INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

NPTEL
NPTEL INLINE CERTIFICATION COURSE
An Initiative of MHRD

VLSI Design, Verification &Test

Dr. Santosh Biswas
Department of CSE
IIT Guwahati

Module IX: Combinational Circuit Test Pattern Generation

Lecture II: D-Algorithm-1

Okay, so welcome to the next lecture of module number 9 that is on D algorithms. So, about the last few lectures, what we have seen is that for testing generally the two types of automatic test pattern generation algorithm if you look at it basically. So that is random test pattern generation algorithm and other is by path sensitization that is why actually fault sensitizes, propagate and justify.

So by path sensitization method in the last lecture, we have discussed that they can be two types of them. So one is by Boolean difference and one is by the D algorithm that is by path sensitization, propagation and justification. Also then we have discussed that this that Boolean difference I am sorry that is that Boolean differentiation and all these for the very difficult problem in terms of complicity.

Because if you go for Boolean difference then this the you have to go for derivatives and all these, so it will lead to and the number of these patterns that may result can be exponentially natural. So due to this exponential large complexity for this Boolean difference based scheme or the based algorithms, we have generally shifted to sensitize, propagate and justify approach. Also what the few lectures I mean over the last few weeks, we have mainly discussed that we have given several examples of sensitize, propagate and justify in which case, also in the last lecture we have seen one example.

In which case what we do we actually this is a stack as 0 fault we apply a 1 and vice versa for stack at for the other way. And then we propagate the value to some output and then we try to justify that that propagation and sensitization are justified by some important values. So this was actually basically the D algorithm. D algorithm is nothing but, sensitize, propagate and justify.

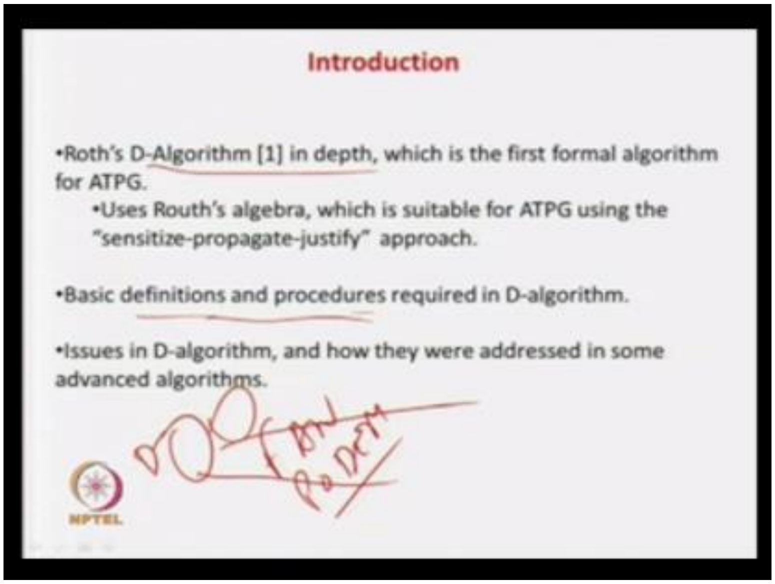
But in these two lectures we will actually have a detail look at them there is a very formal way defining the D algorithm. Because D algorithm has been seen is one of the most prominent or the background or the back bone of digital testing, because in next will see sequential circuits and several other BIST architectures and all, but for all of them D algorithm is the back bone, from D algorithm they have different other advanced versions of D algorithm like podium and fan.

But the basic bottom line or basic backbone, which we should learn in detail is the D algorithm, because it was the starting point for all sensitize propagate and justify the based D algorithm which still occupies the prominent role in digital testing. So in these 2 lectures what will do is that we will try to look at this algorithm here more formally. So in the last lectures as you give one example that for a given that circuit say gate like say if there is a stack at 0 fault then we apply a one.

Then we see that in normal case the answer is one and the fault case the answer is 0. So we could have written this as a, as it could be easily solve our automatic test pattern generation problem but, we say that to represent it formally we have to say that we represent it by some kind of a symbol like D. So that because in Boolean algebra we cannot have A oblique B or 0 or 1 or something like that.

So we have gone for a five value growth algebra when you have symbol like D, D’ X and all. So that is what is representing the circuit algebra or that fault or the circuit having faults in a more formal way. Similarly, for the D algorithm or sensitize, propagate and justify with approach till now whatever examples you have studied and all with the somewhat informal way. In today’s lecture and in the next lecture 2 and 3 of this module will see this in a more formal way that is our basic approach okay.

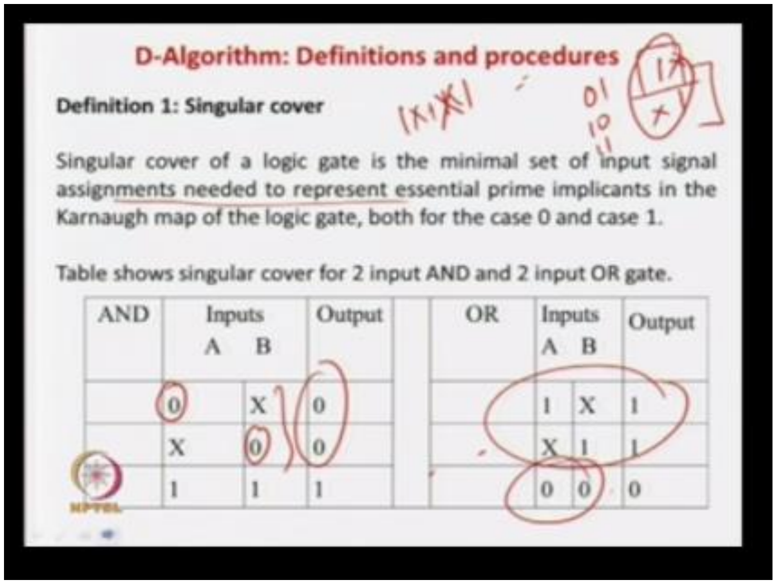
(Refer Slide Time: 03:35)



So first we have already seen about these Roth's algebra, so I mean that Roth's algebra will be used for this what you called this D algorithm. Then before we start discussion on the D algorithm, we will see some basic definitions and procedures that is very important, then we can go for formal algorithm. And finally, will see that some adverse algorithm like podium and fan so this we will see the basic of them; they were all advanced versions of D algorithm.

But as I already told you that D algorithm was the basic back bone from there we have developed so many advanced algorithm, but this starting point was D algorithm for all these sensitize, propagate and justify their approach so it forms the background. This is what is the basic plan of the next two lectures.

(Refer Slide Time: 04:18)



So first before starting D algorithm as we told that will go for some of the definitions and procedures. So first definition is the singular cover okay, so what is the idea, so let us first look at the definition and then I will try to explain you, singular cover of a logic gate is the minimal set of input signal assignments needed to represent the essential prime implicants in the Karnaugh map of the logic gate both case 0 and 1 okay.

So now let us see what does it mean, so let us take the case 0. So let us say that, what do you mean by prime implicants already we have seen in our discussion during the design part of the course, because you know that this course is that design verification and test. So in design module of this course, we have discussed in details what is a prime implicants? So we say that whenever we have a Boolean function represent it as equal to say AB+CD’+F something like this.

So if you say that prime implicants then this one should not be contained in any of them. As well as it should contain an implicants which is neither in this nor that is this contains some mean term, which is in neither of the other implicants of the function. So this is very informal kind of way or study, but already we have discussed it in details in the design model. So in other way of this speaking that this implicants is the is mandatory to be present in this function representation, because it has the prime in term which is neither in this neither in this something like this.

So it has to be present or it is actually sometimes called the essential prime implicants. Then it is mandatory to be present, but in case of prime implicants what is the idea that say it is like AB+CD or something like that plus F some function. Then it should not be contain directly in any one of them, like if you can say that we have function like AB+B sorry AB+B, then what does it mean that AB+B.

Then what does it mean that AB is actually contain AB, because AB is actually AB+A' implicants. So this AB is contained in B, so then we can say that AB is not required because B actually represents in case of AB. So this is some of the definitions of prime implicant and essential prime implicants which we already discussed. So basic idea is that you have to first select the prime implicants.

And then say for the case 0 then we have to find out that is the minimal set of input assignments you get to represent the essential prime implicants in case of 0 and 1. So let us take the case of 0, say in the case of AND gate okay so AND gate 0. Now you see how we can get 0 in case of an AND gate we can get the 0, if their answer is 00, 01, and 10 so these are the cases in which case we can say that the answer is 0 for AND gate.

So now we can represent this by 0 x or x 0 be and, because this 10 and say sorry, I am in these are the three cases, which represents by the output of the AND gate is 0. So now if you want to represent these as an prime implicants case okay, so we can write it as X0 and 0X, because this 00 included here these also here in this case it is 10 is included over here. So this comprises 0 1 and sorry this 0X incorporates 00 and 10 correct.

And if you take the other case, that is, if you take the case of 0X, then it comprises 00 and 01. So basically this 00, 01, and 10 three you do not required to represent, because of the case that it will no longer this are prime implicants case. So we can represent it as X0 and 0X, so it sorry it

X0 and 0X. So, if that is actually you can say, if you think terms of A and B then you can say that it is B'+A' kind of a thing you can write okay.

Because this is 0 and the other part do not care that is XAB'+A'B this other part is is hyphen or which is do not care a part of a thing okay. So that is why we represent X0 and this 1. So, because if you take these two combinations are automatically in these two cases we have 00, 01 and 10 these three bits, these three are actually included in this two cover. So, now we can see that this two actually form a singular cover for the case 0.

Because you do not require to represent this one, this one, and this one. Because these two is comprised over here that is 0X and this 1 actually 00 and 10 are actually compressed in this formula. So, this two actually 0X and X0 becomes the singular cover for 0 of an AND gate for the OR we know that is always the 2L so OR gate output is 1, when we have 01 or 10 or 11, so it is better to represent it has 1X and X1.

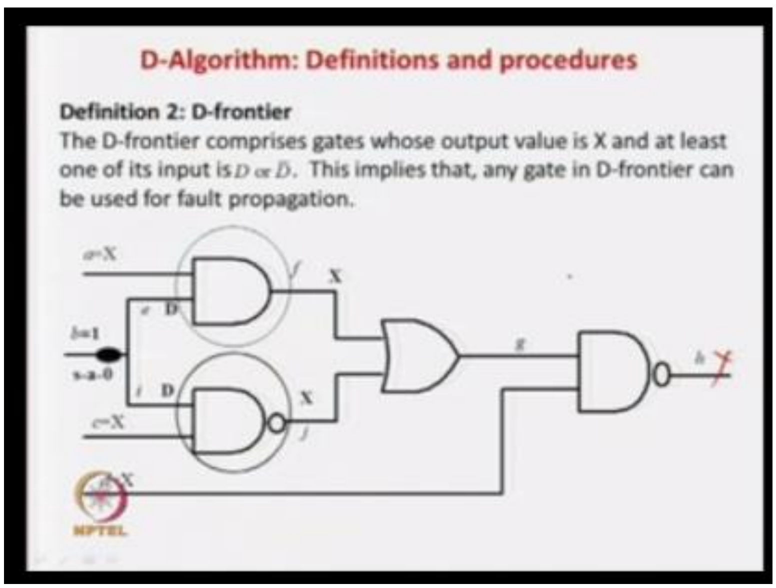
Because these two will actually form the essential cover, because these two are prime implicants, because this is neither contains in this this is neither contain in this. This is 1 have one element at least which is not containing the other. So, that is 1X+X1 actually if you write a function like this that is say for the OR gate you write it as A+B that is actually B that is A/-B that is actually we can better write in this way then the function is something like F=A, then we do not write anything, then plus do not write anything then B.

That is actually 1 in this 1 you do not care and this is do not care and x. So, this are two if you take comprises all this three cases 01, 10 and 11, which is representing 1 in case of a OR gate. So I mean this two actually forms the singular cover for 1 in case of an OR gate okay. So, now if you look at it for the 1, if you look at it for the AND gate when you get the cases of 1. So, this only one combination that is 11 when you get the one the AND gate and 00 in the OR gate.

So, this will actually from this singular cover for the OR gate and this is for sorry this will from this singular cover for 1 in case of AND gate and this is for the 0 in case of an OR gate there both well official. So basically what actual is singular cover says that, so if you want to get the output 0 in case of the AND gate. So, you can either use this or you can either use this. Similarly, this is for 1 and in for 1 in case of AND gate this is the combination required and for OR gate 0 this is the combination required.

That is actually singular cover this is minimal set of inputs and this is actually in other words, we say that what is the minimal set of inputs to represent output 0 at AND gate, you can say that only one input is 0, that is the minimal set and you can also say that input is 0 other is do not care. So, these are the two minimal sets which represents the output to be 0 in case of an OR gate and we can be extra polluted for OR gate. So this is the basic definition of a singular cover.

(Refer Slide Time: 10:43)



Now, we go for what is second definition that is called D frontier. So, all this definition we are looking at will be required in the formal representation of a D algorithm, so now what is the D frontier. So, we know that D basically was introduced because the represents 10 that is normal case, fault case 01 that is actually we can also the D' normal case and fault case. So, this was actually Roth's algebra, we have already seen the represents that was actually used for representing the fault in the circuit.

Now see let us see, what is the D frontier. So, let us look into definition say that the D frontier comprises gates whose output value is X, we do not know the value X and at least one of it is input is D or D’ that is we have a gate something like this. So, this is the either the D or may be D’ whatever and the output is X, if the one other input, I am not talking about the other input. This implies that any gate in D frontier can be used for fault propagation.

Let us now look at what is meaning what does it is a definition, because this is one of the most important definition or you can say notion to be used in the D algorithm. It says that for example, you have a gate like this where OR gate AND gate like this. So and this is say initially all the nodes are say X, X, X okay we do not know say for the time being okay. Now we say that this is D that means, what normal case 1 and fault case 0.

That means this line is capable of propagating the fault output. Now if I know that this output is X; that means, as of now I do not know the value. That means, till we have a scope that this value can be D can be propagated over here if you apply a 1 over here; that means this AND gate is capable of for fault propagation. But on the other hand if you see there is a gates something like this say for example, this same AND gate we have D over here and the output here is 1and the output here is it is not X, it is not the x.

That means, what now this is 1 this also become sorry, this is 0 I am sorry this is 0 and this is not X this is not X, because if it is if it is 0 then this will be 0 and not X this is this will not be X and what is the case and D cannot be propagated over here. So, what does it means again in the D frontier can only be used for propagating the affect of D fault that is D or D’. Let us see this example of this circuit, so let us know that this is a stack at 0 fault here.

