

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

NPTEL NPTEL INLINE CERTIFICATION COURSE An Initiative of MHRD

VLSI Design, Verification & Test

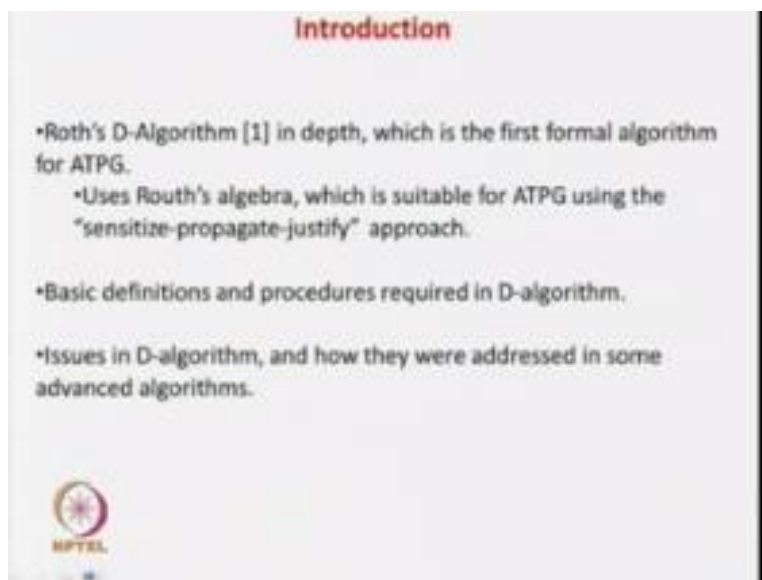
Dr. Santosh Biswas
Department of CSE
IIT Guwahati

Module IX: Combinational Circuit Test Pattern Generation

Lecture III: D-Algorithm-2

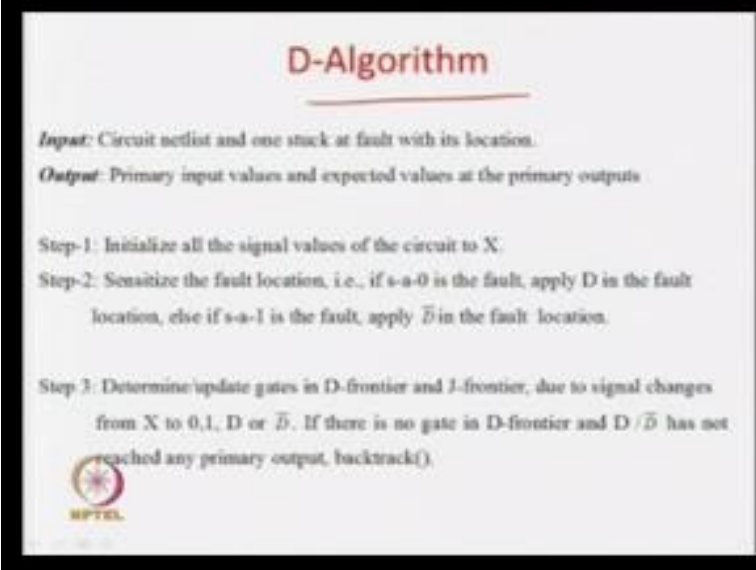
Welcome to third lecture on module 9 that is we discussed in the last class that will be looking at in details about the formula algorithm, for automatic test patterns generations by sensitize propagation and justify approach that is D algorithm. On that, we said that will be actually having two lectures covering the formula, the algorithm.

(Refer Slide Time: 00:48)



So, in the last lecture, we have discussed about the definitions and some of the procedures that would be required for this D algorithm. And then we introduce class will be the illustrating D algorithm in detail. This D algorithm we formally discuss and with the help of examples we try to illustrate this flow of the algorithm.

(Refer Slide Time: 01:05)



D-Algorithm

Input: Circuit netlist and one stuck at fault with its location.

Output: Primary input values and expected values at the primary outputs

Step-1: Initialize all the signal values of the circuit to X.

Step-2: Sensitize the fault location, i.e., if s-a-0 is the fault, apply D in the fault location, else if s-a-1 is the fault, apply \bar{D} in the fault location.

Step-3: Determine/update gates in D-frontier and J-frontier, due to signal changes from X to 0,1, D or \bar{D} . If there is no gate in D-frontier and D/ \bar{D} has not reached any primary output, backtrack().

So, what is the input in the output of the D algorithm, so the circuit net list, generally the D algorithm will be stuck at fault. You can apply any other fault models, but generally this class will be going to look about the stuck at faults okay. So, 1 stuck at fault time, you are taking because you have to remember that this D algorithm actually voids. It is the other way of test patterns generation that is the non random values, so random test pattern generations that is sensitize propagate and justify or logic based approach.

So, this is for the logic based part where actually target 1 fault at a time then we are try to find out which is the pattern that detected. So, that is we will have 1 faults this is unlike which will we detect multiple falls in 1 group. So, we will be having a circuit with a fault and then what is the output of the D algorithm D algorithm will tell which the primary input value to which will actually sensitize the faults. So, drive the faults at the output that is the fault can test this input

value can test the faults and what is the actually have to tell what is the expected value of the primary output.

If the expected value is not there in the primary output of the type reads as there is the fault. So, basically the other faults D algorithm formally tells you what is the pattern to be applied. As a primary input to it take the faults that is sensitize propagate and justify the faults. Also, it will tell you what the expected values of the primary outputs are at that time if it matches the primary output expected. Then, the faults is not their fault being that is being considered if the pattern does not match the expected response, you can say that this fault is then it has been detected okay.

Now, we will see what are the steps actually do go about this D algorithm, so first thing is that all signal values of the circuit are assigned to x. So, this is the very first step and that we take generally D algorithm, so I already in last lecture you already seen that many ways of actually dealing with sensitize propagate justify the base approach, what is that we sensitize the fault propagate the output. Then, we justify that that is not very formal approach or way of dealing with this, so this D algorithm will actually do the same thing. The more formal way procedures like implications x drawing and all those things like some basic definitions like singular cover, so using those things we will be covering formally.

So, in the first step you assign all the circuit values, all the signal values to x that means what in this start, we wrote know what the signal values of the each net of the circuit. So, next step is sensitize the fault that these are the points you already know that if there is stuck at 0 fault, then you can apply the signal value 1 and keep this stuck t 1. We have to apply the signal 0, so if there is stuck at 0 fault, then you have to apply over 1.

(Refer Slide Time: 04:17)

D-Algorithm

Input: Circuit netlist and one stuck at fault with its location.

Output: Primary input values and expected values at the primary outputs.

Step-1: Initialize all the signal values of the circuit to X.

Step-2: Sensitize the fault location, i.e., if s-a-0 is the fault, apply D in the fault location, else if s-a-1 is the fault, apply \bar{D} in the fault location.

Step 3: Determine/update gates in D-frontier and J-frontier, due to signal changes from X to 0,1, D or \bar{D} . If there is no gate in D-frontier and D / \bar{D} has not reached any primary output, backtrack().

NPTEL

That means in the normally case normal case is the 1 and so that the stuck at fault will be 0 implying that it is D in the fault location else stuck at 1 to the apply as 0. Then, if the normal case is there, if the circuit is there then the stuck at 1 fault to be considered as 1 and the answer is D prime, therefore sensitize the fault. Though, it is stuck at 0 fault, you write d in the fault location, if it is stuck at 1 fault, you write d prime in the fault location implying that this is the fault affect which is to be propagated.

Then, in step 3, then once it is done where the first step is actually make everything x, then you sensitize the fault. So, that is you write the D or D prime in the fault location and then you actually determine the gates in the j frontier changes in some changes, some changes what happens because you actually putting a D or D prime in the fault location.

So, putting D prime or D in the fault location now you have to propagate the values, say for example, this is a gate be this is the fault location you make it D. So, when there is a lot of combinations, so there may be two parts, then again at the third part, so by this fault affect can be propagated. So, just whenever you apply this D, so automatically this gate this gate and this gate become a D frontier because the fault can be propagated this gate this gate or this gate.

So, similarly, once this you know that this fault has been propagated to as the output, then the other values of the input has been need to applied. So, fault affect can be propagated to different here that means you have to detect this, so what happens in other way other way studying in the other way. If the moment you apply change x D or D prime sensitively fault, immediately the gates which have corrected to the fault location become the default.

Here, when this you have to select any 1 of these gates from which the fault, so you say that the fault is in which form the gate is propagated. That you seen from 1 gate, now what value has to be given to input of those different errors, so that the faults will be will now actually be falling at the j frontier into them, then what we have to do? Signal change will appear, then what we have to do, then we have to propagate the fault affects through j frontier D frontier and justify through the j frontier.

We have to keep on doing it till you reach the primary output here fault affect is the primary output, but it may happen that there is no gate in the D frontier D and D prime are not reach the primary output. That means it can always happens like for example, I tell you so let us take I will give detailed example, but just to highlight the point say for example, this is an AND gate and this is stuck at 0 faults over here. So, you apply once D over here because this is stuck, so this is the very detailed example will be covering, but it is really very simple.

(Refer Slide Time: 06:43)

D-Algorithm

Input: Circuit netlist and one stuck at fault with its location.

Output: Primary input values and expected values at the primary outputs

Step-1: Initialize all the signal values of the circuit to X.

Step-2: Sensitize the fault location, i.e., if s-a-0 is the fault, apply D in the fault location, else if s-a-1 is the fault, apply \bar{D} in the fault location.

Step 3: Determine/update gates in D-frontier and J-frontier, due to signal changes from X to 0,1, D or \bar{D} . If there is no gate in D-frontier and D/\bar{D} has not reached any primary output, backtrack().

Handwritten notes: A red circle around 'D' in Step 3, and a handwritten 'D' with 's=0' next to it in the bottom right corner.

So, this is stuck at 0 faults over here, so you apply 1 you will get a D prime so this is the stuck at 0 fault, you apply over here 1 you get the answer for sum results, this gate is having input value of 0. So, anyway this is the only path because no other fan out from this, so this is the only path fault affect has been propagated this, by this gate only.

