now this same procedure we are going to implement symbolically, now what we have to see again similar to the previous one.

(Refer Slide Time: 41:30)

**Symbolic Model Checking**

```
EU(Bψ1, Bψ2):
  BS: OBDD for all states of the system. // Analogous to
  "X := S"
  Bψ1: OBDD for set of states where ψ1 is true. // Analogous
  to "F := SAT(ψ1)."
  Bψ2: OBDD for set of states where ψ2 is true. // Analogous
  to "T := SAT(ψ2)."
  B→: OBDD for transition relation.
  repeat until Bx = Bψ2
    Bx := Bψ1 // Analogous to "X := T;"
    Bψ2 := apply(+, Bψ2, apply(+, B→, PreS(Bψ1))) //
    Analogous to "Y := T; W := { s ∈ S | exists s' (s → s'
    and s' ∈ Y)};"
  end
  return Bx
```

*E(ψ<sub>1</sub> U ψ<sub>2</sub>)*

Now it is basically  $E(\psi_1 U \psi_2)$  so we need two BDD's represent the state of state where the  $\psi_1$  and  $\psi_2$  are true, so we are using starting with this two BDD  $B_{\psi_1}$  and  $B_{\psi_2}$ ,  $B_{\psi_1}$  is the OBDD representation of the state of state where  $\psi_1$  is true and  $B_{\psi_2}$  is the OBDD representation of the state of state where some  $\psi_2$  is true. So these are the two things we are getting then we are taking and their state transaction system  $B_{\rightarrow}$ , this is an OBDD representation of the state transaction.

Now as per where about  $\psi_2$  is true then  $E(\psi_1 U \psi_2)$  is true also in this particular state so I am starting with this particular  $\psi_2$  and you are saying that actually making  $B_x = \psi_2$ , so initially I can construct  $B_x$  and all the steps of the state first. So initially where ever  $\psi_2$  is true,  $E(\psi_1 U \psi_2)$  be true so we are just getting this particular assignment, so now we are going to compare this things we are going to repeat this particular loop.

Now storing all those particular steps  $B_{\psi_2}$  to  $B_x$ , then we are again using this particular expression so what happens this particular portion will give me this portion. So predecessor exists  $x$  it is giving me all the predecessor steps  $\psi_1$  must be true or we are doing this particular dot operation that is the intersection. This is the state of state is true, so we are doing this particular dot operation, that is all the predecessor that that is the all predecessor state okay.

Now  $B_{\psi_1}$  is the state of state where  $\psi_1$  is true that is nothing but the expression representation of this particular state  $W$ ,  $W$  is the set of state where  $\psi_1$  is true were we are taking the intersection that means it is going to give the those particular two steps. In my previous example, it is returning my basically predecessor step but after doing the intersection I am going to get these two steps .

So this is the operation we are performing, so these are the state where it is true and now I am going to use this apply part that means, now I am going to do the inner operation that mean first we are collecting the state of the state where  $\psi_2$  is true, we say that  $\psi_1$  and  $\psi_2$  will be true on those particular state. Now we are collecting new more state so we are going to use the inner operation ready in this particular state.

(Refer Slide Time: 44:15)

**Symbolic Model Checking**

```

EU(Bψ1, Bψ2):
  BS: OBDD for all states of the system. // Analogous to
  "X := S"
  Bψ1: OBDD for set of states where ψ1 is true. // Analogous
  to "W := SAT(ψ1);"
  Bψ2: OBDD for set of states where ψ2 is true. // Analogous
  to "T := SAT(ψ2);"
  Bτ: OBDD for transition relation.
  repeat until Bψ1 = Bψ2
    Bψ1 := Bψ1 // Analogous to "X := Y;"
    Bψ1 := {apply(τ, Bψ1, apply(τ, Bψ2, BS))}
    // Analogous to "Y := Y ∪ (W ∩ {s ∈ S | exists s', (s → s'
    // and s' ∈ Y)})."
  end
  return Bψ1
  
```

*Handwritten annotations:*  
 - "Problem" with an arrow pointing to the apply operation.  
 - "E(ψ<sub>1</sub> ∪ ψ<sub>2</sub>)" with an arrow pointing to B<sub>ψ<sub>1</sub></sub>.  
 - "E(ψ<sub>1</sub> ∩ ψ<sub>2</sub>)" with an arrow pointing to B<sub>ψ<sub>2</sub></sub>.