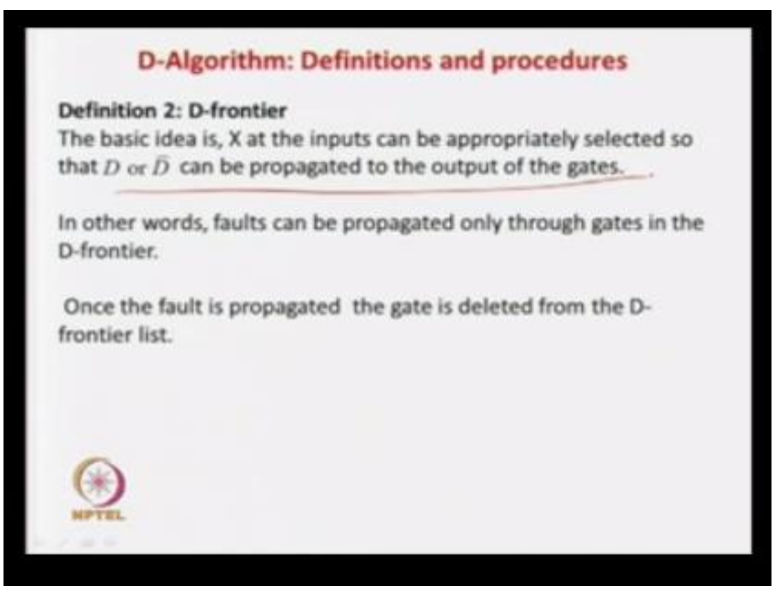
So, obviously we have sensitize you put a 1 over here now what do you do, so one more thing is that whenever you start circuit analysis to make all the input and all the net lengths as X initially okay. So all the lines as X, because we see that we do not know the value. So, only whenever we start moving the value of the X we will try to fill that from X to or we can say that 10 D or D’ as we start computing the values, so it is a stack at 0, so your input is 1.

So, this is D because normal case 1 fault case 0 so, it is a D over here and D over here and we know that input is X, this input is X, this input is X because as of now we have not computed the value of X. So, the output of this AND gate is X, because if it is a X, if his is the X, if is 1 then you get the value here D and if this is X is 0 then you get the value 0. So as of now we do not know the value of X.

Similarly, this a NAND gate, so if the C=1then we get a D' here if C=0 then we get the 1 here, but as C is here and X over here, so this value is also X. So, these are the two gates because the output here is x and x. So, these are the two gates which actually comprise the D frontier that is this fault affect can be propagated by this one or by this one. Because they are the actually comprised comprises the D frontier that is this gates will allow you that this gate 1 or 2 you can say comprises the D frontier, because they can allow you to propagate the fault.

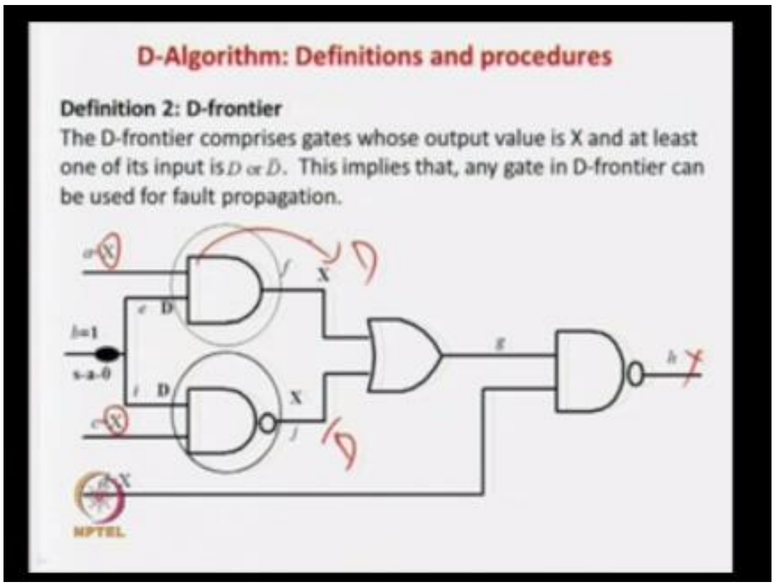
But maybe you think because of some reason or the other that has become a 0. If C has become a 0, then immediately this will not be X it will be 1, then 2 will not be in fault D frontier because D cannot be propagated through this gate number 2.

(Refer Slide Time: 14:30)



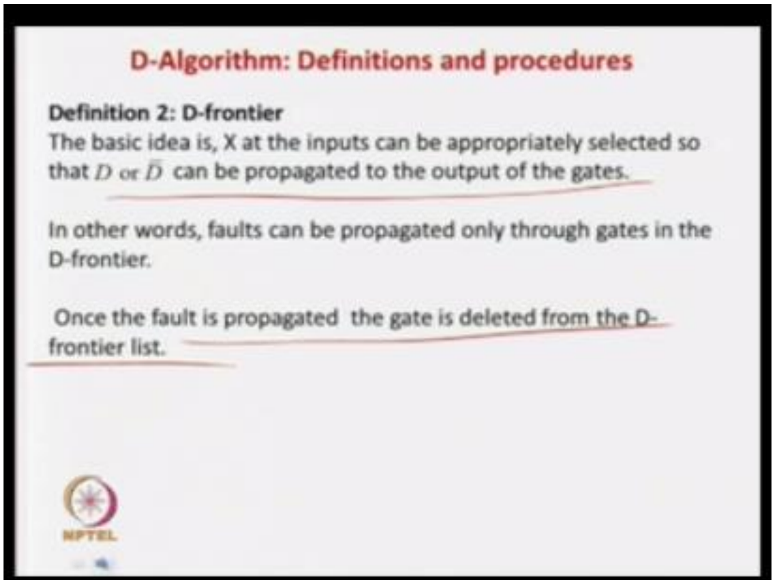
So, that is what actually we say that whenever I gate is in the D frontier then only it can be used for propagating the value of the output of the circuit I am sorry I mean fault effect of the circuit. That is what is the basic idea, the basic idea is that X at the inputs can be appropriately selected, so that D or D' can be compared to the output of the gates that is the basic idea that is.

(Refer Slide Time: 14:45)



So this X can be controlled in such a way, so that these effect of D and here D' can be propagate the output of this gates which comprises the D frontier.

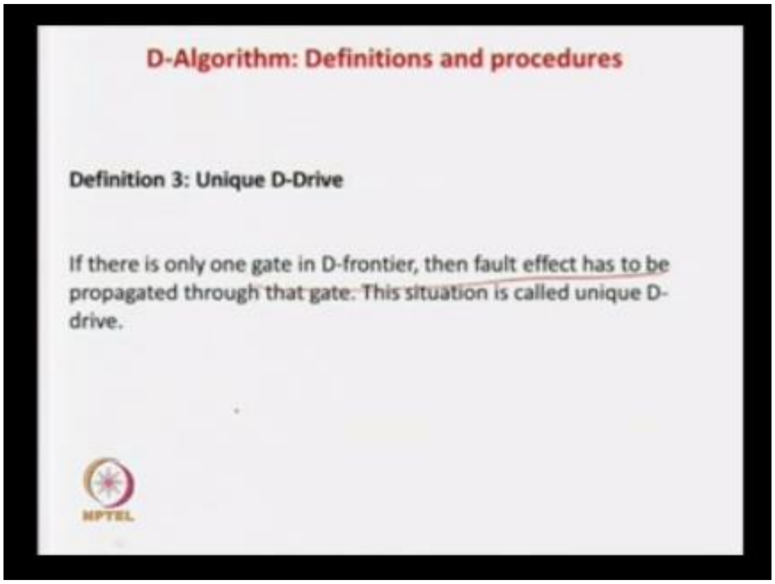
(Refer Slide Time: 14:54)



So we can if there is a gate in the D frontier then we have a scope that we can state this value approved lately or accordingly. So, that the fault effect can be propagated through the gates in the D frontier. So, once it is very important, so once the fault is propagate for that this propagated that is say fault is propagated by making this as one then this one will not no longer be in the D frontier.

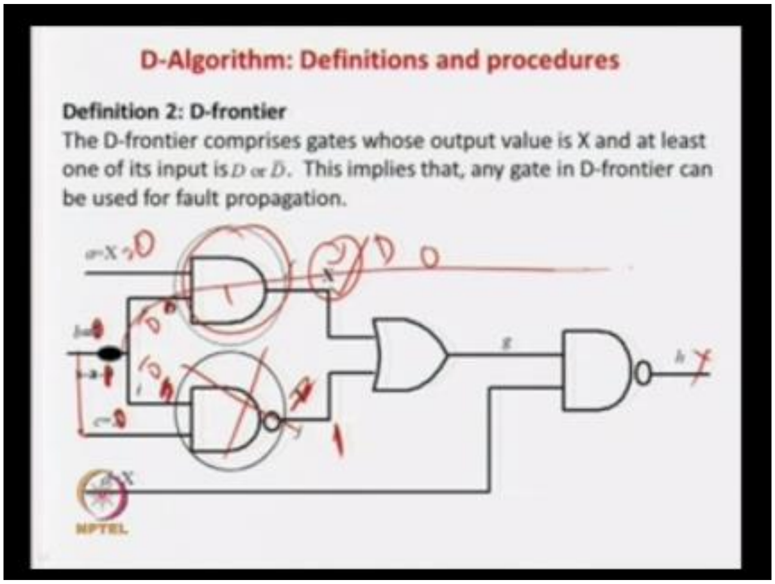
Because already you have used this gates and the fault has been propagated. So, this cannot be again used for propagating and the fault effect. So, it can be it is to be detected from the D frontier that is the basic notion here this is second definition okay.

(Refer Slide Time: 15:29)



Okay, now, there is next definition, definition number 3, so that is called unique D drive. So, what is unique D drive, then is actually called if there is only one gate in the D frontier then is actually, what do you called a unique D drive. This situation because it can be then propagated through only one input.

(Refer Slide Time: 15:45)

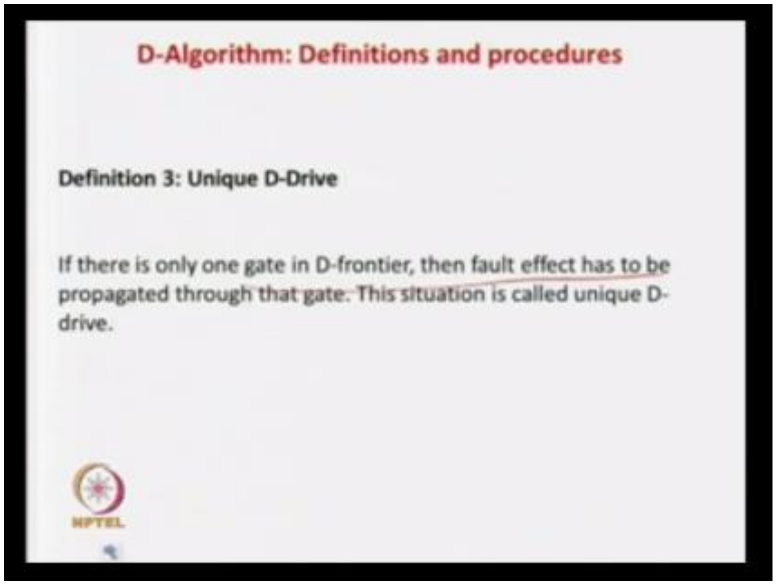


So, let us just see, what is a unique D drive the definition is very straight forward say for some example, say for due to some reason or something like this let us say that these two lines were short some for some reason, then what and let us consider this for this stack at 1 or let us change this one. Then what happens to sensitize stack at 1 would apply a 0 over here. So, the moment you apply a 0, so C will be 0.

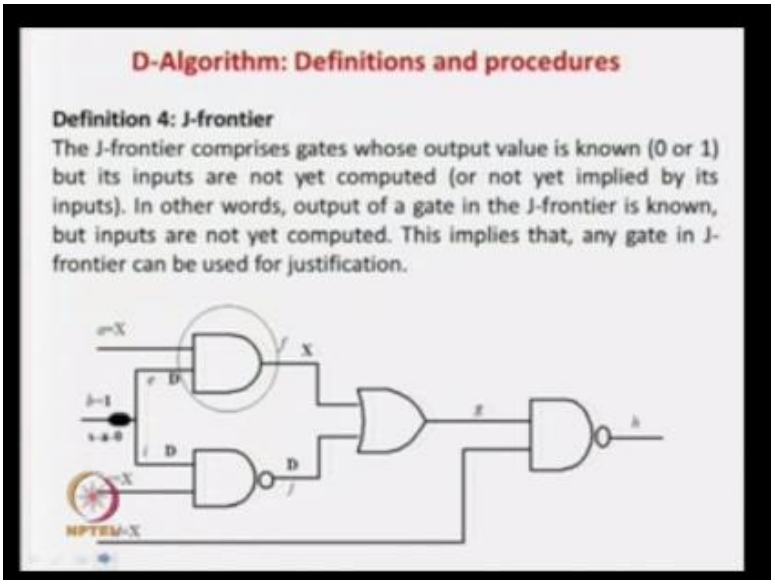
So moment you apply a 0 you will get the 1 over here but, in this case this is it will not be D, it will a D' and this will be a D', because normal case 0 fault S1 and then if you affect and this will be still 0, so this s 1, because this is 0. Now, this AND gate output is still on X, because if you apply X=1 then you will get a D over here, if you get a 0, you will get a 0 over here. So, there is a scope of propagating the fault this 1.

But there is only one gate that is only this gate is D frontier, this gate is not in the D frontier, so it has the unique gate from which the fault effect has to be propagated. So this situation is called the unique D drive.

(Refer Slide Time: 16:41)



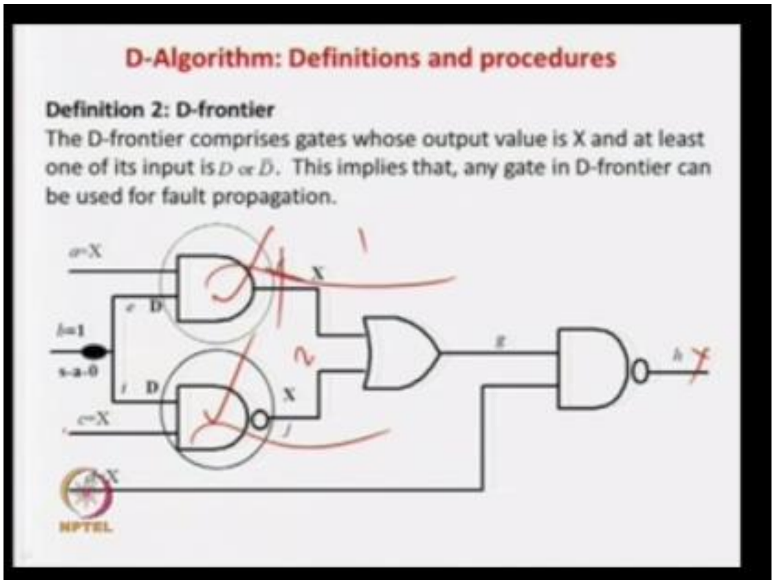
(Refer Slide Time: 16:43)



There is only one D only one gate in the D frontier which can be used for propagating the value of fault effect. So, if for fault unique D drive then you will not have to think for the choices just like I mean we have seeing the last example that always or with also be looking in details in this in this lectures or next lecture that D algorithm like sensitize, propagate, and justify is always not a successfully gate.