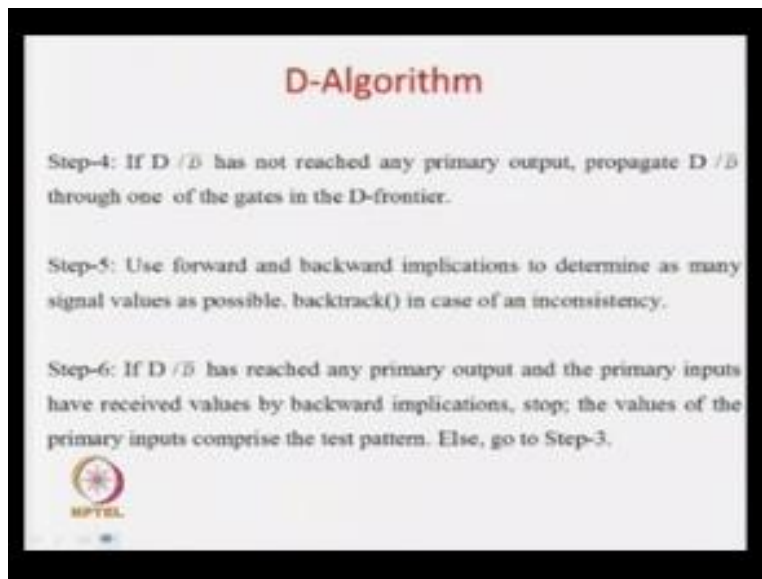
So, this is the only gate in the D frontier, but now define that to justify this you should get 1 over here otherwise fault affect is here. So, you can say that actually what do you mean by gate to the D frontier this will have actually not fault as the D frontier because when this gate is already this fault is already 0 to the output is already 0.

There is no other path for this fault affect will be propagated, so this input is 0 because the output is always going to 0, so this net has no different here you can say in that way. So, in this case also assume that it is not primary output and this also not the primary output this also not the primary output. So, the fault affect is you get a clash and you have to backtrack back track means again you have to try out.

So If they know I mean this fall may be un testable or if you have an another option of driving the part another gate from the root and then you can try for the other option those things will be detailed in the example. Essentially says that you take a fault you affect, I mean that we apply D or D frontier for location maintain D frontiers and you have to keep on doing with its fault affect D frontier reaches the primary output, but this is some point of time no gain in the D frontier.

That is somewhere you D or propagation has been blocked and if you not having primary output and then you have to backtrack. So, back track means in a lead to the case that the fault is not testable because there is no other options left for driving the fault to the output or it may also happen in other way. That is some other propagation the other gates in the due to the fault again in the propagation another gates in the D frontier at some previous step, so you can try for that.

(Refer Slide Time: 08:43)



So, you know that is what it is being said d or D frontier may not in time propagate the 1 of the gates in the D frontier. So, when this step is saying if there is no D frontier then backtrack update the D or j, and then it may some gates at least in the D frontier. Then, what do you do if you not do as the primary output propagate D or D frontier to 1 of the gates in the D frontier that is you get d or D frontier.

If there is the case that there is no gates in the D frontier algorithm has to backtrack and if there is small gates in the D frontier. Then, you have to select any 1 of them any 1 of the gate then you have to propagate the value of D or D prime to do this until you reach what you call as the primary output.

Now, once you have jumped 1 gate that is propagated the value of D frontier 1 gate to the value of j frontier, you have to use forward and backward implications to determine as many signal values are possible. That is you go for implications, you have to go for implications or for implications, or you have to justify the gates in your j frontier. Then, what you have to do you have to at any point any kind of inconsistency that is you require 0 or 1.

Actually, the signal is other way and it is 1 and 0 in respective cases that is I mean you want 1 already signal is 0 and other way round. Then, you have to say then it is not possible to justify the gates in the j frontier and then you have to back track in some other option which is left. So, whenever you have moved the that is what actually say for example, this is 1 gate and this is say D and then you propagate the value of D gate and through this C says that there is an OR gate over then.

(Refer Slide Time: 10:35)

D-Algorithm

Step-4: If D/\bar{D} has not reached any primary output, propagate D/\bar{D} through one of the gates in the D-frontier.

Step-5: Use forward and backward implications to determine as many signal values as possible; backtrack() in case of an inconsistency.

Step-6: If D/\bar{D} has reached any primary output and the primary inputs have received values by backward implications, stop; the values of the primary inputs comprise the test pattern. Else, go to Step-3.

NPTL

So, you know that fault affect to be propagated so the D prime it is D, so this is the and this get affect output of this gate, so you know that this value of 1. So, immediately you have to put x 1 or 1 x to the OR gate, so I mean in the last class discussing why we want to put x 1 and not 1 1 or the other way round.

That implies we try x 1 or 1 x, why not y 1 1 because 1 and 1 in the OR gate or x 1 as the 1 x in the OR gate which we use 1 it is satisfy your case always use x 1 or 1 x. But not 1, 1 you will see what is the problem for outputting the signal values like 1 1 that is you are not put it 1 1 using the concept of singular cover, but why what is the importance of singular cover.

We are discussing we have not given an example which will see in the questions and answers session for the first time you take this read that 1 x used. So, you will get the value of 1, how do you get it actually mean this is D, so initially all this signal values is x this is also x, so you know that it is x point.

So, the value of this fault has to be propagated to this is the gate which is in the D frontier, so now we will get the value of D over here. So, now once you get D so you can think that it is actually becomes j frontier kind of a thing backward implications you can say this value has to be 1 at the fault propagations so immediately if this is 1, this gate follow in the j frontier and the backward implications hither x1 and 1x to be there.

So, whenever when you know that I put x 1 or 1 x you get the value of 1 here and automatically propagate x 1 by this gate. So, that is what we use forward implications, to determines as many signal values are possible. So, when you are when propagating the fault value D prime for 1 value D frontier, 1 level immediately do that backward implications and by justifying the values of the j frontier gates. Otherwise, and then try to solve as many exercises as possible, but it also happens that there can be inconsistency to require 1 but you are getting a 0 or you have to apply 0 kind of thing so in that case of clashes you have to again trackback.

If any other option on the gate in the D frontier rather option of trying it x as possible, but it is also possible that there can be inconsistent. If you require 1 different kind of thing because you

see x_1 and n_1 to signal values which are applicable for justifying the j frontier. So, if you find that x_1 is fine, but 1_x is not fine, then I mean say you get 1_x , then you find that 1_x is not fine, I mean satisfied.

Then, you have to try x_1 kind of a thing, so who will determine all these things see in details by backtrack means that any step if there is option available if there is n number of gates in the j frontier D frontier, you take any 1 of them. So, in this case if you are taken any 1 of them, then you can try with the other way similarly, you justifying a gate j frontier 1 case for this last example we have discussed x_1 and 1_x . So, you try 1_x if you find that 1_x is not solving your purpose you can use x_1 , so there are some of the options.

The backtrack always means that you have try out for options and in this step 6 if d prime is any primary output, then primary inputs have received values by backward implications. Then, stop the values of the primary inputs comprise the test pattern else go to repeat keep on going. So, these two steps propagate the value of the D prime 1 step and immediately justify immediately what do you called implications by implications of the inputs of the gate in the j frontier, you saw as has been possible and you keep 1 doing it D or D prime primary output that is your fault affect has been D primary output.

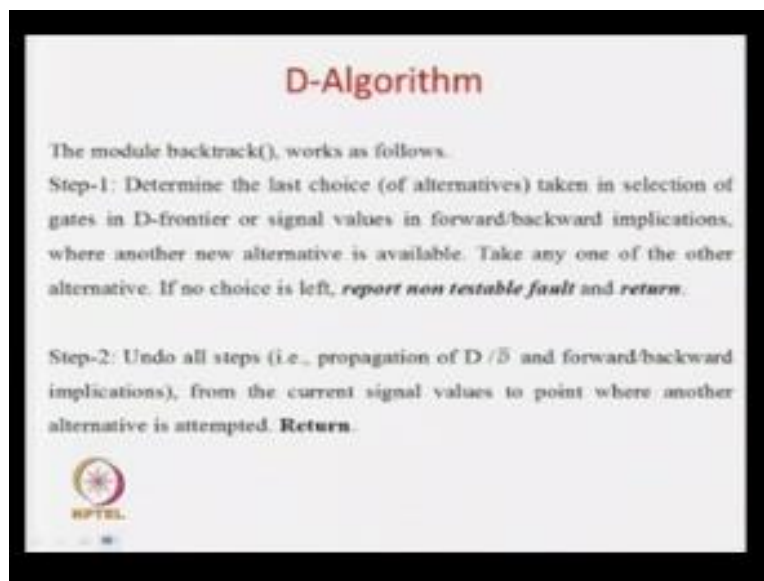
Then, your job is done, so immediately, you find out that x of the input and if x 's are still there is not in ensure the primary input because the stage at that it either 0 or 1 . You need not bother about it, but when your D or D prime primary output then you should be very happy that your fault affect has being the primary output.

Whatever the signal values are the primary inputs that becomes your test pattern and whatever is the fault affect has been the output that says that the expected output for the primary output for the test pattern is so and so. So, for example, if you primary output get D and the input pattern and the input values at the $1, 1, 1, 1$, so it will implies that the input pattern is $1, 1, 1, 1$ for that fault and what it says that your output is D it says that it is in the $1, 1, 1, 1$.

Then, in the normal case the answer will be 1 and in the fault case the answer will be 0, so that is what is implied by the automatic test pattern generation by D algorithm. So, once your job is done once your fault affected is D or D prime, so what is the effect of fault, Then, there should not be any x in and also you have to observe that you have to take care that any x in the any of the intermediate.

So, if there is any x in the intermediate, then you have to justify them using the backward implications of the j frontier gates. But if there is no such x in the intermediate gates in the primary output if there is only some x's then you need not bothered because x has been input in either it can be 0 or primary output that will not hamper in the fault structure of that primary input, but still if your D or D prime has got this is the primary output then you have to go back.

(Refer Slide Time: 15:25)



To, step number 3 D prime taking the value of D prime then what you call add some 1 step to the one gate to the another gate. So all these things will now details will in example but as I told you that they some backtrack what was backtrack means this slide actually tells about the what is the backtrack. So, what was mean by backtrack, so backtrack means what say for example, there is a clash that is you wanted the signal value to be 1 by backward implications of the j frontier but

gets some that means 1, 1 by other case and so forth. So in this case you have to backtrack, and so and you have to try other alternatives so what is the determining the last choice of alternatives you are taking selection of the gates in the j frontier or signal values backward and forward implications of the j frontier.