So that is this inner operation is being performed by particular apply operation okay. So now we are adding some more steps where this particular  $E(\psi_1 \cup \psi_2)$  is true, like that we are these two state now will again repeat this particular procedure to get this state and this state. So this is the way that we are doing so after getting this particular two state we will perform in this particular

loop like we are going to check whatever is the new states we are taking whether were we are getting equal to or previously collected state or not okay.

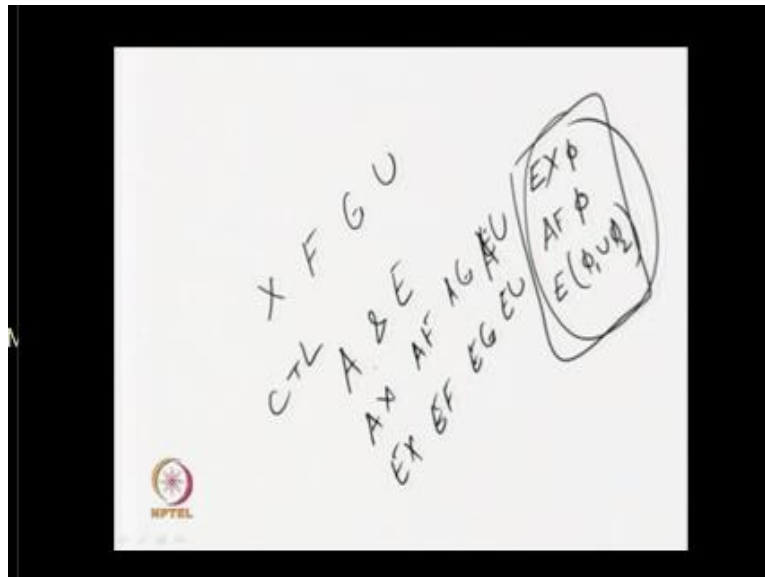
If they are equal then we will terminate not equal we are going to do the same procedure okay, this is similar to repeating this particular steps, okay. So in this work we are calculating it so you just see that always that what are the operator we are talking about that one is your,  $Ex\phi$   $Af\phi$   $E(\phi_1U\phi_2)$  so since we are having the procedure to evaluate these three operator. Now we can evaluate the other operators also because we have seen that this is the operators okay.

(Refer Slide Time: 45:37)



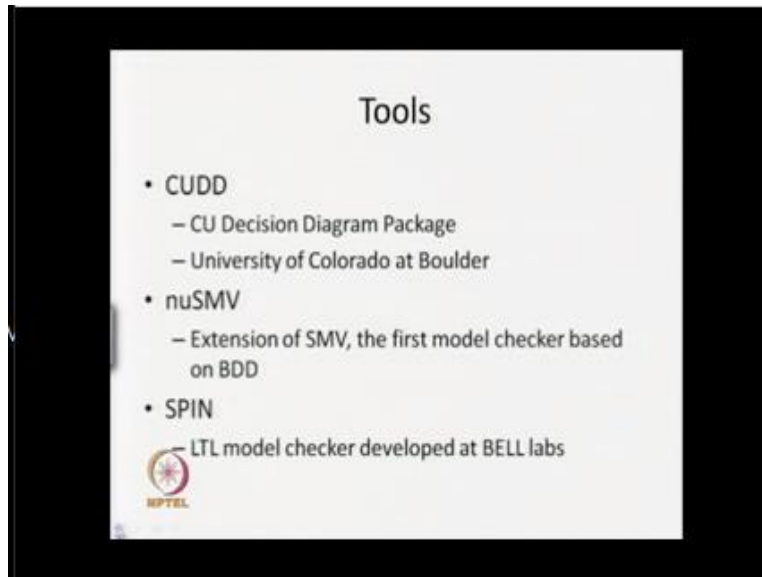
So we need basically 8 operators that we have discussed till now and having port mfile operators, next state user normally until with the respective with this four mfile operators, we will get 8 CTL operators with part A & E all part and their existed part.  $Ax$   $Af$   $Ag$   $Au$  and similarly  $Ex$   $Ef$   $Eg$  and  $Eu$  so out of that if we are having this three procedure for this three operator others can be eliminated. So you jus tsee that we are using ROBDDs to evaluate this three procedures okay.