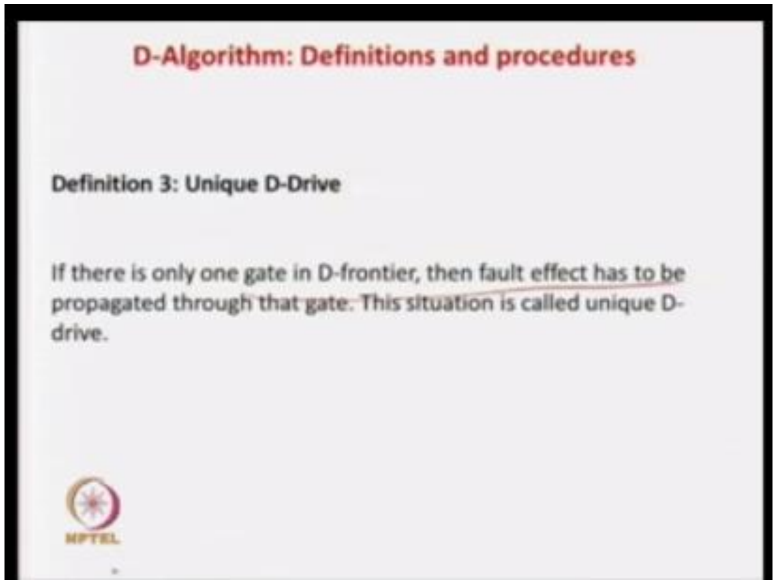
Sometimes you will if there is a like in the last example we have two D frontiers like this one if you gone for the general case.

(Refer Slide Time: 17:14)



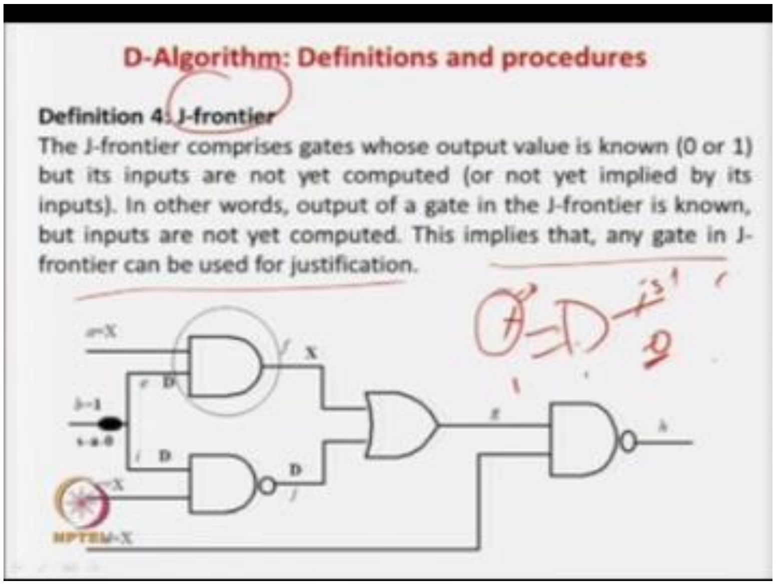
That is if you have just gone for this non modified case ten what is the idea. So, if I gone for non modified case then you have these two gates in the D frontier this gate and this gate in the D frontier. Now it is up to you, whether you see that this gate as the D frontier or this gate has a D frontier to propagate the value of the fault. Now, once you have selected this gate for all propagation then this is related. But there is option one and option two.

(Refer Slide Time: 17:35)



So, sometimes one or the option give you correct result correct result means inconsistence that is you know inconsistency, you can finely propagate the fault effect to the output as well as you can also justify this. But, for some of the selections of D frontier or there is a multiple choice available like in this case of two gates. So, you may land into a problem, because the propagation in one of the gate you may land to a inconsistence situation that is fault can be propagated but, it cannot be justified.

(Refer Slide Time: 18:01)



So if there is a unique D drive then there is no such option to think about the uses only one drive. So, you have to just drive the value using the AND gate, so unique means there is no choice at that point. So, now let us look at the next definition that is definition four is about J frontier, so that J frontier that is mainly used in D algorithm one called the D frontier and one is J frontier.

So, basically you remember this way, so D frontier means what that is you have propagating the value of the effect that is D effect from input to output. So that is what is basically you are doing. So, if you are using a D frontier algorithm, so D algorithm may D frontier definition if you think, so it is breaking the value of some D or D' that is fault effect and you are trying to propagate it to the output by gates having the output is X okay, so that is input to output.

Now, this actually called the propagation, so the D frontier is basically related to propagation, then something we know that to justification. So once you say for example, propagated the value up to here okay, now also that due to this say that this length is length and these are properly justified, so that the propagation and sensitize are successful.

So, J D algorithm and sorry J frontier as we see now is basically related to the justification procedure and it works backward it works for primary output input, so backward direction. Now see let us first read the definition and then we look the meaning and implication so the D frontier what we saying. So, frontier what we are saying, J frontier comprise gates whose output value is known either 0 or 1 but, it is input or not yet compute or not yet implied by it is inputs.

So, in other words the output of a gate in the J frontier is known but, inputs are not yet compute that is it implies that any gate in J frontier can be used for justification. That means, what say for some reason you know the value of this 1 is to be 1 or 0 1 whatever you know that I have to get at this 1, so you should not write 0 oblique 1. So, you know that I must get the value of 0 in this I must get the value of 1 in this as something like this but, now this is known but ,from that this is already known but, this 2 inputs are say x and this is also x the inputs are not yet computed.

But, not yet implied by the input that is the inputs are not yet know or at least one of it input is not yet known, at least we know they say 0. So, this we know that 0 for some but, this is x that means to justify the value of x here. We know that x is to be completed to 0 that is, in these gate what happens the output is known either one or both of the inputs are still x. That means this gates still as the scope that is 0 can be justified over here. So, let us will take this bigger example later we just derive the simple gate and sustain the concept.

So, let us take this case, so we say that the output is here is 1 and with for the time being let us assume that inputs are x. So, we know this gate can be in J frontier because you know the output is the 1 now for make this successful we require this 1 to be 1 and this 1 to be r 1 okay so because this may be for requirement. This may be requirement because see it may be drive in some sachet 0 fault. So, it is massed at just that it fault can be sensitized. So, this x equal to 1 and this x equal to 1 you can make that it can be successful.

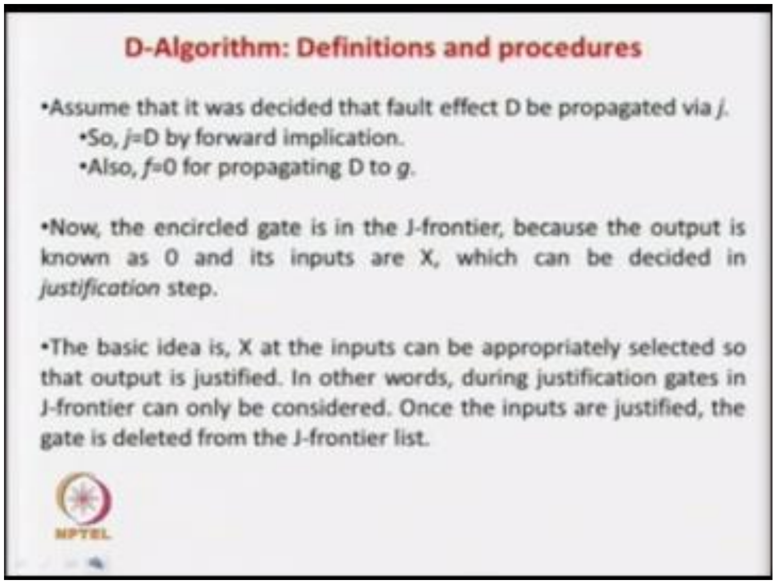
Okay now, let us see for some for reason both these inputs are 0 and 0 1 for some reason you can think because of some other integration some other set this 0. Now, this is 1 and this is not x and this is also these things are not x, then this cannot be the J frontier. Why this cannot be the J frontier because this is sachet 0 and this, inputs are not x so that means there is no scope that you can control this gate. So, get the value of a 1 over here. So, this for this gate we justification procedure is stopped. In other words say for example, they can have a gate like this, this is a sachet 1 and this is 0, you have to apply see you know this is x and this is 1.

Okay so these gates still get J, frontiers because we know that still this x is with you can control this x and you can apply 0. So, that you can successfully control the gates justifies this value. So, there is one 1 x that means, sometimes it can be possible to justify and sometimes it may not be like. For example I mean get in the J frontier always does not mean that it will be successfully justification okay but, it may be like in last case it was not but know in this case sachet 0, so you have to apply a 1. So, this is 0 and this is x for some reason this is 0, because of some other reason like fault sensitization and some other case because of some other I mean in this some other inputs are there some reason this is 0.

So this is x. So, this is in J frontier because this is the x. So, you may think there is a scope that is the x can be control, so that the output is 1. So, you try to apply this as 1 but, because this may be input is 0. So, you can never get a 0 over here. So, this gate is in the J frontier but, it is not successfully allowing you to propagate I mean justify the value of the output of the AND gate. So, gate is in the J frontier, so at least one input is 0, 1 input is x of the gate but, it does not just imply always to be successful in well you called justifying the output of the gate. But, at least over inputs are not known the output is known that is output is known means that is the requirement.

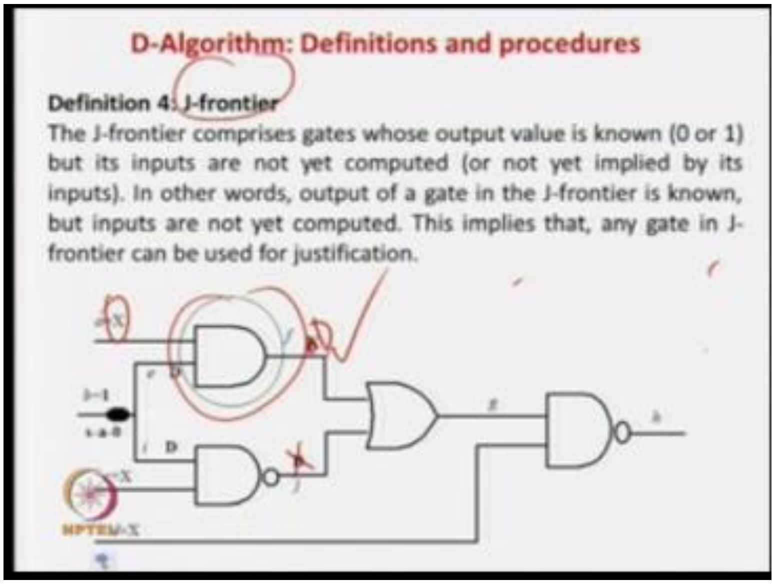
And inputs are not known that we have to compute that inputs from where from where the requirements of this justification algorithm you have to find out that 1 also. So, if you have gate output is known but, inputs are not yet known and they have to be computed. So, you can think that gate can be used for justification, so that gate faults in the J frontier. In this case say what do you say just take again example.

(Refer Slide Time: 23:26)



This case if you see what do you want to over here. So, output of the gates are known say for example.

(Refer Slide Time: 23:27)



In this case say if you see what you want over here so output of the gates are know say for example in this case let us think that this is 0 D, because this gate D has to propagated over here correct. So, also this gate has to be propagated over here. Now, if you look at this gate, so what do you know we know that in this case say for example, because of the D frontier so, this initially both of these variant D frontier so somebody has selected that this should be D frontier and not this.

So let us keep it has an x, so we are not yet selecting. So, let us the things that is we have been selected has the D frontier this is not in selected. So, once it is selected for the D frontier. So, what going to happen, so this 1 will be x will be converted to D and because it non inverting gate and sorry this is not D this is x because getting not selected by D frontier for the fault propagation this one. Now, what happens, so in this last just look at this gate so, output is known to be D 1 of the inputs we know is to be D other input is x.

So, this gate will lie in the J frontier, because now this is D this is D and we need that we need to find out what is DX. That means, this were going backward and still these gate has a scope or justifying the output of D over here. So, obviously, you know that x equal to 1. So you select x equal to 1 then this input actually justifies the propagation of this one, so this gate falls in the D J frontier. So, what we have done initially we know that if just look at this case this is also x let us think, this is wrong for the type of is x actually.

So, initially everything is x so you apply sachet 0 1, so D and D over here. So, see that this for the D frontier this one also D frontier this one is also D frontier because in these two gates you can propagate this value of this D through this suppose both of them are D frontier. So, let us select this one as the D frontier. So, once you do that the value of D coming over here, now this x is gone these two remains in the x because this is not selected as a the D frontier now once you selected the D from the output is known.

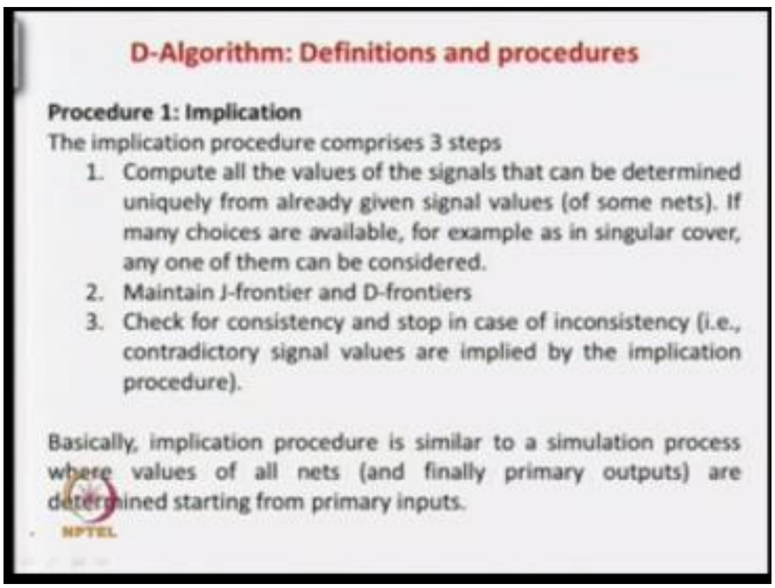
So once the output is known this input is x. So, you can basic idea for J frontier you can select this appropriately this will be 1 or 0 whatever the case, so that this output is justified over here. So, this actually this gate as the input is 1. So, it is actually you can used for justification of your approach. So, this actually faults in the J frontier, if actually this is actually gate is taken as the D frontier. Okay so the output is know and input you can compute for sometimes they mean all right should not think in that way, that this gate is in the J frontier.

So, it can be used for propagation that is Ø or sometimes gate naught also taking in the J frontier can do your job, because of some other reasons of some other inputs say that these are automatically D comes out. So, in this case what happens this in J this gate may not in the J frontier but, still you have actually it is justify your outputs. So, this is also be in D case, also for some times it may also happen that this x is 0 for some reason then this case J cannot D cannot be propagated over here in that case, this does not remain in a J frontier and it does not serve your purpose.