Whether new alternatives also available, so what are they say for example, there were lot of choice in the gates in the D frontier. So, you took 1 any of them, so you can try with any other outputs similarly, there can be number of signal values like for example, if it is there OR gate so you get here 1. So, it can be $x \ 1$ or $1 \ x$, so in the n gate answer to be 0, if you want, so it can be $x \ 0$ or $0 \ x$, so there are two choices available.

So, these are saying that signals values in forward and backward implications in the gates of j frontier, so there is some alternatives if you tried with this 1 now you will try with this 1 or if you are tried with this one. So, try out and do say for examples these are inconsistency then you undo all these steps so what do you mean by undo all these things means retry wherever you have given some converted some signal values x to some values D , D trying 1, 0. So, you make all the signal values x and these two are placed like for example, you started with the for example, it is the end gate.

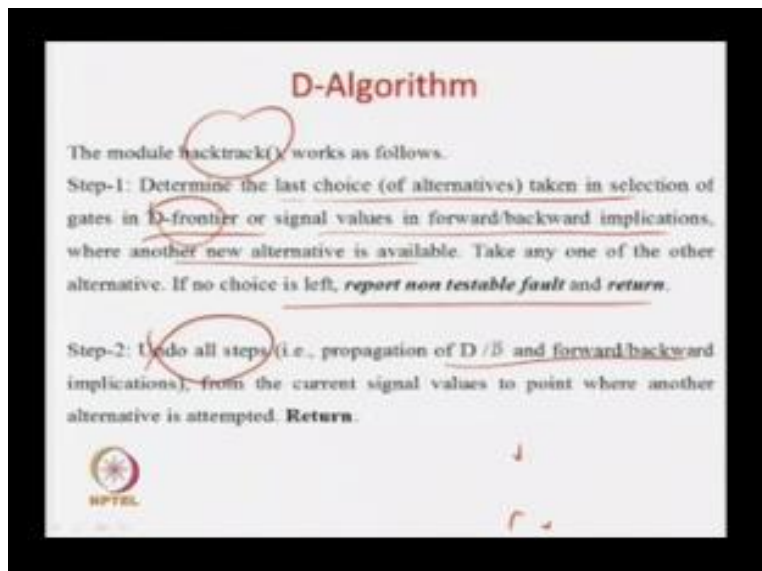
So, it was stuck at 0, we applied our we got D and there is see three gates c gates, so this is 1 D frontier this is 1 D frontier this is the 1 gate is D frontier selected this. Then, you found a inconsistency, then you backtrack up to this, so what do you mean by backtrack to this point, that means all the signal values which we say that propagate the value of D here say some propagations here. And that is we converted some signals with values x to 0 1 to x from x to 0 1 D or D prime so you have to undo all things.

So, everywhere you put that value of x and this case whether option was there, then you said that this path this things, so you start working with this path. So, similarly doing that backtracks, undoing means you have to convert all the signal values for 0 to 1 D or D frontier till you reach the point option are available. Similarly, for example, if there is an AND gate you require 0 over

here use x 0, then you find out a failure, you keep on that tracking and change all the signals values 0 1 D or D prime to the x.

Then you find out there is a problem over here, so you know change the option to x 0 and again try out. So, these actually called, actually this is called few alternatives what is actually called you clarify the effect and this one. Okay now it happens say in some point of time, you find that there is no choice available then you can say that fault is untested or then the faults cannot be tested because of the redundancy.

(Refer Slide Time: 18:41)

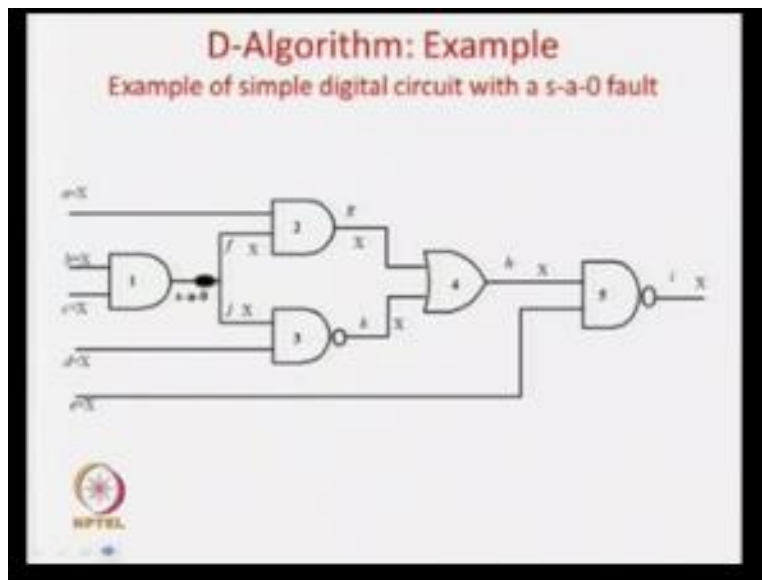


So, for that also 1 example we will see, then will be question answers session so what is the in doing on the steps what we discussed in step number two undo all steps means actually propagation of D prime and backward the current signal values. I mean in last step attempted that is what discussing actually undo all the steps means.

So, what is backtrack says that backtrack says that you undo all the steps till you reach alternatives and the options. So, that means what we means in a very broader sense this step says

that you call out the inconsistency come back to the point, where you get a there more than 1 choice and from the point, where you backtrack points.

(Refer Slide Time: 19:27)



These are the options take all the signal values undo all the steps undo all the steps means you change the values of signals x from 0 1 or D or D prime that is you are trying in or other alternatives. If there is no other alternatives available, then what you have to do is then you have to declare that nothing was possible that is there is no other choice available and what you say that is the fault is un testable and so on. So, when what is the idea behind this one the idea behind this one is that try out all the choices.

That is why this D algorithm is called a very expensive algorithm because you might have seen in the last few day module like sensible propagate justify in this entire step, but in today's example you will see that it is not the in case many times, it will have inconsistency. Then, you have to try out many other alternatives as circuits have generally circuits have lot of fan outs.

So, it takes a normal fan outs, so on the fault locations you have all the options in the D frontier. And so, if you fail in the first option then you try for other option and keep on trying you

successfully propagate the fault affect to the output. That means what is the killing point of D algorithm, so had it been in the case that your D algorithm already always successfully, you can select out the best options.

Generally, case is there that faults are un testable means generally there means redundancy in the circuit if few faults will be there which will be un testable, but then why do we call like D algorithm is a complex algorithm. This is because any other time we cannot hit the correct option at the first time.

For example, there can be among the D frontier and j frontiers there can be only very few choices through which the fault affect can be propagated to the output propagate. So, why it happen when you fail when you count option two then you again and you can understand that this is actually a in browser complexities because see in the first level there will be two gates in D frontier.

Second level there is three gates in D frontier, and then there are two gates in the D frontier so for. So, you have to try out this is two options and from then option 3 from this, then there can be again two options. So, it grows like NRI tree, so if you are try out all the options exhausted, then you have to try out the n r a tree at each level. There are n different options available in D frontier kind of thing, so it actually grows up exponentially, so that is what the difficulty of D frontier. So, if you could have made out the intelligent D algorithm actually we will see in the backtracking.

If there is some intelligent search in the tree then D algorithm will not be that complex, but in general sense what happens you keep on trying out the options one after another. That is actually the killing point of this D algorithm say it is become very complex because you do not have very guided direction of search, but now in the advance algorithm the guided directions of searches are slowly coming in to picture.

So, there can be advance discussion in this course, but generally this idea is because of the advances in the D algorithm and henceforth to find out good search bases I mean among the big

search base of the any other kind of a thing. We have discussed we find out good tracts say for example, here we have failed in one path and you use the information that have failed in the path and use that information to select a better alternative for the j frontier faults in the next.

So, that is how it goes on, so if we had been in the case that D algorithm, sorry D algorithm is always very successfully sensible propagate exhaustive happens in just one goal propagate exhaust. Then, D algorithm will have very would have been very similar in complete way in random test patterns. That is because concurring that way, but will have been in very good alternatives compared to random test patterns.

It is for the easy to test part, but this is not the case, but generally for many of the options you will find that you have taken a wrong decision and again backtrack and keep on doing this. That is what only for difficult to test use for D algorithm or its variants and for the introduce test part we go for random test patterns.

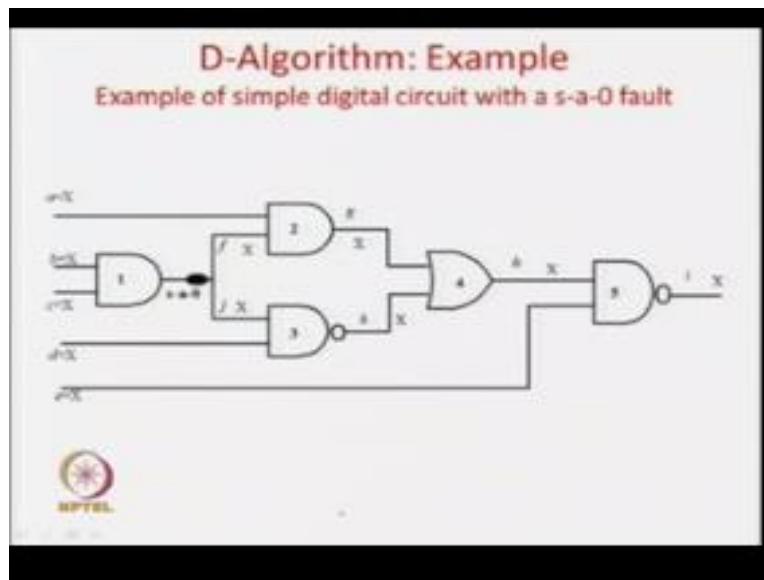
So, I mean that the D algorithm for the life is not very green here because there are lot options many of the options you made bound to take wrong options and again you have to back track. That is the number of faults we cannot be tested as very high, but in that cases you have to try many options in the phase, but that is not the killing point the killing point for a successful gate is 99.9% which testable in a circuit for all the successful cases.

It is still there, lot of options available and it is the duty of the algorithm D algorithm is very fair algorithm because this is the first algorithm of kind in a D frontier. So, it does not say about how which way or which gate which way you have select the option, but almost algorithms too tell you guided direction of searching the order that is if you talk about D algorithm for that we does not give any options you randomly keep on explanatory state phase which is actually take making it very complex okay.