(Refer Slide Time: 46:43)



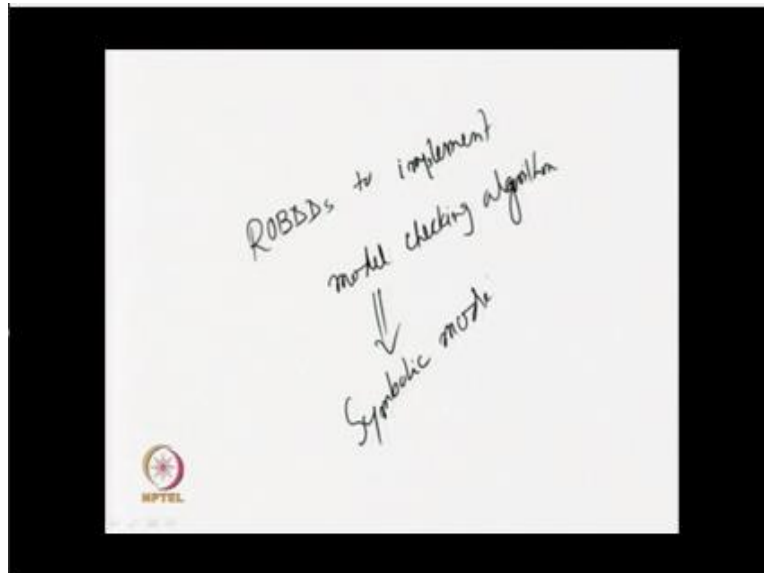
Now we are getting a symbolic models checking algorithm, and what is the advantage you are getting you are representing the OBDD with the help of ROBDD's and in most of the cases we are formed that at a compact representers. Again I would apply on that the sides of ROBDD it depends on the ordering of the variable but to get the exact ordering is hard problem. Already I have mention all those issues, okay.

(Refer Slide Time: 47:21)



Now we have seen that the models checking algorithm are symbolic model checking algorithm forward is so, we use ROBDD's to implement model checking algorithm we are getting this is your symbolic model checking, okay. Now you just see that now we are having enough idea about our model checking of what we can do with the help of but it cannot be done although it has been studied.

(Refer Slide Time: 47:52)



Now in this particular I would like to mention some tools one of your tools is your CUDD, this is basically a Decision Diagram or it is a binary decision diagram so this is better for your BDD is CUDD is developed in the university of Colorado at Boulder, okay. So in this Colorado Boulder is developed in this particular package this is a academic purpose and now what you can do is to download this particular card pickers for me from university of Boulder, university of Colorado Boulder and use this particular package to construct BDD to manipulate the BDD's used it is having all the record algorithm.

Another tools we are going to mention which is a nuSMV ,this is your symbolic model variable pack it is the extension of the symbolic model verifier which is passed model check up on this BDD's. Which is developed by Gan Mackmiom is the student of the and final model checking procedure during is CMU he developed this particular SMV. Now it is being some modification aextension and will released as your new SMV.

So this is your symbolic model variable they are using the BDD to represent and the system and this model, so with the help of these things we can module our system or look for this model also. Another one is I am going to mention this SPIN, this is a tool developed by BELL labs

usually the algorithm module setup and already we have mentioned about what is LTL, which is so we are having it. What you can do you can just try to download this three package and try to walk with this, working is also very simple.

What you need to do you have to or you have to look into the syntax how we are going to give the input to this particular tool, so you have to know the syntax of new SMV how to provide the input and you have to note the input syntax for the SPIN, they are using a particular language called PROMILA, so that you need to know the note the syntax of this particular to give the model of your system and after specifying the operators it is going to check whether the property is true or not okay.

So this is the way that we are doing it, so now what happens now you just try to I am just giving this information to if you are interested you can download the package and activate those package.

(Refer Slide Time: 51:02)



Okay, now what we have seen in this particular case this is basically we are discussing about your system design verification, so you are having a design and after coming up with that design