Okay so a gate being in D frontier you can say many things so this gates in D frontier that means, you can say that I can use sorry this gate is the D frontier that means, you can use this for justification but, gate in not D frontier does not imply much. So, gate not in D frontier say a equal to 1 then it will help you to justify automatically this is 0 the gate is not in the D frontier because as it exist 0 or 1 already predefined then both inputs are known and output is also known.

So, you got result in the J frontier, so sometimes it may help you if it is 0 that will not help you because they cannot be propagate and if it is 1 it can help you because D can be propagated. Then it does not remain in a J frontier, so gate not in J frontier may or may not help you that depends on situation to situation but, if a gate this gate is in a D frontier because this is known and this is x, then if the gate is in sorry J frontier most talking about J frontier. So, if gate is in J frontier then it can be used to find out that whether it can be used for justification or not.

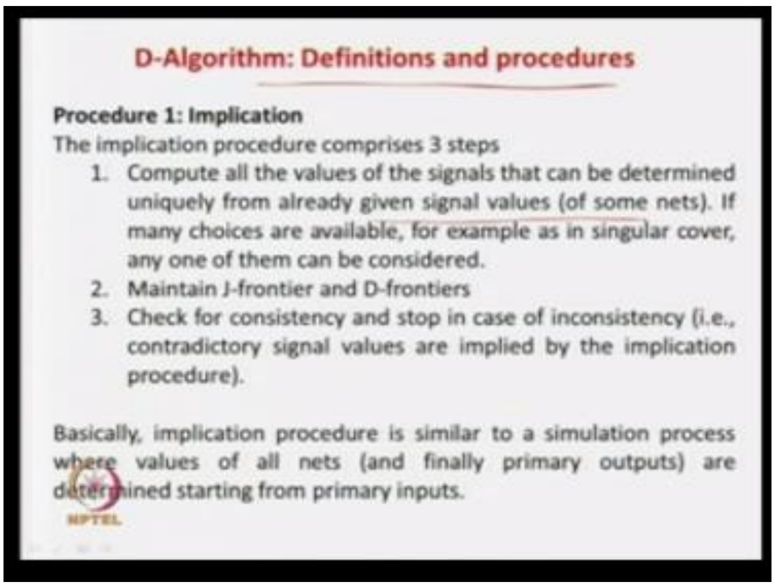
(Refer Slide Time: 27:20)



So, the gate not in J frontier it may help or may not help again the gate is in J frontier you can study that it can be used for justifying your approach. So, sometimes it may be successful, sometimes it may not be successful but, whatever gates are in J frontier that can be used for justifying your inputs. So, what we do basically we propagate the fault, so J D frontiers and then it reach the primary output and then once you reach the primary output default then you have to find out which are the gate which are in J frontier.

Okay and then we have try to justify using the J frontier, so will take some complete examples in next lectures, so which will allow you to understand the concept better. So, whatever which we are saying this D I mean, this last example which we see that assume that this was decided at the fault effectively propagate J. So, that is we are actually affecting the in this same example actually we are may not say same idea is there we may not I mean go for this one this same except the you have discussed about the J frontier and the D frontier.

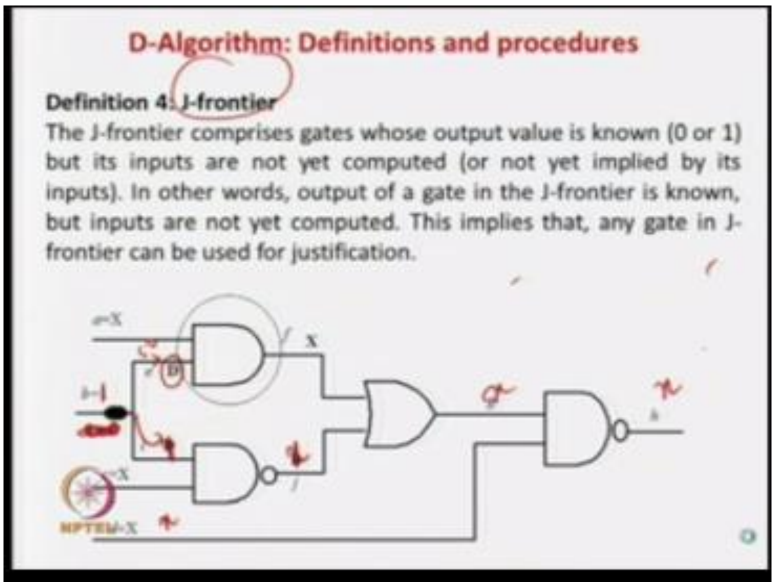
(Refer Slide Time: 28:20)



So, this thing is this I mean elaborated the in a detail manner over here, so this example is detailed over in this slide, so you can see. Okay so now, we go to the D algorithm that is another we are looking here some definition, now we see and some procedure that is sub algorithm or sub states of this one. So, first sub state will be implication. So, implication actually procedure or sub module after D algorithm it comprises three steps compute all the values of this signals that can be detect uniquely from the given signal below some values if many choices are available for example, a singular cover any one of them can be considered.

So, what it is then that is residing in next step then we come back maintain J and D frontiers and check for consistency and stop in case of inconsistency that is contradiction, basically the implication procedure is in is similar simulation process, where values of all nets are determined starting from the primary inputs. So, what basically implication does, so we try to find out compute the all values of the signals that can be determined uniquely from the given signal value of some needs.

(Refer Slide Time: 29:26)



That is for example we take this one okay so that means, what, so initially everything is x initially we you start your design or start your implication procedure and you think this x over there. Then what we do say for example, we apply a D 1 D over here this sachet we know that here 1 is must to be apply now once D is 1 is apply till what we know that immediately this two are D and D okay but, we do not know the value of CD and A because they have to be determine by using D frontier and J frontier but, immediately, we know that this is D and D.

So for example, also if you think that there was another say also let us take the other way, so let us not take the fault over here. Let us take default C here sachet 0 in this net so what happens we know that D has to be apply. So immediately, 1 D as apply we know that this is D, so this is the implication procedure and also if to sensitize this we apply a 1 similarly, we know that this value will be a one.

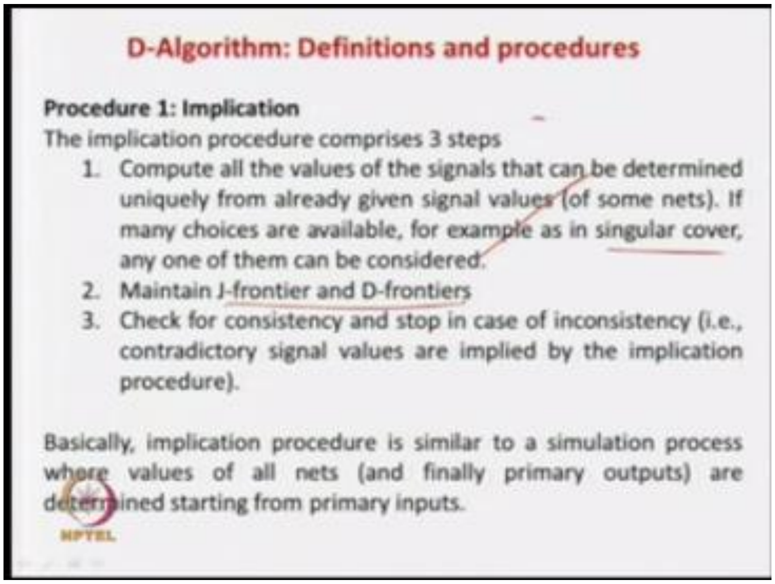
So, that is but, other are cannot compute because others we have to know the D frontier and the J frontier. So, suppose some of the values by the sensitization process some other mechanism. What happens that once you set the values for some reason like in this case say the value of 1, then automatically this is sachet 0, so it will be 0 and this net does not have any fault because we know that in case pin out circuits. So, this net and this nets are actually independent of each other. So, if, the net number I is independent of net number E then actually this value can be directly incorporate from here.

Now, why it can be incorporated because we know that D equal to 1so immediately this I will be equal to 1 because the fault is in the other branch of the fan output. So, this actually comprises your what do you called this form I mean this is implication procedure that is circuit initially we have some all the nets will be set as x. Now, next what you have to do you have to just find out that, what do you have to find out that some of the net values either than directly compute. Like in this case one can be directly computed over here, this was the implication but, the value of F value of G for the value J of a this cannot be computed directly.

Because for that what do, you have to do you have to first find out whether you will use because if you having a sachet 0 fault at E, then what happens then your F this gate actually this is the only gate this will only become your D frontier then you have to actually propagate the value D through F. So, once you propagate the value of the D through F, then you know that this is this then your this gate will become your J frontier and then you have to find out that weather what is the value of x.

So, x you cannot it is difficult to directly get it by implication, because by implication what will do is that, what we do is that by implication you just find out the values which can be known directly okay so that was the first step we discussed that compute the values of all this signals and can be determined uniquely from the given signal values of some nets if any choices are available. For example, you can get the singular cover many choices are like for example, like for example, you have AND gate this output is actually 0 for some other net.

(Refer Slide Time: 32:35)



So, we know that the inputs can be or these implication can be x 0 or 0 x. So, you can take any one of them that is what you say that is the multiple choices available you can take any one of them and this is may be considered. So, then for once you have done this implication then you can also find out that what are the J frontier and what are the D frontiers basically you will find out what is D frontiers that is how you will propagate the value then you can get the J frontier out of it.

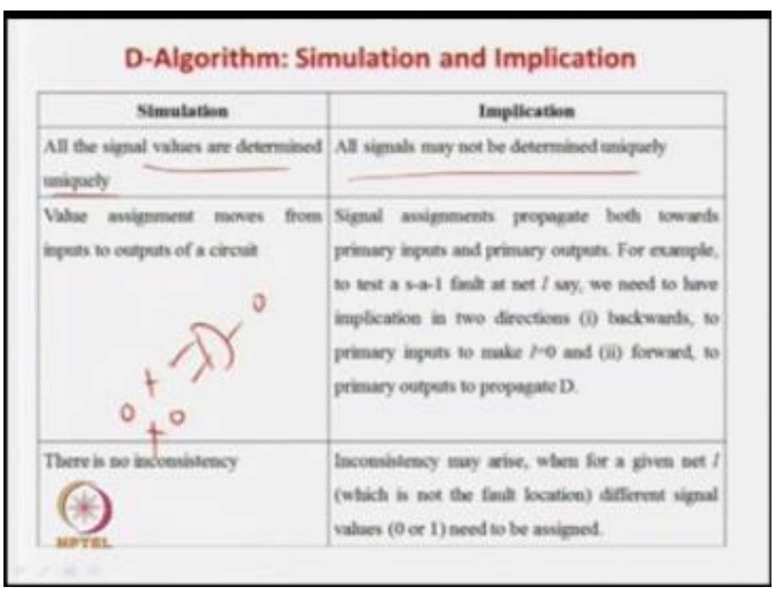
So, if they some inconsistency, inconsistency means that you want the value to be one for some net and because of this, what is the implication and all you get the value to be 0 then actually say inconsistently you have to stop. So, basically implication procedure so this is your procedure with the values of nets or determines it from the primary inputs. So, it just like a simulation, simulation what we do know the inputs and we try to compute the outputs. So, we are seeing in last few in last 2, 3 lectures back even driven simulation we have seen compile code simulation we have seen.

So, in that case what we do we start the, we give some input to the circuit and we try to find out what is he output of the circuit that is the basically the idea. So, in this case also what happens we try to, because they fault in this circuit so, what we do is that, we first find out the sensitize for sensitize fault, then initially all this circuit nets having x then you sensitize default then we try to find out which gates of the circuit can be determined 0 or 1 or D or D' by this value.

So, once some of the nets are been determined this is step one, then if this multiple choice you can take any choice f them like x 0, 1x, 01 00 whatever if some multiple choice are there like 0x, x0 1x, x1 something like this for AND gate and OR gate kind of thing that is by the primary input cover we have seen. So, any once of the choices you can take and then we try to find get the D frontier and the J frontier all this things you can find out then you can go for justification and all.

So, we try to find out as many values of net uniquely as possible. But, having step in maintains the J frontier and the D frontier that is very important and also there is some need you have to apply two values.

(Refer Slide Time: 34:27)



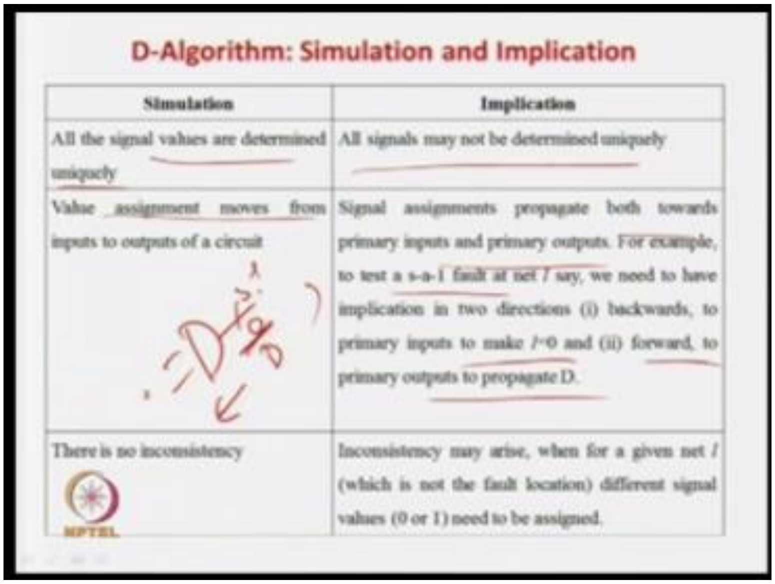
Then I mean once you say that if it is to be 0 and by just I mean what we called implication the value 1 and vice versa. So, you have to stop at the stop at time and say that it is an inconsistency so, this implication is actually this you can called as a very important module of the D algorithm. So, when we will get the D algorithm in detail, so will in more formal we see how it feeds to. So, basically this is the heart of the D algorithm.

That is you first uniquely determined what the value of this signals maintains J frontier and D frontier and then in case of D algorithm you have to add two more points that is if there are n number of D frontiers so you have to take in there are n gates in D frontiers. Then you have to take any one of them and then you have to go for the J frontier justification and if you keep on doing it the D frontier, is the primary output.

So, basically will see that this implication is basically is the heart of D algorithm. So I mean but this slides difference between the simulation and implication let us see, so in each case all this in case of simulation, all the values are determined uniquely. Because these some primary inputs obviously, circuit will have some primary output there is no choice for AND gate if your input is 00 you will get the answer is 0, if your input is 11 you get the answer 1. So, there nothing called x0, x1 all these things are not in simulation, simulation you have simple inputs you get simple output.

But, in case of the implication some signal may not be determined uniquely, why because already we have seen that for a AND gate if the output is 0 so you take x0 or 0x. So, we can also take 00 but, we generate this 00, because there is a problem which we will see later, that we will see later why we will not taking. So, that is one thing, so in case of implication some of the values may be x we are not determined uniquely. So, the value of assignments used for inputs and output for simulation.