So, that is what is the brief discussion that why we I have been most prone to go for random test pattern generations as much as possible and apply D algorithm only for difficult to test faults. So,

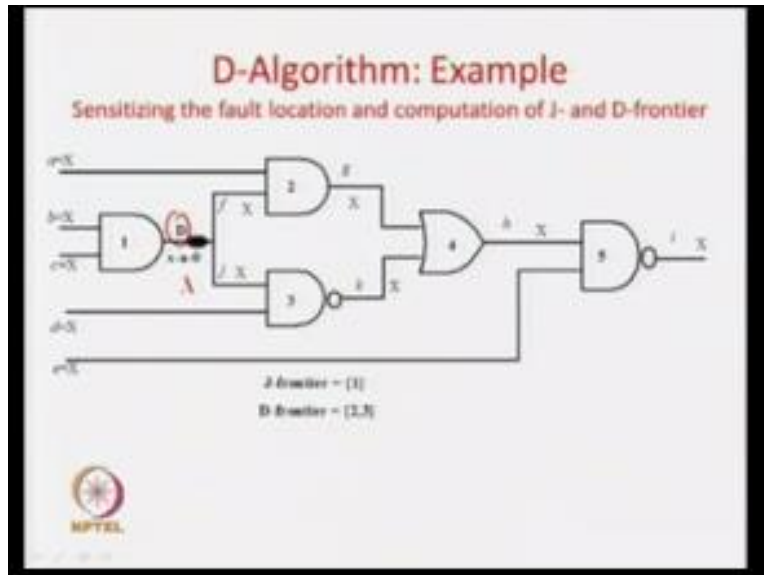
now because we many times we go for lot of backtracks, now here we will see an exhaustive algorithm, I am sorry exhaustive example which will tell you how exactly D algorithm works.

(Refer Slide Time: 24:01)



You also see many cases of backtracks for here, so then it can give you very good highlight of the D algorithm of the lot of options possible. It may not be green as the successful, sometimes it will fail if you take many other options, but we will see in the many formal lotions as if you have discussed the kind of D algorithm in the last few slides said that these are the circuits you van observed that all the led's having xxxx in all initials to xxxx..

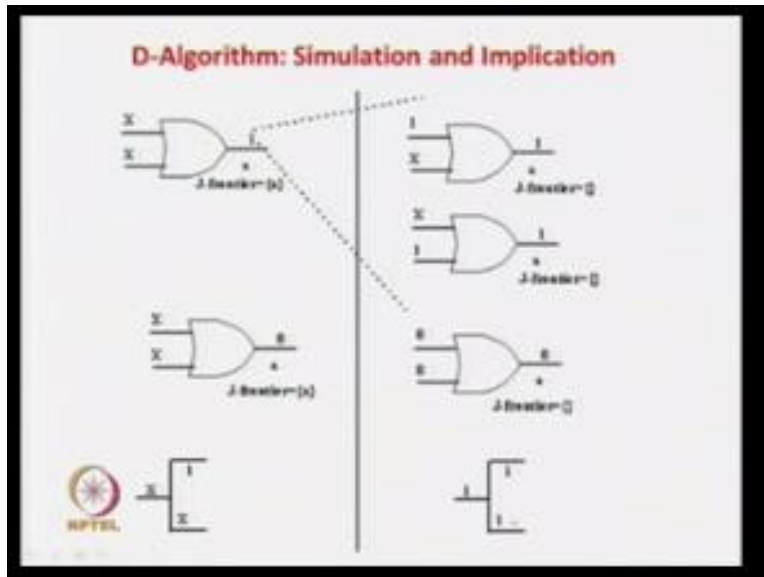
(Refer Slide Time: 24:51)



So, there is all the places has been initialized to x, so now this is the stuck at 0 fault so we want to find out the test pattern like sensitize propagate justify approach so this is first step. So Now, next what is the case now stuck at 0, so you have to apply 1 over here, so that is the very straight forward way to know. So, normal case for fault 1 you will get the value of D over here, now once you applied d by definition of D frontier, what you know that there should be a gate that's from where you actually can pass out this fault.

So, you can observe that this is the fan out also initially; you can say that these two are actually append this. So, this will two options can use for propagating the fault affects, so this is the gate whose 1 input that will be d this x can be eliminated through D this can out, so because you already know, let us go back to slides.

(Refer Slide Time: 25:26)



So, you can see this D algorithm, so you can see that this is for the fan out this is for existing. You can see the just the definition, now just we have to quickly verify this D frontier the same kind of this thing. So, you can see that the inputs are D and so what is the D frontier comprises. So, the output is x and at least 1 of this input D or D of the definition at D frontier comprises of a gate the input is D or D and the output is x.

So, you can see this 1 like the definition of D frontier that the output is x and the 1 of its inputs is j. So, obviously these two gates will be part of this D frontier, so you write that two and three are in D frontier kind of thing. Now, you see j frontier, so you know that this is D, so these value has to be 1 over here.

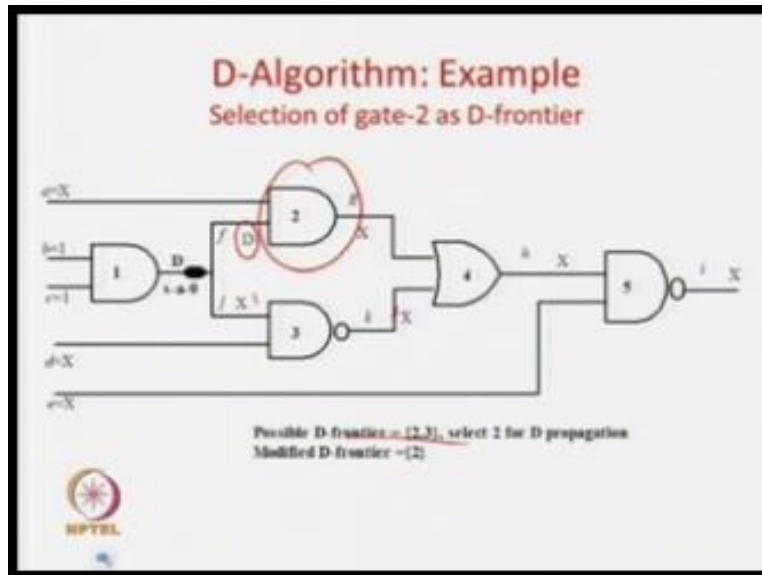
So, what is the definition of j frontier j frontier says that a gate where output is 0 1 that is known, so this output is D that means you have to know that you have to apply 1, but if the inputs are x. So, this gate it has to be 1, so by backward implications if you justify how can get a 1. So, this get number 1 will be a part of D frontier, sorry j frontier j frontier as we already discuss that it mainly handle that track and D frontier go for the forward direction and j frontier tells you about backward direction.

Now, what happens, so this get output is 1 we know to sensitize the for the our inputs are x so this one will actually be part of j frontier and fault affect is D inputs of both the gates are D and output is x. So, this gate will actually used which will part of D frontier, now our choice is here is the choice that you remember this is the point of choice point of choice. This is a point of choice over here, you can take either this gate or this gate for this fault propagations.

So, these are point of choice you always remember the choices of point know what happens, as first as I told you that every step then actually what you have done. So, if you just propagated the say for example, to sensitize this fault, so you have sensitize the fault by propagating the fault affect to a D frontier, so automatically it gets a j frontier so once you have done this. So, D frontier has been certificated or j frontier has been certificated X.

So you have to slowly J frontier here D frontier so j frontier is required one, so this X,X obviously one another one okay because that is why singular drive so singular cover problems So, if you have 1 over here the inputs the n gate should be 1, 1, , so you apply 1.1 so we will solve your j frontier problem that means I mean resource of the j frontier while the D frontier is empty because the faults because the problem of j frontier by backward implications. Now, this 2 or 3, they actually it should be d or d, so you should remember that these two gates are in D frontier you have take any one choice of this, now you just think it.

(Refer Slide Time: 28:30)



Which choice is there, so you just written the 2 and 3 are the possible D frontier, here this thing. So, you have to select anyone, so for example this guy because I told you d is a flat algorithm. So, it will not if you use any kind of estimations to tell either 2 is better or 3 is better, but the advanced algorithm I just input Other algorithms which are advanced version, they will tell You they can give you a guideline you take 2 or 3 which one you have to use for the propagations, but as this is not the case over here.

So, you will not directly go for that, so will, Give for the traditional D algorithm as they have dealing with that, so they will take two arbitrators take two for our fault propagation okay, So, 3 we are not considering so this mean of possibility D1 also there, but cannot possible the fault propagations into this. Suppose, the time being you can keep it as x it is not an issue, so we are going to affect the fault propagations over here okay.

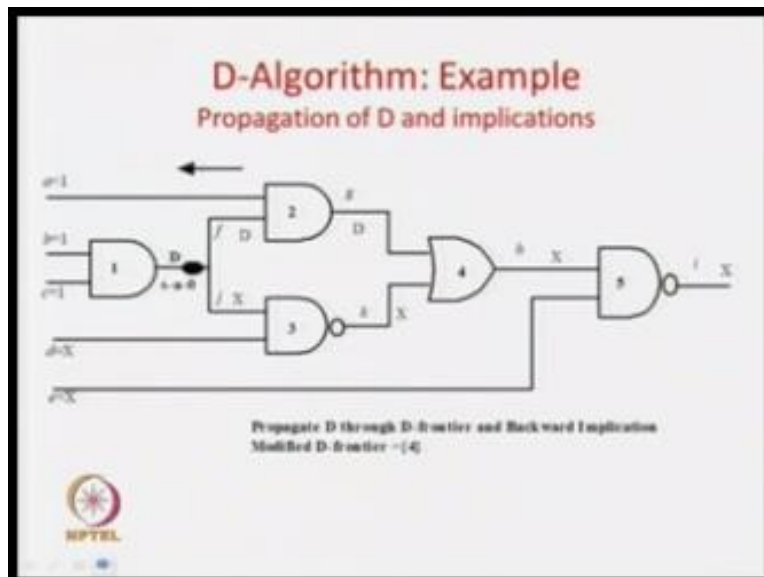
So, there was two choices because this d was here and D was there, now you are going to propagate the faults affects to this, so will keep back it to x, so we will see the modulation why are keep trying to more number of lines as x. If there is a chance that will see with an example in

the courses of the selection, so you could have written d over here for the time being it will not write this you can maintain this.