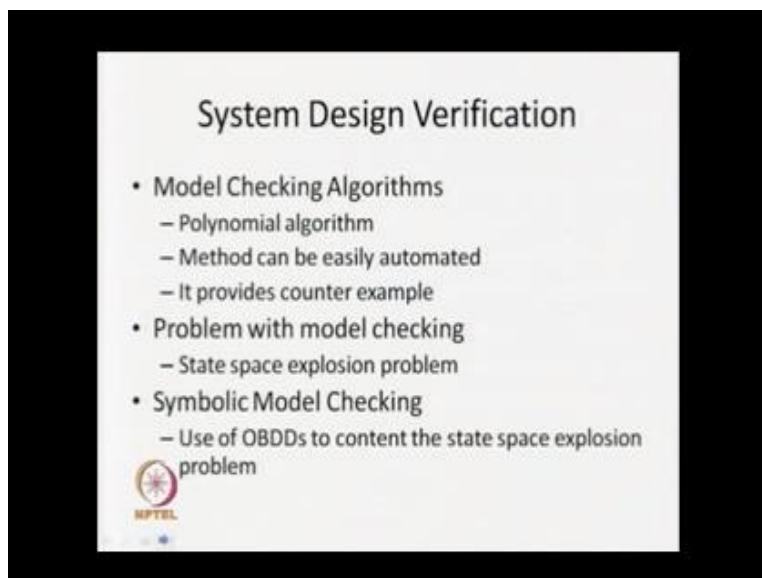


you try to verify or check whether my design whether the properties are true in the in this particular design. In this particular case what we discussed or what we need basically we have to come up with the model of the system design something say simple one or I can talk about that you are going to design for your traffic light controller.

So traffic light controller we have to come up with a model in this model we are basically representing with the help of Kripke structure which is your kind of a finite step message. After that we have to give or specification order property must be satisfied by my model and here in this particular course we are discussed some specification language which is called CTL Computation Tree Logic okay.


We have discussed with this particular CTL, we can give the specification with the help of this particular CTL language and after that we need the verification method by which you are going to check whether the given specification is true in the model that you have given or not. So for that we have used this particular models or we are using the model checking algorithm.

(Refer Slide Time: 52:23)



**System Design Verification**

- Model Checking Algorithms
  - Polynomial algorithm
  - Method can be easily automated
  - It provides counter example
- Problem with model checking
  - State space explosion problem
- Symbolic Model Checking
  - Use of OBDDs to content the state space explosion problem

 NPTEL

Now after that when we look into the model checking algorithm what we have find or what we have observed that we are going to get an polynomial algorithm of 1 model checking algorithm this is the beauty of model checking algorithm. If I talk about the model checking algorithm particularly well said that this is your CTL model okay. In the CTL models we are getting the polynomial algorithm, so which can be manageable handeked and the method can be easily automated.

Now since we are having an algorithm which is a polynomial time algorithm so it can be automated so that can be automated. And already I have mention about some tools like NUSMV SPIN like structure and another advantage of this model structure that it provide contarism, if I am giving a model and I am giving a specification of property we are using this particular model check up property is to it is going to say that yes.

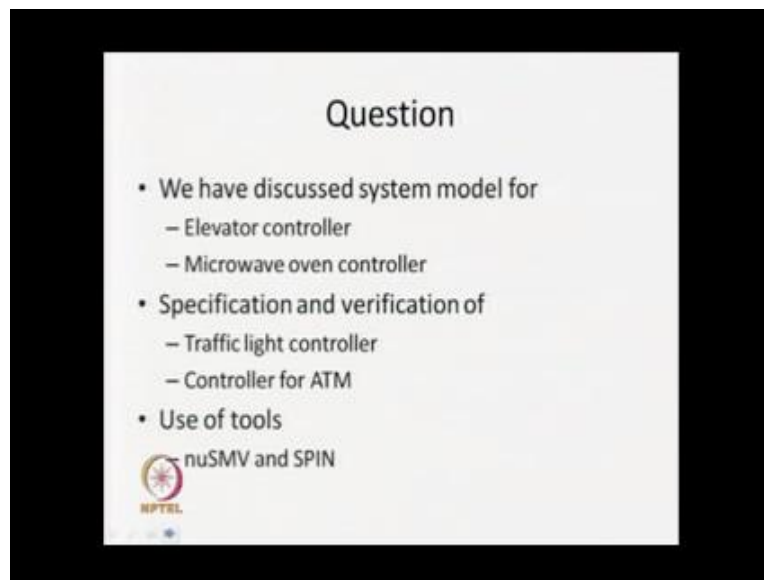
But if the property is not true it will give me a counter example If an execution stress and it is said that if you follow this particular execution stress and your given property is passed. So this is of some sort of feedback mechanism it is keeping some feedback designer team. Now designer team can concentrated on those particular execution stress and they can fix the part okay.

So this is the advantage about this model checking algorithm. Now what we have seen, we have seen that we are having some problem with this models check up. What is this problem is express the problem of express. Already that I have mentioned in the state space of the system model that we are going to get is exponential with respectve to the number of variables that we have.

Already I have mentioned that to the number of state variable the number of states will be  $2^n$ , if n is increased by 1 and it will be  $2^{n+1}$  which will be exponentially in nature, how to restrict this particular state expression is a problem or how you are going to work with the regard system. So for that we have seen how compactily be represented and how it can be represented, what we need we have seen that the structure BDD can be used in this particularOBDD can be used to represent the diagram and most of the time we get complex reaction or the particular steps.