(Refer Slide Time: 36:00)



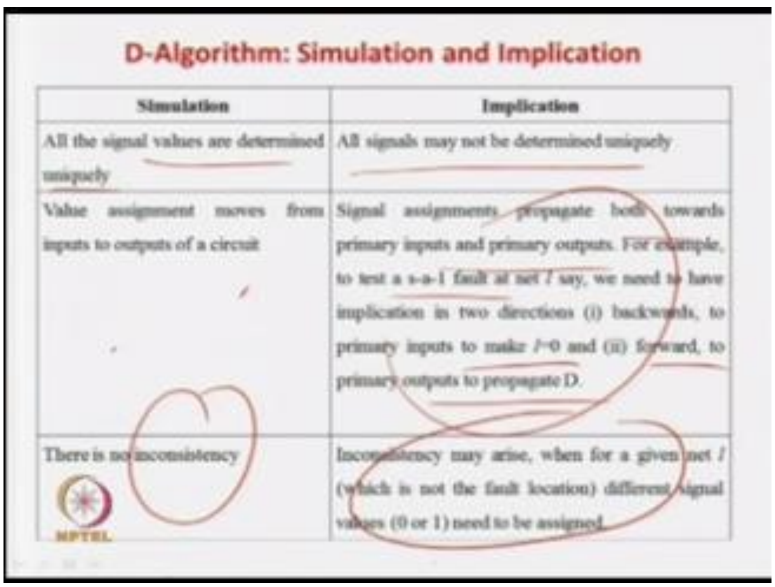
So, simulation primary inputs you give 1011 in the circuit and then you compute the output, so that is very simple. So, in case of implication what we will do it is not a unidirectional flow that will not flow from state this one to input and output. But, there what we will do we first find out the some points we find out some fault, fault is there then sensitize it then we find out the D frontier.

So, D frontier will propagate the value then once D frontier propagate the values some J frontiers may be created, for in J frontier you have to justify the value of I mean justify the inputs. And then again and use a D frontier is stated as similarly, we take the D value the output and in every step one, one step which we are propagating the D value at the same time we are trying to justify also that the D frontier take the values and propagated.

So, also we tried to justifies the values using the input of gates in the J frontier so, it is actually it propagate both ways primary inputs and outputs. For example, take as sachet 1 on net A we have two implications backward to make the primary inputs is 0 and for output is the primary output to D. That is what actually that is saying that if you have AND gates, so this one so, this is actually sachet 1, so you apply a 0 over here. So, this is a forward direction you can say, so because we can considered these to be D frontier.

So, it will be actually D’ in this case because normal case 0 fault case from case sachet 1. Now, again once this is 0 this is D or you know the output to be D that is normal case 0, fault case 1 sorry D’ normal case 0 and fault case so, I mean I am just sorry, so in, so we are talking about the AND gate.

(Refer Slide Time: 37:32)

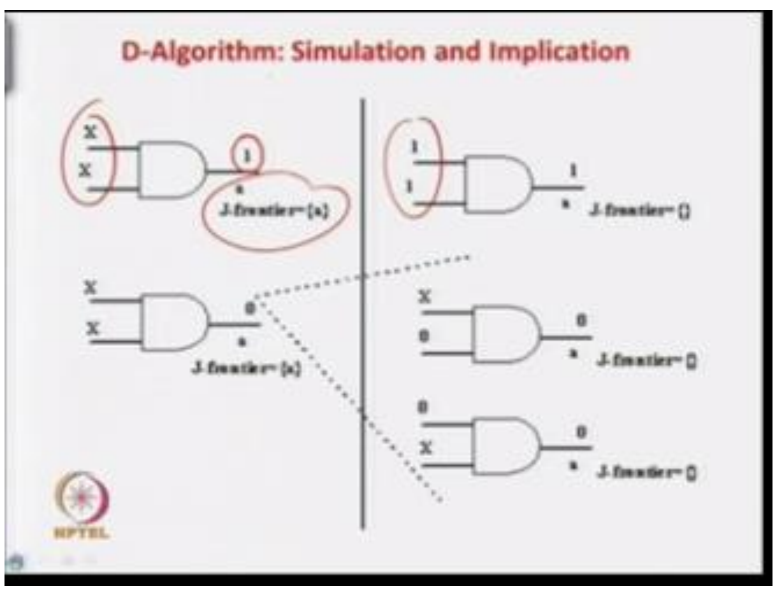


So, in the AND gate this is sachet 1. So, you apply a 0, you need a 0 over here. So, this normal case it is 0, fault case it is 1, so this is D’ so we know the value of the output of D’ both the inputs are x. So, it has to be output should be 0 so we can have x0 or 0x anyone one of them is π. So, that is what saying that in case of simulation we if you have 00 the input of the AND gate we get the output as 0.

So, the unidirectional flow but, in case of implication this is both way, first we have sensitize this faults then this actually becomes D frontier kind of a thing, so you will get the D now again we justify this 1. So these are because it becomes J frontier, so it is actual moving backward and forward, so backward is may be primary input 0, forward to propagate the primary outputs this D' has to be propagate to some primary outputs okay so in case of implication sometimes we have to sensitize the default.

And then move the fault where is the output that is forward direction and backward direction because for D frontiers are done to get J frontiers so you have to take the value backward and this is and that is up to for justification so then this y is the difference between simulation and implication another difference is in case of simulation there is no inconsistency, because you give a input and the output you can determined uniquely right. And there is no question of inconsistency but, in case of a implication then the lot of inconsistency can be there because it is same.

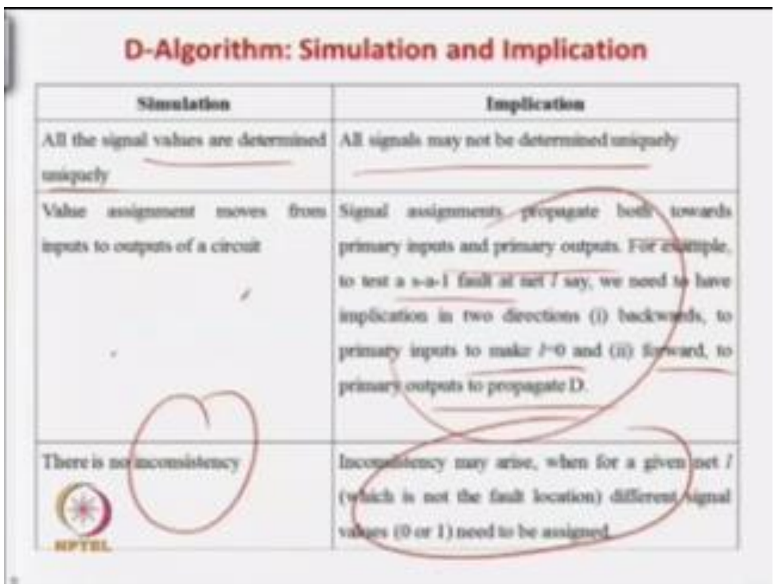
(Refer Slide Time: 38:51)



Net you have to assign a 0 and you have to assign a 1 so same net I mean while you are propagating the fault value then you say that you have to assign a 1 and but for justification case it says that you apply 0 such cases we have already seen in the last lecture also be seeing in details in next lecture so then is your inconsistency but in case of simulation, there is no questions of any kind of inconsistency, because some inputs are there you get the output a unique output okay so now let us see some example of definitions and the procedure you have seen because some

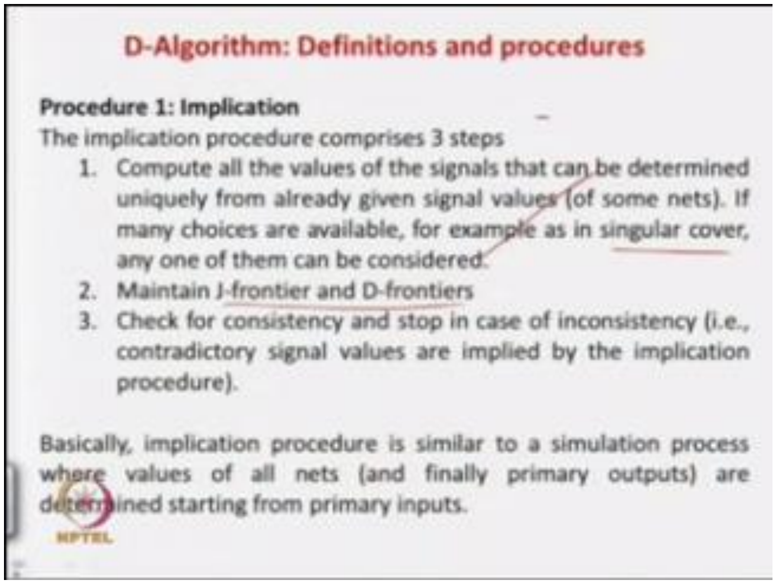
Of bit complex okay so we will just try to illustrate that with in some examples okay so in this case say this is the AND gate so the output is 1 so when the output is known then the inputs are x and x so the output is known then we say that this is actually here J frontier J frontiers means, I already told you that output is known but, both the inputs or single input may be x so we know that the output is 1 both of them must be 1 1 that can be covered from the first.

(Refer Slide Time: 39:43)



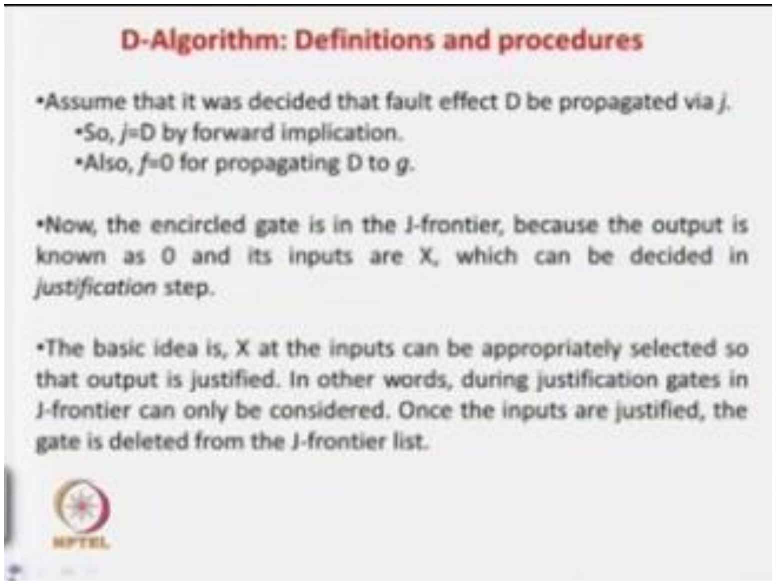
In what do you called.

(Refer Slide Time: 39:43)



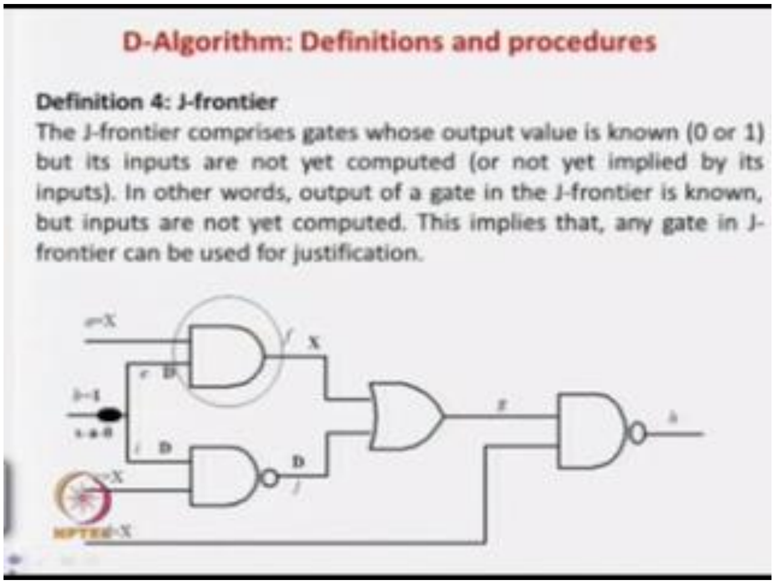
This first.

(Refer Slide Time: 39:44)

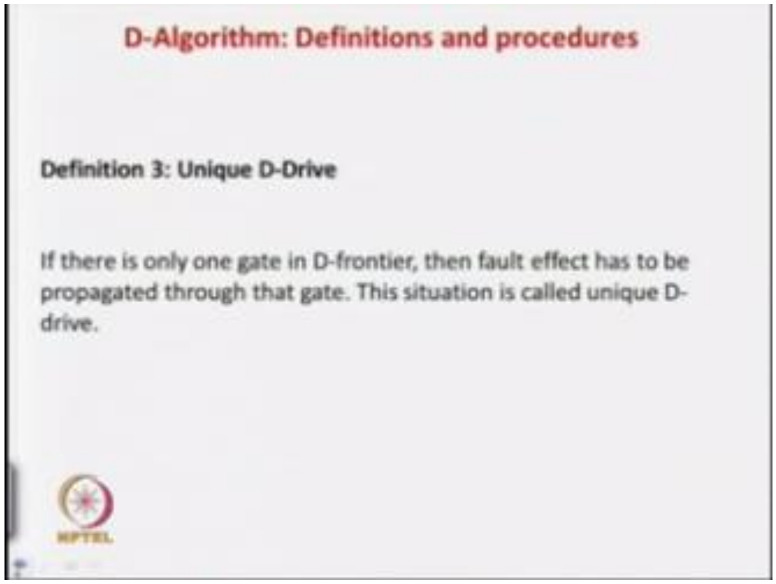


I mean this slide.