For the time being, you are not propagating the faults affects through their only propagate through this so d over here okay, so, the modifier D frontier is only x and three we have digital because Dilute d would have been nearby definition c would also have been in the D frontier. One of the inputs is d and the output is not known, so it is a possibility of I mean D frontier says that one of the input is d or d prime or the output that means the chance of propagating the fault affect to this gate, so if I write at d over here.

So 3 should also be the part of the D frontier here, but the designer or the test engineer has decided that you propagate the fault only from this one force of the time b we do not make it as d. But in fact it will be d only because the fan out fault as fan out step, okay so this is what we have been done so it is been selected you write a d over here and this is the case.

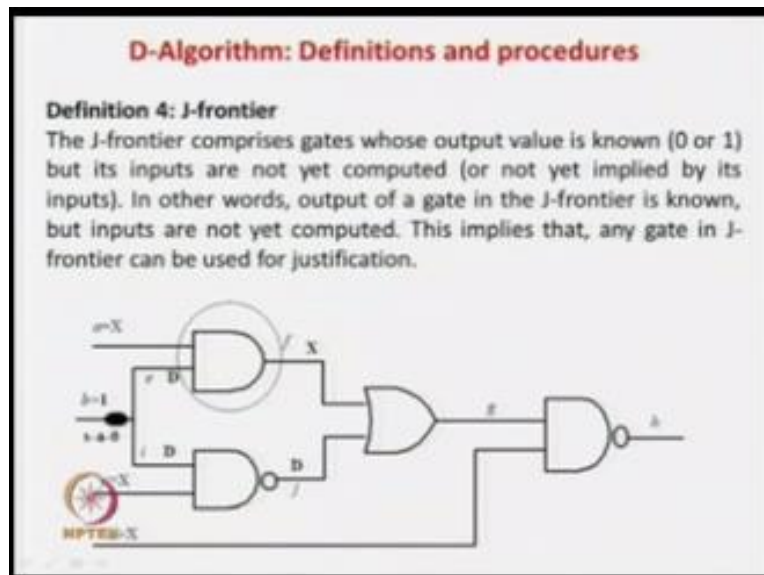
(Refer Slide Time: 30:26)



So now, so this was your D frontier area so you will be up to propagate the value of d, so you know that d is over here, so and 1 of the input in this case was x, so it actually propagate the

values of d over here. So, this gate is x, so you can think that this is a type of a j frontier kind of a thing, so in this case acquired propagations you have 1 over here. So, I mean t this point I have to mention that strictly by the definition of j frontier you will look very strict definition of j frontier, it says that there are some discussions over this. So, j frontier actually tells you that the output of the gate is 0 or 1 it says that what j frontier is saying.

(Refer Slide Time: 31:03)



The output of the gate is known to be 0 and 1 any inputs are not yet known, so it says that for example, some gate is there output is actually for in this case say for example, you know that this input is d and so in other words, it is a gate say for example, the output you know is to be 1 or 0 and for both the inputs are x actually what is saying again the output value is known any inputs are not yet computed. That is you do not know the value of input, so in the last class also we have seen some examples kind of a thing like some fault affect maybe this for example, in this case say the fault affects may be some d over here.


And you do not know it, so then this example we already discussed in the last class that very simply that this is the case. So, you know that d will be propagated from here to here as we have discussed if the input is not known, so you can consider this as the j frontier.

(Refer Slide Time: 32:16)

D-Algorithm: Simulation and Implication

The second gate has output of 0 and so by singular cover we have two options for assigning values to the inputs; the two options are shown in the right side.

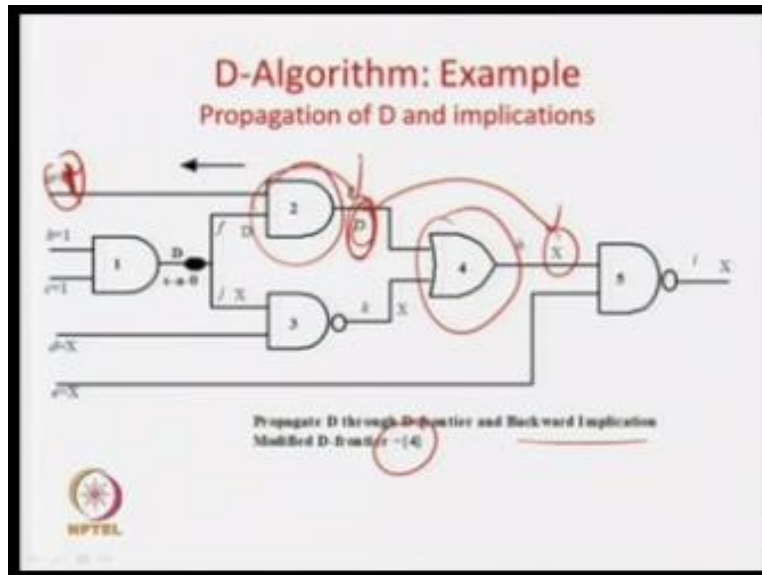
It may be noted that we could have also kept another option where both the inputs are kept 1. However, this is avoided because if both inputs are fixed then flexibility gets reduced and chances of inconsistency increases.



So, there is sometimes we do not call this things like that, so in this example so what we have seen that 1 of the input of the gate are d, sorry one of the input of the gate was x and the other input is d and the output you have also considered to be at d. Now, sometimes we may call that it can be very easily justified by a backward propagations, that backward propagations 1 input is d and the output is d AND gate so that the input has to be considered to a one. But sometimes it may also happens that the other way round that is both inputs are x so very strictly speaking sometime people say that the both the inputs are x I mean if the 3 input AND gate the 3 inputs are x it maybe that all the inputs are x and the output is known.

So, very strictly say that is the only gate in the j frontier and the other gate if the 1 is known, and the other two inputs are x then, you can say that solve this using what do you called this backward implications, but strictly there is they may not call it as the D frontier.

(Refer Slide Time: 33:08)



So, the next step which was saying that so you have considered sorry, we have we are saying that this gate this person has selected the gate number two as the propagations for d. So, you have propagated this point now you can see that the output of this gate d is known and 1 input of this x so you can think that this is a kind of a j frontier and by backward implications this is already known to be d.

So, you can say that is equals to 1 so that is why the j frontier backward propagations you have done it. So, it is saying that knows you can see that propagate this D frontier and backward implications that is the 2 in the j frontier and immediately you have solved it to the 1. So, very strictly you may not call it as the J frontier kind of a thing because just one input you can easily use that backward implications.

But in the definitions and terms you can I mean in the formal terms you can discuss the algorithm. You can say that as this input is unknown the j frontier can easily make it 1 and solve the problem backward and justify this input. So, justify this input, so that this fault d is propagated the idea was at 2 is left at D frontier and fault affect is as the output now, input d is

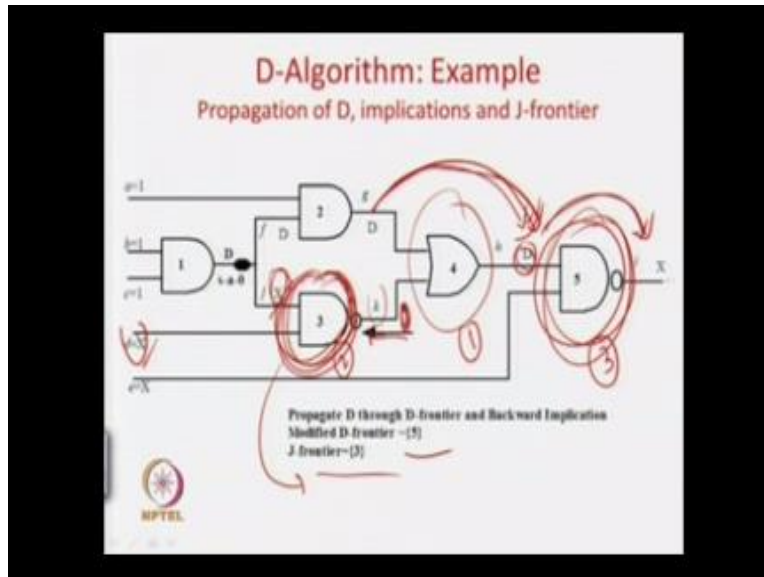
known this 1 input was x , so by backward propagations of this gate two you may get a 1 and everything is justified.

So, we have already discussed that backward implication is the j frontier and forward implications is the D frontier kind of a thing. So, now you can see for this gate what happens d is here that is only 1 part and where you can propagate the so at this all the times you have to observe that there is 1 x path, from this path to the output this is the part which is having all values of x over here. Already, we have discussing that in first lecture that for the fault effect to be propagated from the D frontier or from the gate of j frontier they should be at least one part, all the needs are having x , there is this path will be used for fault propagations.

So if there is some gates if the value of x is not there. So, it is 0 or 1 some it happens like for example, some reasons we can say that is this gate has become a 0 so this affect will be a 1. So, there is no x path from this point to this point, so only this fault affect propagation will be stopped at this gate and the fault cannot be detected. So, what we have discussed in the last lecture that this fault has this always has to be this fault affect is d from this gate in the D frontier there should be at least a path which is an x path.

So in this case there is an x path available, so the faults can be testable maybe testable this point, you can say that, okay. So now this one is gate four in which 1 of the input is d and the output is the x . So, you think that this fault has been may be propagated through this so modified this D frontiers gate number four. So, when we call a gate in when we call a gate in a D frontier we call a gate in D frontier if and only if 1 of the inputs is d or d prime and the output is x that means that the scope of propagating the faults values to the output. So, it happens in the case of gate number 4, so easily we may say that gate number four is in the D frontier and this two is deleted from the D frontier because once the fault propagation has been done that we have seen in the last lecture you have to delete it.

(Refer Slide Time: 36:10)



So 2 is deleted and 4 will come in the D frontier, now this is your d, now what happens, so you know that it is to be a d this is a d. So, what are the values for the D frontier to be propagated, so the backward implications you can think that the value will have to be a kind of a 0 because this is only when the value is 0 then only the fault effects can be propagated.

So, you can think that whenever d is over here, so gate d is in the D frontier, but just after you propagate the value to here. Then, it is no longer immense a gate in the D frontier to delete this, okay and then output is known but the input one of the inputs also you know obviously, the other input is not known so you can get it by backward implications, and then that is very easy to say that you can very easily you can handle the case and you can say that apply a 0.