So by using OBDD we are coming up with a another model checker called this is your symbolic model checker okay. In case of symbolic model checker we are using OBDD's to replace and we are somehow contenting that they are restriction the to the particular state of express. So these are the things we have discussed in this particular state okay.

(Refer Slide Time: 55:19)



Now question what I am said if you look back or lecture then what happened we have seen how to design a elevator controller okay, we have seen or we have discussed a simplified model of a microwave oven controller okay, these two we have discussed in our course of our lecture. Now what we can do since we have the model you know what are the properties is you need to be verified already I have mentioned something along with that the simplified one of it.

Simple one that we have discussed about your mutual exclusion of your problem or the sources, these are the things that we have discussed in our course of lectures. So now what we can do that I have mention a about some tools like your nuSMV or SPIN, now you try to module this system on those particular tools. So already I have to discussed about the model try to convert this particular model to your input language of your speed on nuSMV and you see how you are going

to write the specification for your particular and check those particular properties this is an example and question I am giving you to download those particular tools install in your machine and try to check all those particular models are going to walk.

Now I am giving some of the problems also what I am talking about that traffic light controller, many other time I have discussed or I have mention about this particular thing. Now you take up some complicated system complicated poor network and try to design a control of particular junction. So that means we need a controller, so you try to come up with a design, come up with a model and come up with the properties that we need to this particular controller.

Another one another system you can talk about the now it is most of the controller of the ATM automatic teller machine. So how it is going to work you must know and try to analyze it and find out that come up with a basic model. So you try to come up with a module of this particular ATM system and after coming up with the ATM model then you try to find out what are the properties that need to be satisfied this particular model.

Now use those particular properties so that is why I am saying that use the tools nuSMV or SPIN in this is the field available purpose. Now you can look for the designing or modeling of this particular system and see try to prove the properties of the particular system in the particular case what will often you will be knowing how to use this particular tools also.

(Refer Slide Time: 58:11)



Now what are you looking into it this is some sort of your designing of your digital system in this particular course we are talking about design test of your system. Already I have mentioned that basically if you look into the design cycle you will find this particular step, first you have to come up with the specification of the system then we have to come up with an design once you have come with a design we are proposing further and the system and check whether this system.

Then we are coming up this particular verification methodology going to check the properties in our model, once you are coming with this particular, once you verify this property and this is satisfied that system is going to work correctly, then go for the implementation after implementation what we have to do of a particular testing whether my implemented basically it is going to find and next step is your installation marketing and after that you have to maintain you have to go for the upgradation. So these are the design cycles steps that we have seen or we have discussed that these are the steps when we are required when we go for the designer system.

(Refer Slide Time: 59:32)



And in this particular course, this is about the particular digital VLSI design if you see that in this course we are talking about the design of the digital system and if you look into it find that we are having three parts of this particular course. One is the design issues, second is the verification issues and third one is the testing or testing of your design. So in this course you are having these three issues.

In the first module of the particular course we have discussed about the design issues how we will proceed to designer system, so this is the first part of this particular course and second part we are talking about the verification so this is the second part we are discussing till today. Now in this particular part we are talking about the verification issues, so what we are doing over here, we are coming up with the design while you go for this particular design and coming up with a design.

You are getting a model, after getting the model we know what are the properties it must satisfies so we will come up to the specification and we will apply some verification methodologies to check whether this properties is true or not, so in this particular part we are discussing about this

particular verification issues of how we are going to do go for verification and how what we are going to do.

And all the issues what we have discussed and in this particular part we have introduced one particular technique called model checking we have seen how we are going to use this particular model checker and what are the problems and we have introduced about you are symbolic models also. And that third part of this course is your testing and the next part we are going to talk about this particular testing.

Once my application is over how we are going to test the system that before using in the market this is fault may basically in testing what we are going to do we are going to do about the going to find out the fault and we are going to release the fault is to the market. So this is basically course implemented course completion. So next class onwards we are going to talk about the testing of the VLSI system okay. That is all, Thank You.

**Centre for Educational Technology**

**IIT Guwahati**

**Production**

**Head CET**

**Prof. Sunil Khijwania**

**CET Production Team**

**Bikash Jyoti Nath**

**CS Bhaskar Bora**

**Dibyajyoti Lahkar**

**Kallal Barua**

**Kaushik Kr. Sarma**

**Queen Barman**

**Rekha Hazarika**

**CET Administrative Team**

**Susanta Sarma**

**Swapan Debnath**