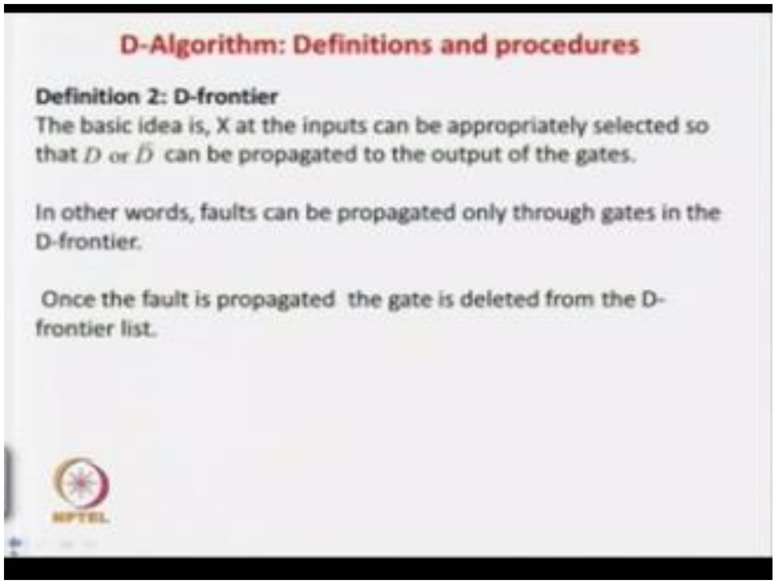
(Refer Slide Time: 39:44)



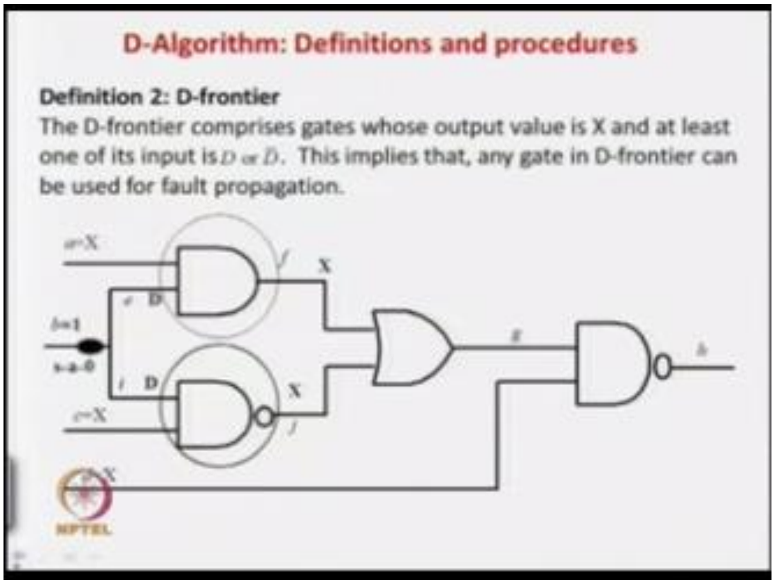
(Refer Slide Time: 39:45)



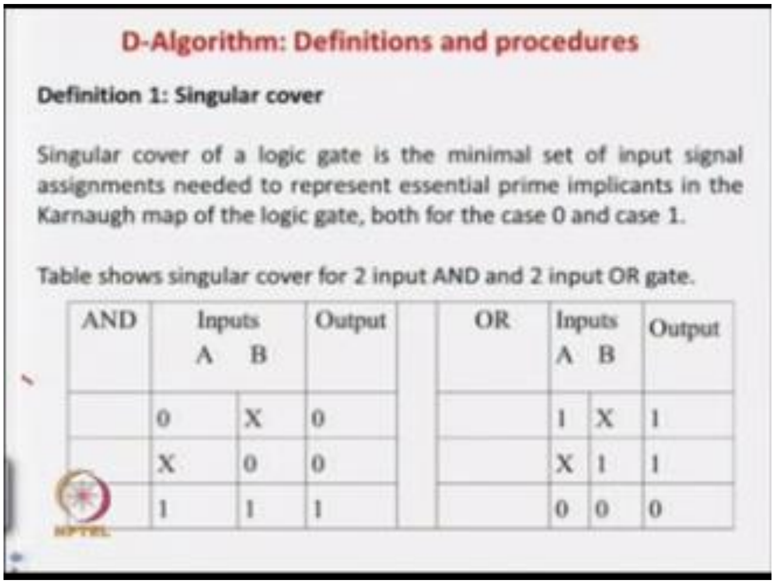
(Refer Slide Time: 39:45)



(Refer Slide Time: 39:45)



(Refer Slide Time: 39:46)



D-Algorithm: Definitions and procedures

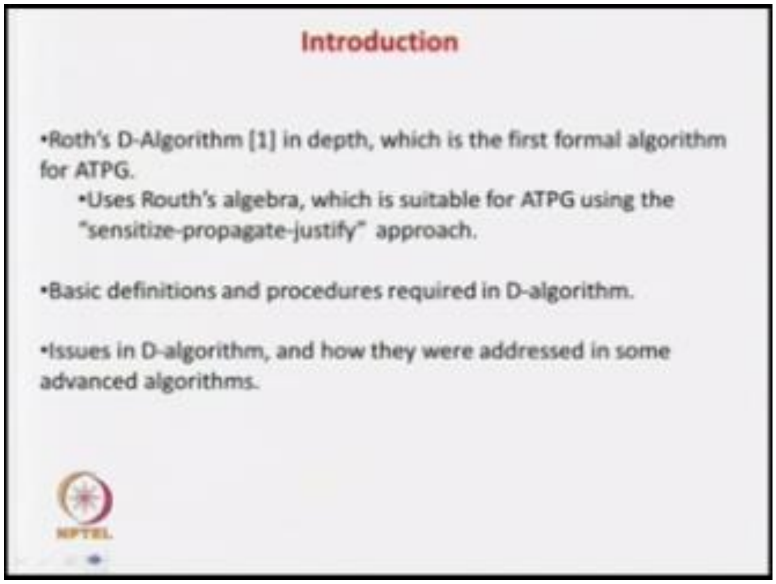
Definition 1: Singular cover

Singular cover of a logic gate is the minimal set of input signal assignments needed to represent essential prime implicants in the Karnaugh map of the logic gate, both for the case 0 and case 1.

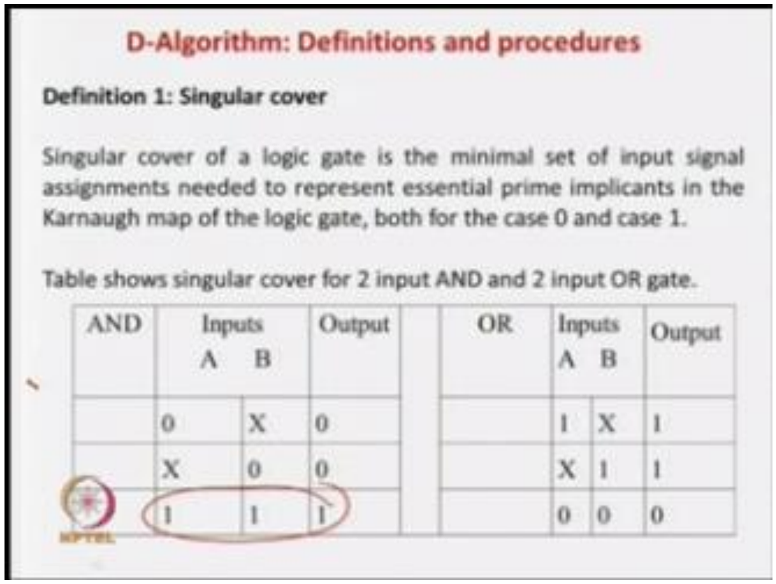
Table shows singular cover for 2 input AND and 2 input OR gate.

AND	Inputs A	Inputs B	Output		OR	Inputs A	Inputs B	Output
	0	X	0			1	X	1
	X	0	0			X	1	1
	1	1	1			0	0	0

(Refer Slide Time: 39:47)

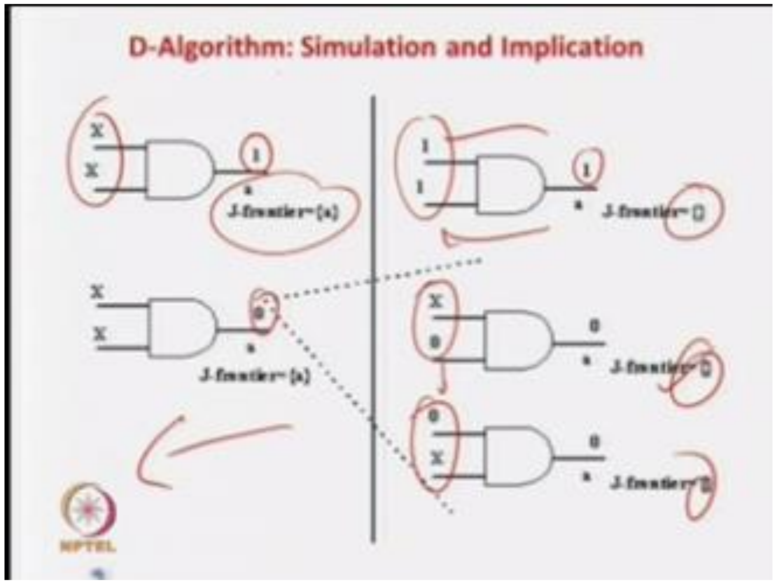


(Refer Slide Time: 39:48)



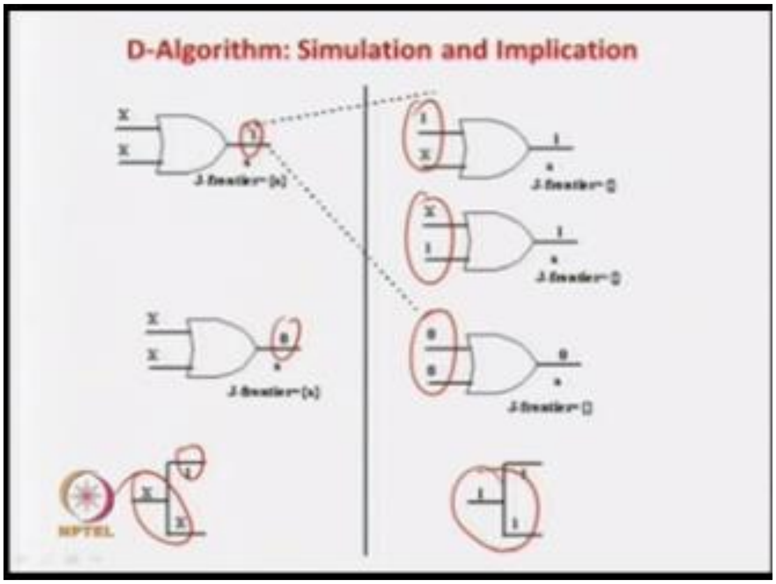
We have shown this second slide that is on singular cover that is 11 in case of an AND gate this is the singular cover so that is actually we have to know this table when you are doing this J frontier and D frontier this nets.

(Refer Slide Time: 39:57)



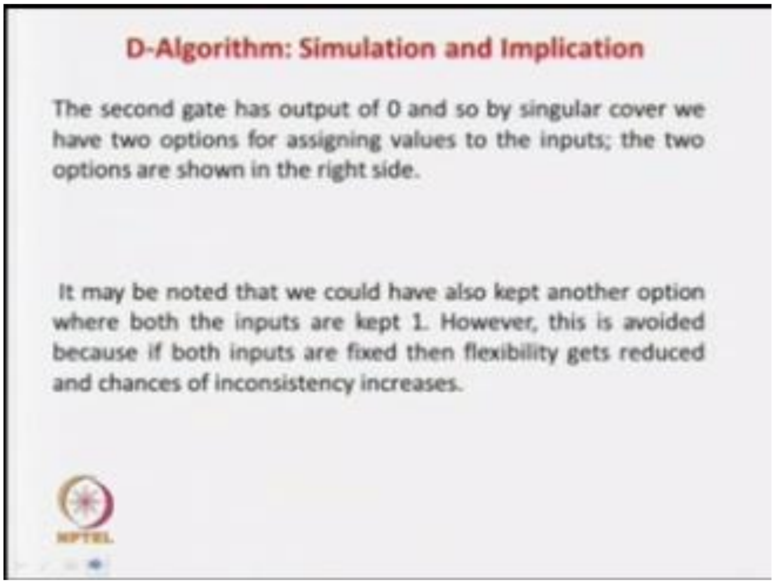
So 11 for this one to defeat this gate from the J frontier because I mean we have we got this 1 and we have justified this 1 and 1 so this gate cannot be used anymore for justification now, if you see that this and gate output is 0, so this two values of this singular cover so you can take any one of them for justification in a backward direction because we already seen that in case of J frontier that is by implication is backward traversal and forward traversal, J frontier correspond to backwards traversal so you can use any one of them and then you have to delete these list from the J frontier.

(Refer Slide Time: 40:30)



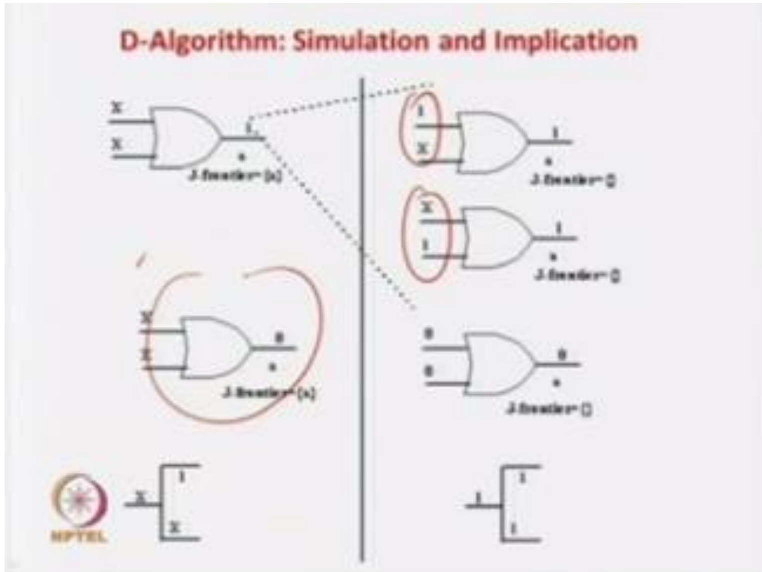
Because this already been used similar you can take for the OR gate so OR gate 1 that means there are two choices so you can take any one of them so OR gate 0 there is only one choice so after that is on the case and this is for fanout is very simple so if this is 1 so then this implications so this one both of them has to be a 11 so this from this fanout.

(Refer Slide Time: 40:52)



Is very simple.

(Refer Slide Time: 40:55)



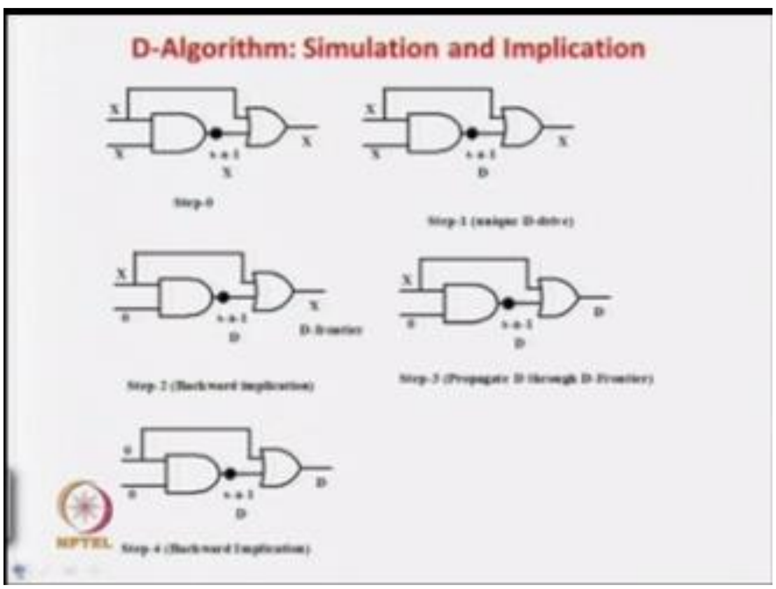
Because may this one is to be 1 then you know that actually says that one of this output is known then you have to determined this 1 so if this output is known obviously, this 1 will be 1 because this J frontier definition say the output is 1 and the input 1 input or many inputs can be x okay so in this case in this case you can see that this is the only one input because the fan output so this can be set like this and one more thing you have to know that this a, so wherever this justification has been done, so this has to be deleted.