So, you can think that there is backward justifying backward for gate number four in the j frontier properties, now this has been done. You can see that now question of gate number 3 and gate number 5 has been propagated, so has risen, so what you have seen that whenever there is a propagation of this d or d prime some new D frontier and j frontiers arise. So, in this case what happens you propagated the values from here to here, so immediately 4 will be deleted from the D

frontier, but it gets a problem for this, then you have to find out this signal value here and then, also it creates a D frontier.

Because it gets the j frontier because you have to find out what is the values of this net number k by backward and also, it will be it will create another j frontier for 3 because you have to find out the value of this two nets, okay as well as it also created a new frontier also gate number 5 because there is a chance of propagating the values of d from here to here. The output is x, so this, so this propagation actually creating three things, one gate number four you can call to be a part of j frontier kind of a stuck because this signal value output will be known, but one of the input is not known that has to be done by backward implications.

Now, once this is done, once you solve this problem that is you put a value of 0 that means next moment another gate arises that is the output of gate number 3 is known to be 0, but the inputs are x and x for which we have to know the values and how do you know the values of this one, correct and then another gate, this is say you can say this is number 2 also one more thing is created that is number 3, so that is what you get the value of d over here, so there is only 1 path there is a gate number 5 is the output and the input is d.

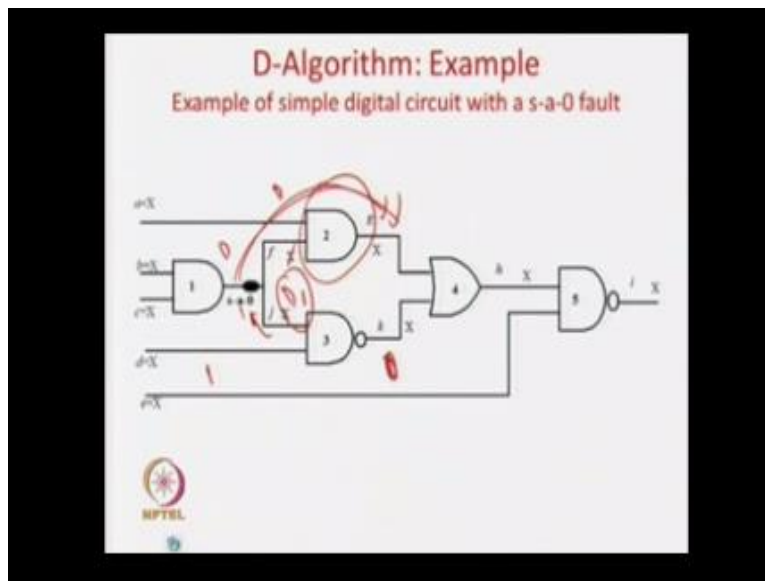
So, this gate can be tended detailed in kind or propagation of the affect, so it will become another gate in the j frontier D frontier, sorry so modified D frontier is 5, modified j frontier is 3. So sometimes in definitions we will see sometimes we do not very exactly called gate number 4 as j frontier, why because 1 of the inputs is d the other only 1 more input is unknown that is the we just say that it will a 0 by backward propagation, but 1 of the input is known, so only one more input you have to justify, so we may not call it as a D frontier in a very strict term, but even if you do it as I do not see any harden in that.

Because j frontier is responsible for backward implications and D frontier is responsible for propagations. But now, for the gate number 3, you can see the both the inputs are x and the output is known to be y. So, by very strict definition of j frontier the d will be highly in the highly be very strictly, it can fall in the j frontier definition because input output is known and

both of the inputs are unknown, but to my understanding you can also call 4 to be as j frontier as d has been propagated to be there and it has been deleted from the D frontier list.

You can immediately call as because you have to justify this, but now it is has been fall sometimes some definitions are strict sometimes you follow them strictly and sometimes there we can also with such slide use of notation you get call them both to give a very coherent feeling for both the definitions. So for we know so once you have decided this signal to 0 backward propagations. So, this gate number 3 will be in part of j frontier and gate number 5 will be in part of the D frontier. This will it was in the propagation of the faults affects of 8 and this for gate number there you have to get a 0 at the output, for that you have to find out the values of signal x at j and signal x at d.

(Refer Slide Time: 40:29)



So, now let us see next what happens so how you get a 0 value to the output of the end gate, so it can be only by 1 and 1. So, by this D frontier value propagation you have to get 1 and 1 over here so that is what is the case so modify the D frontier is 0. So, you have to apply 1 over here 1 over here and that is being told over this case now what you see immediately you know what happens now you get a 1 over here. Similarly, if you get a 1 over here, where the output is

known and what do you call say if this already I have discussed the output is known. So, we require also 1 over here that is because of backtrack j frontier of the over here. We know that this is a d, now actually here clash over here.

So backward implications we require one at the output or this gate that is, so there is a inconsistency and you have to back track okay. So, mean also we have been discussed that, sometimes we would have also said that very in the first slide of the lecture first slide of the example. We said that this was a x and this one was also x, very first slide I mean just if you look at this example we may discussing that why we are putting a x kind of a thing this is example, so inconsistency at this point because we required one. We required 1; sorry we required a 0 over here, kind of a thing for that you require 1 over here and 1 over here. So, that was this I mean part of inconsistency.

So but if the first we said that as we are going to propagate the value of d over here. So, let us make this d over here and keep this x because we are not converting it in to a d over here, but we could have also said that this is the d we could have also said that this is the d because of this fan out propagations fan out d will it will be d. So, even if you are propagating the value of what do you called fault affect to this D frontier gate number 2, but still we could have said that this input number net number j is d. But we have not done it. We have said that keeping it x and we are giving a scope up lying up 1 and 1 over here.

So that I mean we are trying to establish a 0 over here. So, for that we require 1 and 1 over here, so we are trying to make this afford over here and then you are finally, ending with the clash over here. Now, you can ask for the question that in the very beginning if I would have a put and if I would have a put a d over here, then what would happen in the case. Then, I would have say that try for this, this gate number three we need a 0 over here, so you can have 1 over here and also require a 1 over here. Then, there could have been that the clash would have stopped here I could not have I many not have moved up to this step you could have same 1 step in the computation that is d over here.

But I require 1 over here to make a 0. So, clash on stop, but in this example, but exactly following this algorithm we have because we discussing now. So, what we have done here is that we allowed it to keep it x we allowed x over here in this point and then what happened is that we have to resolve this clash at stuck at point okay. So, here you can see the one more step having a problem that is this two way of saying in the approach you have taken. So, we are keeping as many x as possible, and then you have seeing that you have lost one step or require one more step in the computation.

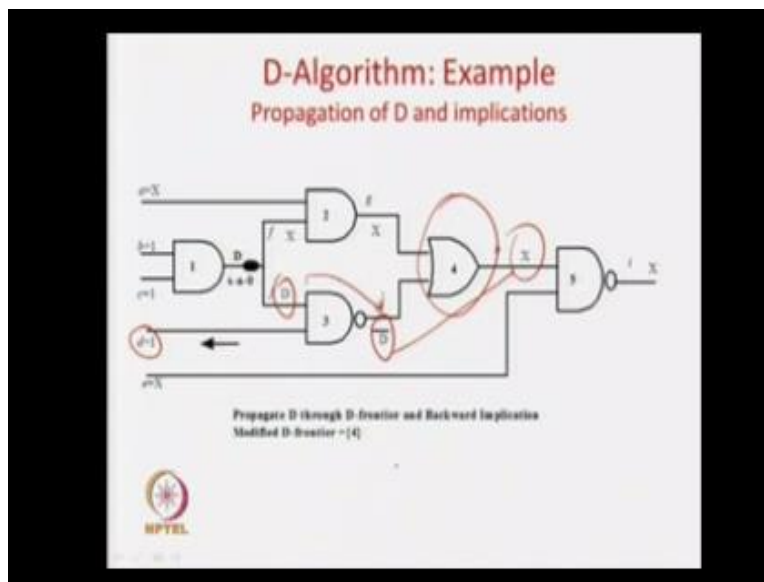
But if you have taken the other way round that if you have applied a d over here in the time because initially it would have applied d in place of j, then or backtrack or inconsistency would have been for net number j itself. But now, but it has been called as the output of input number for over serving, but actually our input having example we have to keep this example is not suitable to show that. In the end, we will see the example where will see that if you keep as many x as possible there is a more probability that your inconsistencies are less.

But if you hard court more signals. If you have you could have a hard court d over here, but if the hard court is 1 at some place, but if you do like this then what can be the problem is that many times we land into more number of inputs consistencies. Then, the case that when you are not having the hard courts signals here that is in this case our conflict occurred at the output of gate number 1, but if you hard court the signals or conflict could have occurred in the gate number j. So, that is in this case this is the good thing our computation we are able to catch the affect of the faults much before end, but in this case it is have boom.

But generally, by putting more number of x trying to different d conflict has possible, somehow if you could have differed the conflict to the primary input and beyond, then could have been very successful that is the philosophy is having more number of x delay your conflicts. So, if you can somehow delay your conflicts to a reasonable amount of time you can pass it through the primary inputs then your problem is solved. So, that will be a very clearly give in an example in the end, but in this example you could have argued that I should have been accepted that it is the case.

Then, if you could d and in case of j then conflict would have in j, but as you are not doing it by strictly following the algorithm or conflicts that at the output of gate number 1. So, one more step delay, but will show with the example, but this delay can be tolerable, but generally by putting a more number of x's, the circuit is more flexible and more prone to delay getting conflict. Sometimes, we can we will avoid the conflicts by putting more number of conflicts, so that will see in an example in this thing. Then, we will go to this next step of this.

(Refer Slide Time: 46:07)



So that is just we have seen, so there is having inconsistency. Now, instead of digitally inconsistency over here digitally inconsistency over this thing, but it has come out of cost of putting x over here, but in the example we will see that x also as an advantage refereeing of this one. So now, let us coming look too much at this advantage or disadvantages of putting more number of x because it is a secondary element in the D algorithm, but the primary algorithm is primary job of this algorithm is to find out the solution. So, we had a inconsistency at this point that is at this point there was a inconsistency at this end.

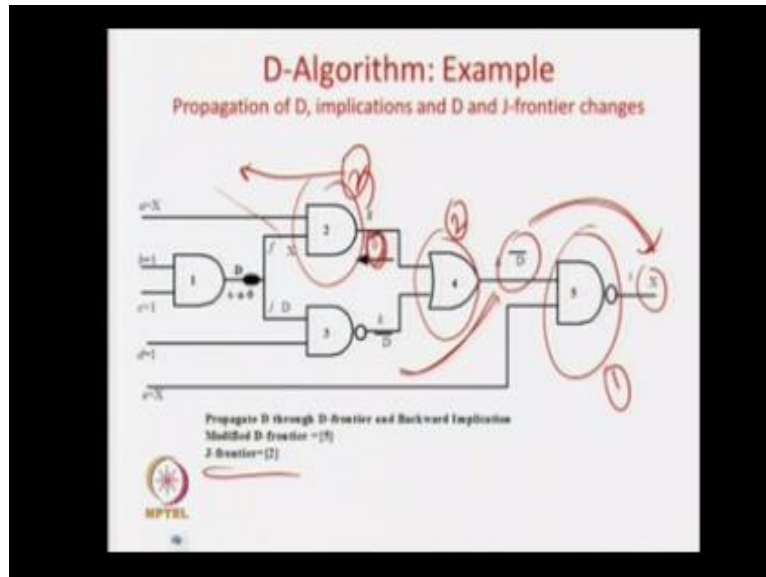
So, to have got x over here in this place, you could have put a d over here inconsistency with the here, but the inconsistency here. So, but anyway such inconsistency point you can think of this

way so let us 1 step add this 1 inconsistency called at this point, what we have we have a choice of we have a choice of t in the here. So, then we discuss whenever we have this choice we have to take all d in the choice in this case we have taken, we have to take as there was choice would have been no our choice. You should have a declare that this fault is unstable, but now we have a choice of this point between gate number 2 and gate number 3 what we will have to do with that. We have to take another choice that is gate number 3 because the choice of 2 and the 3 that is the D frontier.

So now, we have among these 2 and 3 frontier, we have taking the 3 frontier in that, so as I told you this is d, so this is should d, but to keep the circuit more flexible or in kind of some way of mean delaying this clashes as much as possible. We are keeping it x because you are considering the gate in the D frontier and you have also observe that when you are doing backtrack. All these points are again made x that is what is taken that you already discuss that whenever there is a kind of what you have to do that. Whenever there is backtrack, you have taking another option, you have making all the gates will be x all these x's has been done.

So, now this was in the D frontier gate number three now you would a D prime over here, so once you put a D prime it will obviously fault affect will propagated. So, this will be D prime inverting gate, so now you can see that this gate number four will be in the D frontier because this output is x and the input is fault value actually the scope of propagating the value. So, this becomes a gate in the d front here gate number four and here j by j frontier, sorry j frontier procedure thing that this input output is D prime input is D. So, this value has to be a 1 justify this one, so D front by j frontier that the gate number three these have to put D over here, this is the modified step next.

(Refer Slide Time: 48:50)



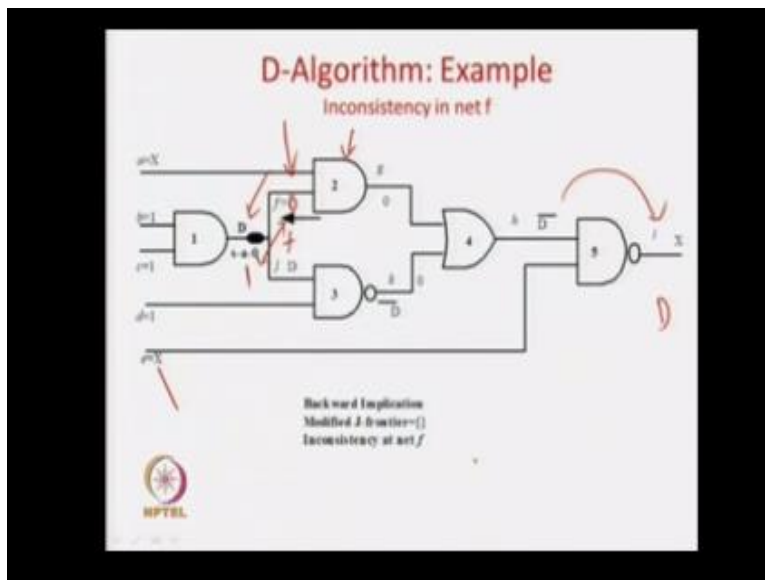
Now what happens? So, you get a d primer over here the output is x, so obviously this one would become part of your D frontier because this propagation is what you can call this probability that it is this fault has propagated through. So, this one is part of your new D frontier correct, so one more thing is the case that is we are propagating this value to the output here. So this value. So, by this two things are here, so one number is that gate number five is becoming your gate in the D frontier as well as the propagations. So, you know that this concept j frontier or gate number two this value has to be 0, why this value has to be 0 over here because the output of the gate is OR gate.

You will have propagate the value has to be 0, so that this is very well known fact. Now, also you can see gate number three what happens to gate number two the output is 0 and both the inputs are x and x. So, again or this gate number two will become a part of j frontier because you will have to go for the backtrack design correct and the modified j frontier is number two which we have done. So, what we have done actually seems again, so actually be propagated to over here. So, first thing is that get them five will become a part of a D frontier okay.

Next steps this probability of propagating the D prime to this next this value what is the input value of gate number four. So, that this D frontier fault affects will D prime then it propagate to gate number four that is a 0 has to be applied over here. So, that you can also be solve that backtrack, sorry the backward implications to now the j frontier of gate number four. So, you can say that it is point number two that is being done as well as whenever this value of two 0 has been done over here that backward implications.

So, immediately you know the output is 0, 0 and the inputs are x in gate number two will become a part of a j frontier. So, these three things, sorry one, two and this is the third thing these three things happen. So, this next step in the algorithm for this propagation right now,

(Refer Slide Time: 50:49)

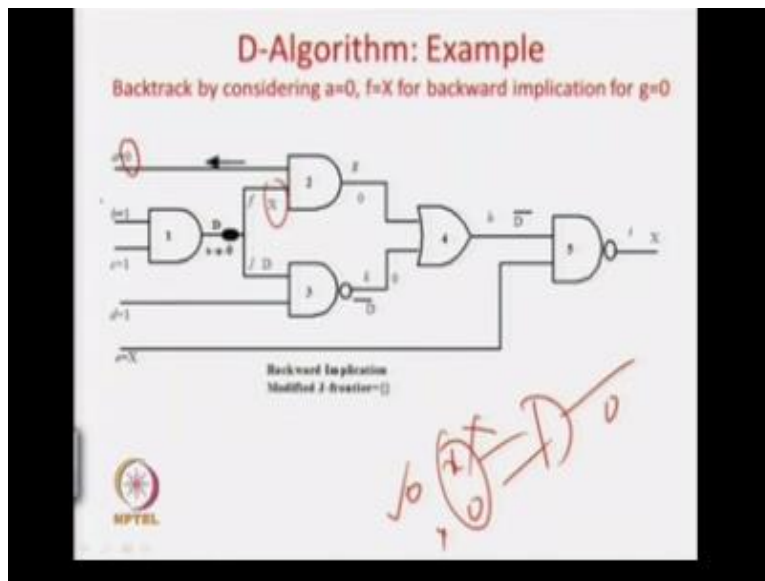


Again as I already told discuss that to put of this end gate is what know you have a D over here through this steps are here. So, you know that here concentrating about this one, so you know that the output of the end gate is 0 and so what can be the all the possible cases here the case we know that singular cover, it can be x 0 or 0 x. So, you already know that now, let us try the case of, sorry, sorry, sorry, we require a 0 over here. So, we require a 0 over here, so in this case you could have put a, sorry this is a mistake on my personal type. So, actually we require a 0 over

here, so could have say that $x = 0$ or this option has been taken, but there to be another option that is $0 = x$ that is you could have also taken 0 over here.

And could have kept end at another option here. So, due to some reason see that this person has not taken this option he has taken just previous option that is x and 0 over here. Now, this is the case now you can take this the picture and now this, so by modified j frontier, so two you have stated to be 0 and x by singular cover. So, this becomes 0 and now you can see that I would have this one. So, you will get D over here over this and this also this 1 more step you could have done it, but you see there is another inconsistency is that started place will be just immediately detecting this because you require a 1 over here. By this backward, what do you called backward implications or the j frontier in gate number two it was saying that I require a 0 over here.

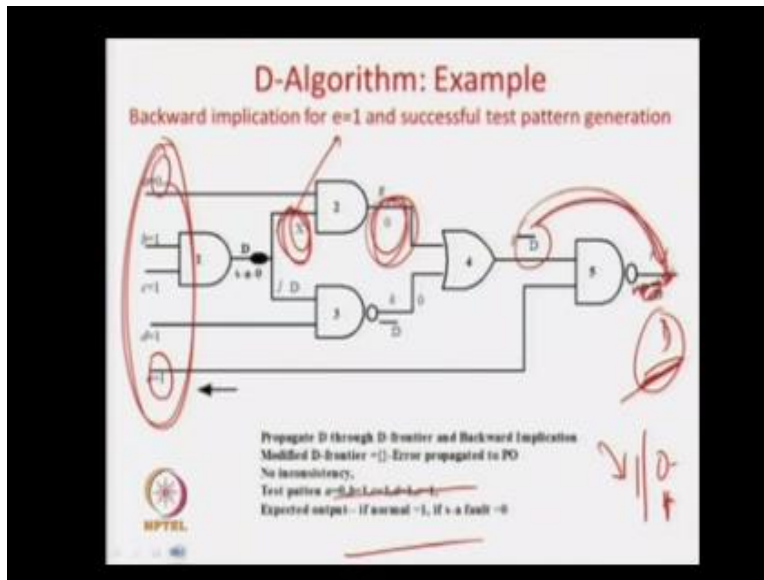
(Refer Slide Time: 52:40)



So here again inconsistency or stop over here. So, now what you can do, now you just seeing another way of doing it. So, how do you do that, so one more option was there that is see that there is one more option that was in the end gate the two option of getting a 0 over here. So, one is $x = 0$ and y that is 0 , so the last slide we tried with this and we found that there was an inconsistency. So, let us try with another option again another option over available, so what here

does here. Now, you put a 0 over here and you put a x over here there is this option, now you again j frontier will be two will be there in the j frontier and once you do this if you deleted.