So, this gate has to be deleted from the j frontier because this gate already has been used for justification okay so sometimes here both the value of x so it can be justify sometime it may not be possible like for some reason as I told you it may be one then what happens it says the this x this is not x this is 1 still it will be for J, frontier because here sill this co x co because to justify the values of this 0 then what we will do this x try to make it 0 but, then it is actually inconsistence below making this x 0 will not help because.

This also leads to be 0 but scope is not there and then this actually net will be coming into inconsistency so already we discussed that always J frontier I mean J frontier have a scope to actually justify the output but, always it may not be helpful but if in the case this x is 0, then this same gate still is in the J frontier with the outputs is 0 and input is the case so this input you make it to 0 and it should be solving your purpose but always a gate in a J frontier may not solve your purpose that may be inconsistency.

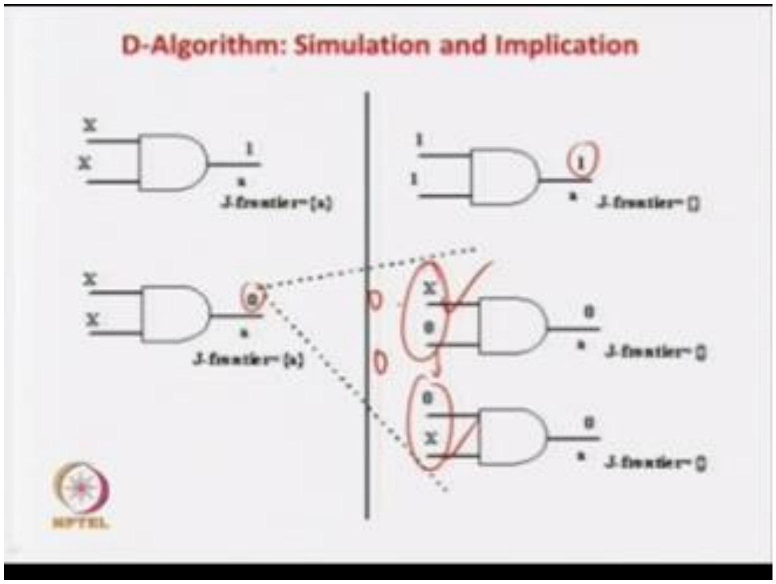
But still a gate in the J frontier means some scope that it can be used for backward propagations or something backward propagation and can be may be used for justification which may give you successful but also may lead to inconsistency okay so now so, this is what is been seeing this second gate has like this saying that this second gate okay sorry what is does say so the second gate has output of 0, so by singular cover we have two options for assigning values to the inputs, two options are shown in the right side.

(Refer Slide Time: 42:49)



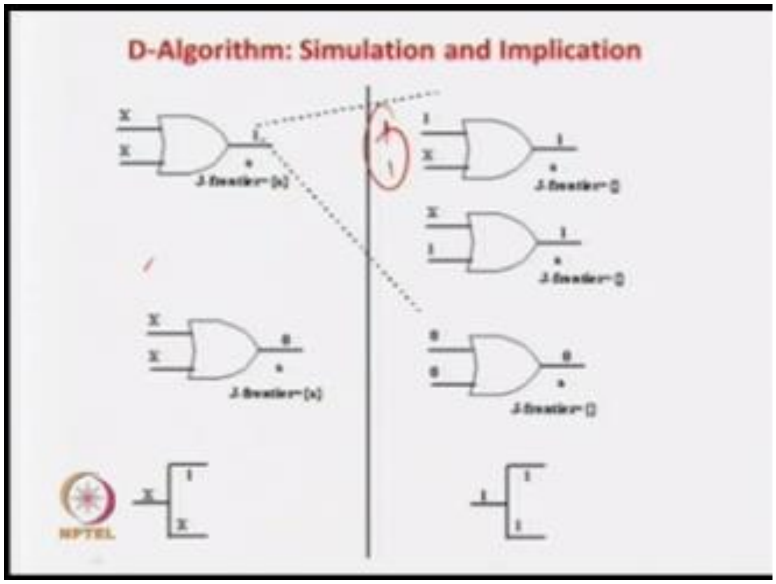
So it is sorry.

(Refer Slide Time: 42:50)



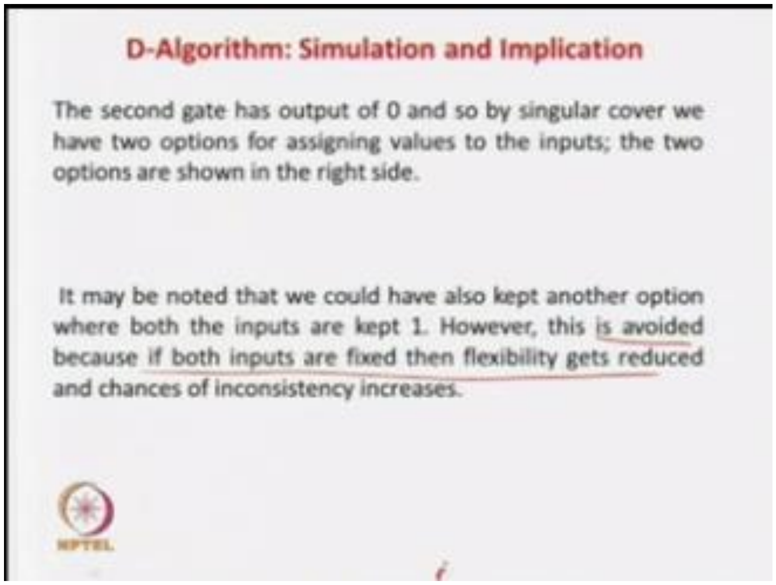
So we are talking about this gate actually this two gates, so you see that the output is one kind of a thing over here so here one is also the case here so that we are talking about this general case that is second gate over here it is second AND gate, so it is saying that the output is 0 over here so there we are having two choices so which one to use or you can also take this third choice it is 0 and 0 so why not used so this is either used this or this it is not saying that you do not use 0 0 now why in this case, so that this is one portion definite in our mind.

(Refer Slide Time: 43:17)



Similarly, in this case 11 is 1 choice which is left which you are not using so why we are not using that is very important.

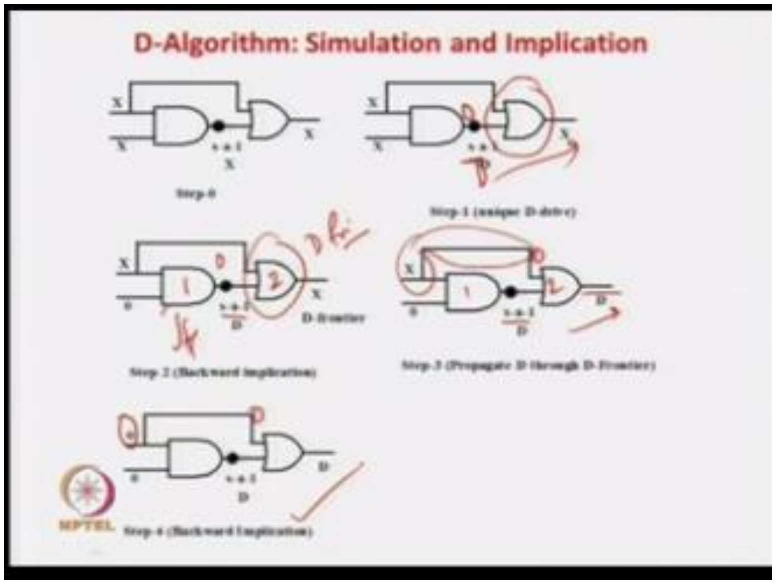
(Refer Slide Time: 43:24)



We will see o, we could have another option to keep the input has 1 but however this is avoided if both the inputs are fixed then the flexibility is lost and cases of inconsistency increase that is what they are saying that if this is 0 output of an and gate is 0 you take x or 0 x if is for an OR gate sorry if is an OR gate like this, if the output is 0 sorry output is 1 sorry output is 1 of an your OR gate so you take x 1 or 1 x the y they do not try to avoid 11 and 00 here in this case.

Now why it is said it says that if you heart chord okay if by J frontier for back propagation I mean in backward track I mean you justifying using the J frontier gates. So they say that if you put the value 00 if you put the value 11 then actually you are trying to make.

(Refer Slide Time: 44:16)



Some of the things hard chorded some of the nature you are keeping 1hard chorded and then what happens hard chord the nets to much then the flexibility is lost and the chances of inconsistency are higher so we will see that by an example so what is the example over here, so is show that this is a AND gate, so small sorry circuit so this sachet 1 so initially all the nets will be 0 this steps 0 so in step in one we have sachet 1 so just sachet 1 obviously we have to apply a 0 over here so 0 means normal case 0 fault case 1.

So it is it will be a sorry sachet 1 so normal case 0 fault case so in this case sorry it will be a your D prime so it is a D prime over here, now this is there now what is so the is only unique say D frontier so this is the only gate where d affected and it propagated so the unique d covered so it called unique d drive so obviously this is to be taken so sachet 1 now this x so this only this gate is only D frontier so you will have you have to propagate now in this case let us see so this is the propagation so now in this case if you look at so this is the only D drive so this is the only D frontier okay now this gate for this gate you.

Know the output and both the inputs are so it is d and the both the inputs were x if you look at this output is D prime we know and both the inputs are 0 sorry both the inputs are x so both the inputs are x so for x so this what and you have to get the output this is say the gate 1 say this is gate two so output of the gate 1 should be 0 okay so now there are three choices basically x0 0x so these are the two choices allowed by definition so we have also asking the question that if you take both of them 0 what is the problem.

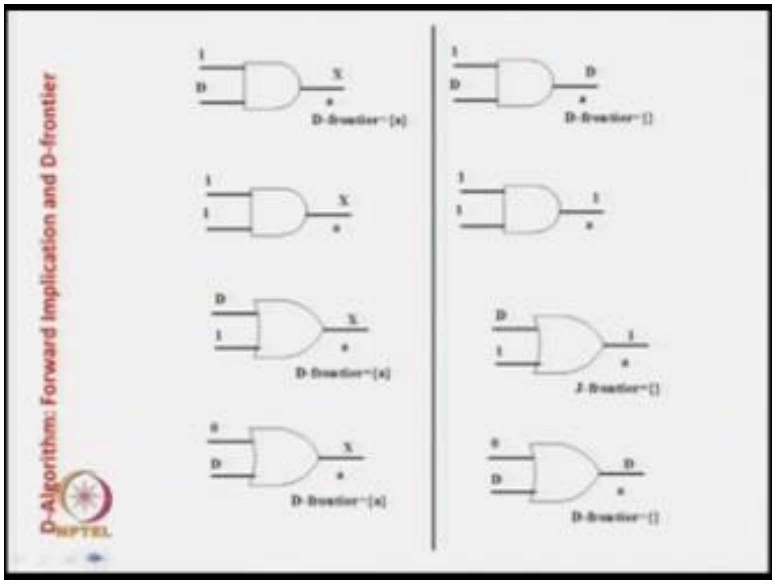
So let us go for the choice x0 will see the problem later if you use this. So, x 0 may be there, so here usually the choice x0 because the output is okay so let this be the case so x0 so let us first look at this two choices and the other choice problems and all will see later that is heart chording of this signals will see later okay so now in this case this is d prime, so x0 the choice they have taken so now if that is this is in the J frontier this gate is in J frontier.

So the output is d prime so and we require a 0 over here so this inputs are not known that is the x and x so you can take x 0 or 0x so they have taken x0 so now this is done now this is x and this is the case now this is the gate so now this is D prime okay correct so this is a d prime over here so now this affect have been propagated from this 1 so this gate 1 so gate two gate two was the j frontier.

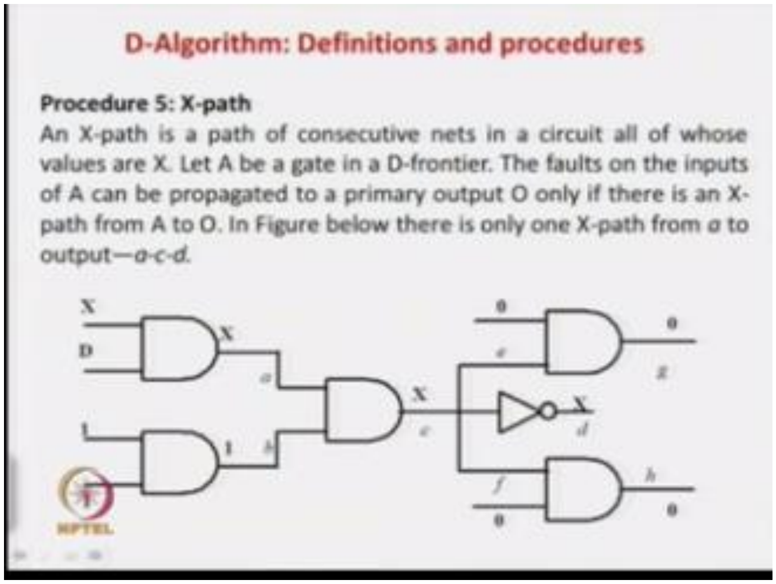
So immediately this affect d prime has been propagated to this 1so this gate is deleted from the D frontier so now we have to this net this part is now will become your j frontier because we know that that this values has to be a 0 because for an or gate this value will propagate area so this value has to be 0 so now this net this fan out net will become a J frontier so we know that by this j frontier rule of the fan nets so this will has to be a 0 this is to be a 0 so now if this in backward implication and it is done so this way it solves a problem of your implication.

That is that is already we have seen so this is simple D algorithm basic idea of the D algorithm shown here so now we see that they were not inconsistency in this case okay.

(Refer Slide Time: 47:31)

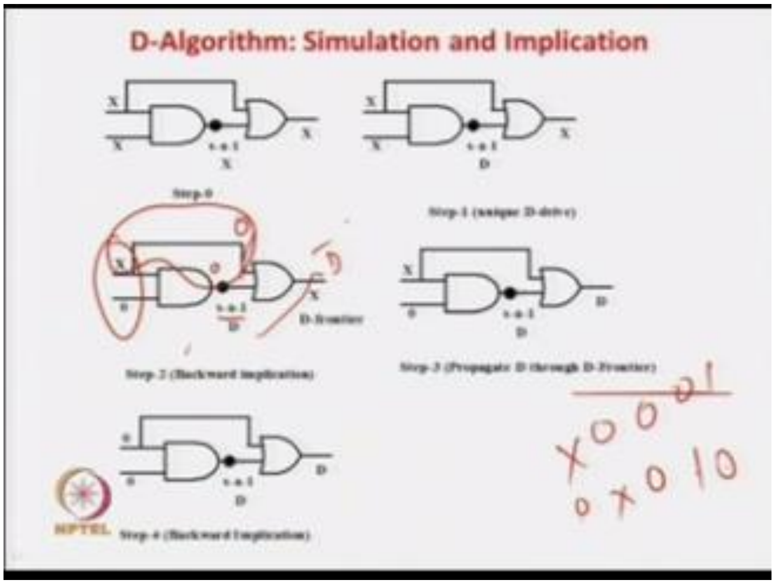


(Refer Slide Time: 47:32)



Now so this was.

(Refer Slide Time: 47:33)

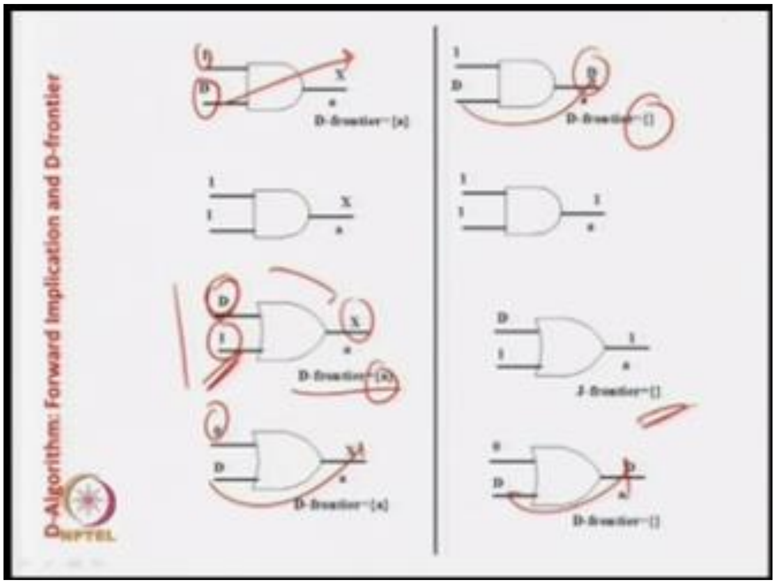


One example okay we will see later that what is the problem or what may be the problem if you try to hard chord you said of this x0 if you put this case or If you put the case 1 0 this also possible okay and right also you can have take you could have taken the case has because we require anywhere this d prime so only we require 0 over here so 0x is the case 0 0 is 1 case 1 0 you can take or also you can take 0 1 so anyone of this hard chording sorry so this is one so any one of the choice like the choice.

Is x0, 0x 00 you can also take 0 1 and 1 so any of these choices like 1, 2, 3, 4, 5 is apply over here you can get the value of 0 over here correct that is the output of this gate you can get this same that mean this is this any of the choices you apply in this case you get a 0 over here so you will get a d prime over here now this will become your D frontier so you have to propagate the values so it will become d frontier so it will propagate the value here so once this I mean D, D prime is propagated here this will no longer remains in the D frontier then you require a 0 over here and then this will become your J frontier.

And x will gets some value kind of a thing so in this example we have seen that there is no inconsistency because we have use the case of x0 or 0x whatever that was been allowed so later we see what may be the problems if you try to use the strict cases like 0 0 1 0 0 1 instead of keeping the x that is will see later.

(Refer Slide Time: 49:01)



But it was just scheme just example the elastic or the implication procedure words and also miniature version of mini of a very broad version of D algorithm okay some more definitions are actually example of the definitions or procedure we have used so say for example in this case same like it is a example of D frontier. So, I mean if it is a d and this is a 1 so this is the unique so you can drive this 1 from here so the output is x and you know the input is input is d so you can easily value the drive this value of this 1 from here.

So again the D frontier has to be is lost, so this we are propagating the value then this is the lost so other things are very straight forward this is just a value computation this another example we can also very simply says that say 0 1 and the d so the output will be say for example the output is actually in this case output is the d frontier is this 1 so either D frontier because this d is over there this output is x so you can easily propagate the value as 1 but in this case again this is very this very important point you have to note that this a d frontier like.

J frontier something always all gates in D frontier always all gates in the J frontier may not help because you have seen many examples of J frontiers in which case this gates were in J frontier but is not helping similarly this also happen in case of d frontier like in this case is the OR gate so this is the d so it is a d so the values can be propagated the output is x so the value of d can be propagated so it can be assume then another input is 1 because of some reason so this gate is in D frontier to the output is 1.

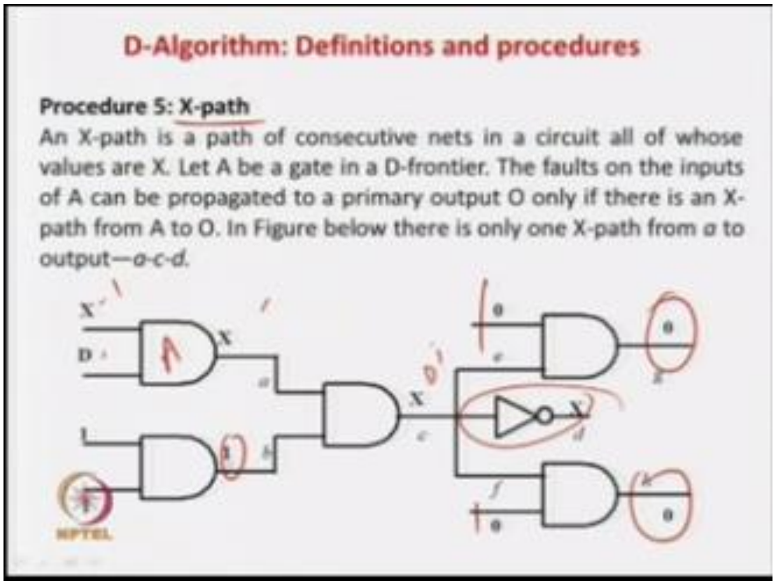
So the value of this 1 is not propagated and again this gate will be lost from the D frontier so you considered these gate to be in d frontier because by definition the output is x the input is this so there is a scope of propagating he value but because this input is 1 so nothing could have been done so as this gate was just to that this gat was in D frontier or it could not do much now why you could not do much because the value of d was not propagated because of this one.

But still we have by definition such case will be in D frontier but, when you are be trying to propagate the value the output will find already the output is not propagating so such type of gate are examples of gates in D frontier which is not of much similarly also by definition we have also seen that some cases of gates in the J frontier which we thought that some of the value affect I mean the value could be justify but, it could not be.

Because of this for example like already many times we discuss in lectures that it is we require a 0 over here sorry we require a 1 over here so this is was x and the was 0 so this was x so and one input is 0 the output required is 1 so we also considered this gate is in D frontier because these some scope that we may say to get the values 1 or in fact it is not be successful because this one input is 0 which will make the output of the gate as 1 so this was one example which a gate is in J frontier but not helping it as similar this is an example of gate in D frontier but, not helping in the propagation.

So this is request to be, so if it is 0 then the d affect will be propagated and this one will help in this 1 correct.

(Refer Slide Time: 51:54)



So I mean sometimes the lots of gates were be in so in other words the lots of gates in J frontier and D frontier but they always not be successful or always not be I mean lead to fault propagation in case of D frontier and always they are not lead to some successful justification in case of J frontier so there will be lots of gates in J and D frontiers but some may be successful some may not be successful that is what is the shown were the elastration and this example so before I mean with the help of this definition so this one more definition that is actually called the x path.

So, as we already discuss that on the inputs of your nets circuit so initial we mark it as x then by y some if you know the values of some of the nets and that is the implication we try to find the Values so for example let us have this circuit, so let us know that this some for some reason we know that let us assume that we know the values 11 so by implication this 1 will be 1 so let be this x and d, so the output will be x, so this is the case for and by some reason let us assume that we know the value of this 1 to be a 0. So, the output is 0 to be in known by implication so this value.

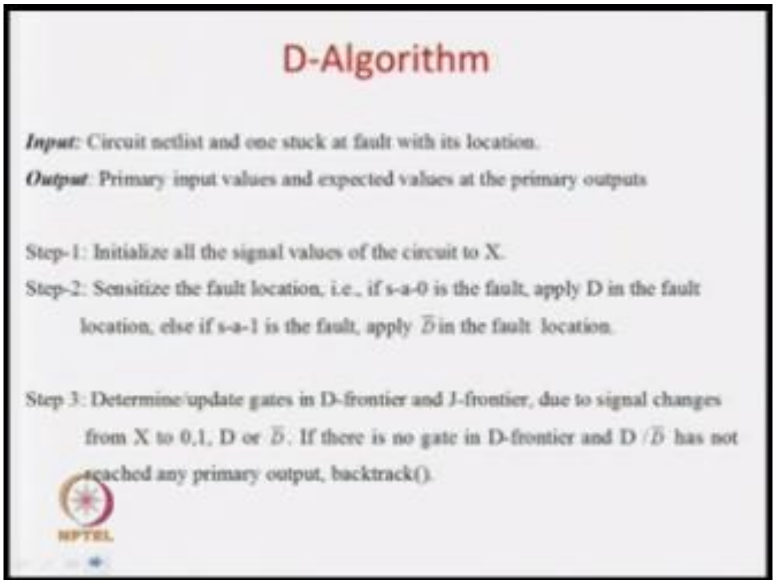
Let us assume for by some means we assume that they are not primary input may be by some either combination and logic as input or because of some other reasons we know the value to be 0 by implication also we know the value to be 0 directly now in this case you see that this only 1 value as x so in this case you can considered that his is the D frontier because this because this is output is x you can propagate you can propagate the value of this 1 can be d. So, once it is d then this gate will become J frontier then you have to get the value as x as 1 and so forth in our implication.

Procedure goes forward but for time being, so what is say that in this case if you look at it quickly so this 1 x over here, so if it is x, so then 1 more x over here and this is invertors, so this is one more x over here and we considered that these are the primary outputs so there is a path from this d frontier there is a path from here to the primary output where everybody is on x, s, his at least one path so if there is no path like this say for example have this gate not been there, so have this gate not been there then what will happen there is no path from this D frontier which is all come arrange to the x gate that is output of the gates so if it will come here some other this 1 is also1 and this 1 is also 1 so after this fault propagation will definite the here so that is in other words what we have saying that there is a D frontier or in this example we are getting a single D frontier.

We will say that from this D frontier this at least one path where all the outputs of the gates are x that means this is the only path which can be used for propagating fault definition because either this x can be D or D prime this x can be x or D prime this x can be converted into D and D prime so this for this will propagate for only to this path but this path and this path the outputs are not x so that means, if it cannot be anything and this x path is not there. So, fault propagation will be terminated at this path or this path so it says that x path is an path of consecutive nets in this circuit.

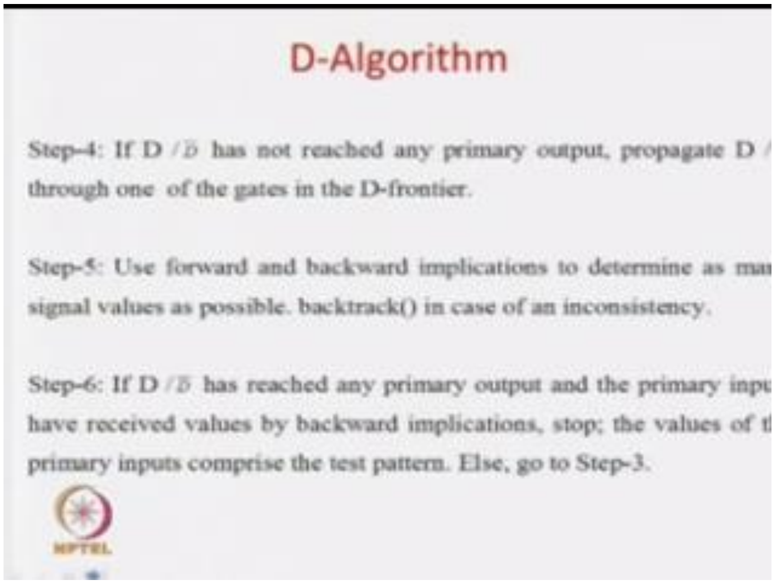
To all good values are x so let AB in D frontier because this is the gate a staying in D frontier the outputs or the inputs can be propagated to the primary output o only this is x path from a to o that is saying that this is the input and this is the output so from the gate with the D frontier that is fault effect is there so that effect can be propagate to the primary output if and only if all these path of gates were outputs all are x because this x can be converted into d or d prime and default effect can be propagated but if there is no gates no path like this, then we know that.

(Refer Slide Time: 55:19)



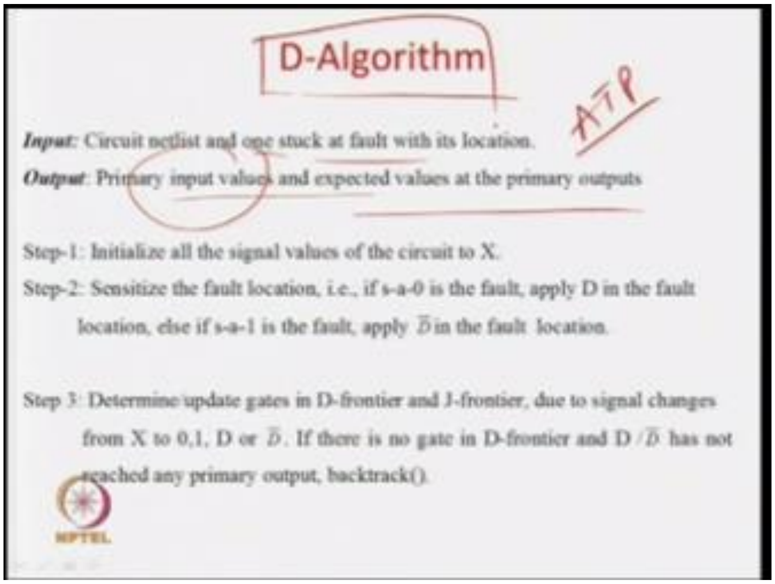
The output then the idea is that you cannot propagate the value to the output and actually in fact your lost by.

(Refer Slide Time: 55:25)



This one so that is that is all the definition we require for knowing the D algorithm and next what we will study actually the D algorithm in detail.

(Refer Slide Time: 55:45)



That is your circuit net list and primary output values and expected values at the primary outputs that is circuit net list and one stock at fault and the output. And the primary input values require test the defaults and what is the expected the primary output that is your Automatic Test Pattern you call this ATP so in next lectures using all the definitions we have studied today will try to elaborate and look formally into this D algorithm using this definition and the procedure which are discussed to be.

So using that will try to formally adequilate D algorithm and then you have to be do then in very I mean what you call in details we take an examples. And try to find out how D algorithm basically works on that algorithm using implication J frontier and D frontier propagation so today in next lecture we go in elaborate D algorithm D algorithm with example thank you.

Centre For Educational Technology

IIT Guwahati

Production

Head CET

Prof. Sunil Khijwania

CET Production Team

Bikash Jyoti Nath

CS Bhaskar Bora

Dibyajoti Lahkar

Kallal Barua

Kaushik Kr. Sarma

Queen Barman

Rekha Hazarika

CET Administrative Team

Susanta Sarma

Swapan Debnath