(Refer Slide Time: 53:10)



So now, let us see what and hence you require a 0 over here, you require a 1 over here and all these and this already that propagate, you should get a 1 over here that is fine. Now, you get a x over here, now what is the case this is fine, so what does this x has been mean that anything that is the output 0 here is not at all depending whatever activity over this line. So, that is why we are keeping the x and not a D, so you could also put a D here you could also put a d over here and you could have done it.

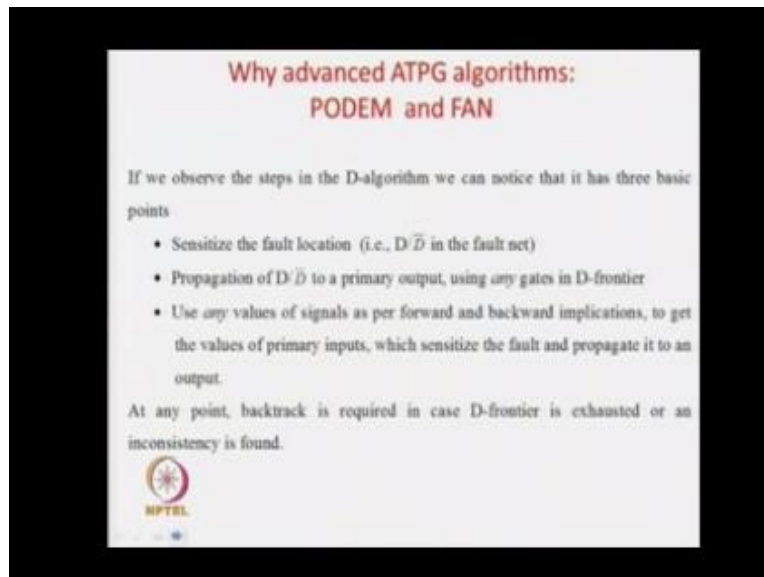
Now, we have not done this because we are concentrating c as the D frontier, now or the other option instead of trying, we also x 0 which created inconsistency. Again, we tried with the other option wherever if there is a conflict we just go back to the last option where we had made a mistake. So, last option taking a wrong thing was taken over here that concept of taking x 0 and it was stating the problem there we stop.

Now, we reverted it back with when we have the same things to apply 0 x, now 0 x is the whole solution has been done because the repair output 0 over here to support the all the problems. This, x is here stands for saying that whatever this D priming whatever may be there, yes the output here is the not dependent able, so 0 over here will solve the problem and also this prime d primer has been propagated to, sorry it is not the D prime d that is the D prime as we have been propagated to the output your job is done.

So, you are propagated the output from here now it has been is the primary output. So, this pattern is state pattern what is this is state pattern this is the best pattern and what is the output or output is normal case one fault case zero. So, this will actually say that mean propagated to the output and your job is done that is you have found out the test pattern. So, this is the test pattern and what is expected in the fault guess that 0 is the output that is what you got the one and the other, so you have stopped over D algorithm this point because this fault value as your primary output and the output values.

So, whatever you have whenever you get the value propagated to the primary unit that is your test patter, but know in the case that there is ever inconsistency in for some reason that other inconsistency here for the time being save this is not in the example. So, let us assume that 0 x would not have solved your purpose, so should have been there that the fault is not testable.

(Refer Slide Time: 55:29)



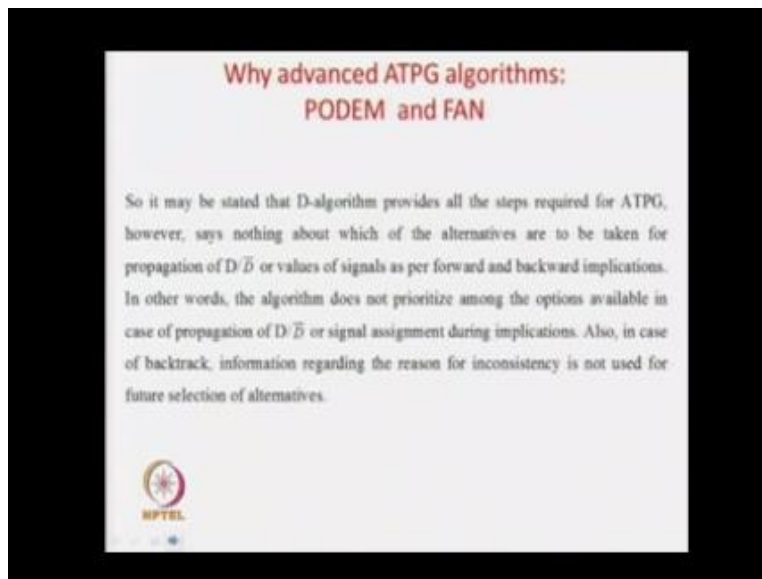
Why the fault tubes have been not testable faults would have been testable because there is no more option available. So, there is no more option available because there was only two D frontier, which was applied and there is another option $x 0$ and $0 x$ are the and get numbers that also you have tried out. So, there is no more options you have to stop it over here, so this was the actually detailed discussion over the algorithm here there is an example. Now, you might have observed that D algorithm will never tell about which of the options is better.

So if somebody would have told us then you could have directly taken in at the gate number three at the D frontier in the first case. Instead of using a $0 x$ that is input have been this 1 has been D frontier instead of using a $x 0$, you can directly take a $0 x$. You have solve the problem, but nobody is there in the D algorithm to tell you which is the better option you take it faster you take it safe there is no multi talents will be there that is what the problem of D algorithm. So, there were some advance algorithm podium and fan actually these steps here that there²³would be a guide better D frontier is a better option to take.

So, that the inconsistency will be less and that is less type of backward propositions that is the basic motivation of the advance D algorithm and mean time. So, in order to going to very details

in this course, but such will give a very brief idea that they are, so I mean that is what is told that the D algorithm propagates and justifies whenever there backtrack is there. So, D algorithm will never tell you that why is that it will tell you why is those backward propagations, but it will not tell you.

(Refer Slide Time: 56:56)



What you can learn from that backtrack, so D algorithm propagates all the steps, but it does not prioritize among the options available and reason for the inconsistency is not used for future selection of alternatives that is very important. So, whenever D algorithm fails, it will never tell you it will never tell it will never learn that this is the case that might there was an inconsistency that better eye is used for learning mechanism in by choice of the next option also. It will not set up fault, but two is the first priority different propagation this option is the first propagation this option is second.


(Refer Slide Time: 57:36)

**Why advanced ATPG algorithms:
PODEM and FAN**

So the advanced algorithms basically improve D-algorithm by providing a guided choice of alternatives based on several testability measures namely,

- After a fault is sensitized, if there are more than one gate in the D-frontier, then choice can be made based on conditions like,
 - The shortest path (in terms of number of gates) from the fault site to a primary output is to be taken
 - The path having gate inputs (other than the one used for fault propagation) with low values of CCI and CO (SCOAP values) are preferred than the ones with higher values.
- If more than one signal values are possible for nets after implication, consider the one that will result in low CCI and CO values.

Use the reasons for inconsistency for future selection of alternatives

 HPTEL

The algorithm is just select the randomly and that is why this is the problem. So this advanced algorithm PODEM and FAN, what they do is that there sometimes they will give the lot of and measures the choices to find out which is the better alternatives like sometimes there may be say that, this is the hottest part from the faults at to the primary output. It says that there are three or n numbers of paths available to take which path you take the path which is the shortest minimum number of gates to the output take that part then that is very good. So, if there is n D frontier shortest path to the primary output can be a choice.

Similarly, you can use SCOAP values to find out there which of the easiest path to drive your faults and which are the difficult faults to drive your path. So, like for example to propagate the fault affect to the OR gate one input should be 0 and there other option which we have to propagate the same output by end gate. So, obviously thus say that fault affect has been propagate to this input of the end gate and this has to be one same fault affect and the propagated through OR gate also say for example time being.

So this is the fault affect being propagated. So, this one you require as one and this one requires as 0 propagated. Now, by CCO value you can find out easily it becomes to make this one or it is

more difficult to make this 0, so whichever is more difficult. You avoid that gate whichever is easier to control for example, if it easier because you have to control and take or gate one is the control you have to take is there. So, with this type of few random disable purist this what actually they will give you the priorities that you take this gate as the first option and for D frontier propagations. You take this singular cover for the justification of the j frontier and then so forth.

There is an inconsistency, it will learn and again reset the priorities. So, this is how the advance PODEM algorithm, a has been developed and D algorithm was the first algorithm in this case. So, D algorithm has its own chance and everybody has learned to D algorithm, what we have seen in the two lectures because that is the billing point of all what do you call, say it is the billing point of all the D algorithm to the backbone of ATPG and whatever ATPG can advances having or made on the background on the you can call the or the base of D algorithm. So, before we close we just see very two quickly the questions use D algorithm, so that the faults below is untestable.

(Refer Slide Time: 59:47)

Questions and Answers

Question 1: Using D-algorithm show that the circuit given below is un-testable:

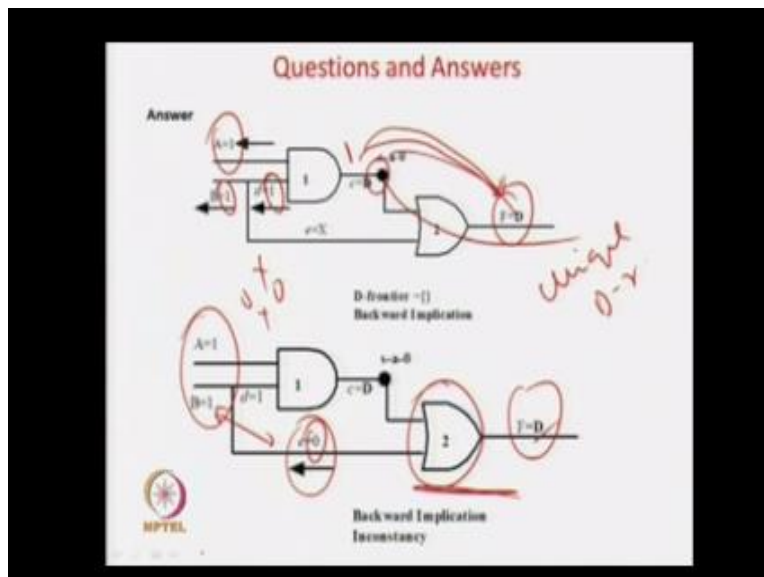
Answer

D-frontier = {2}
J-frontier = {1}

NPTEL

Now, how do you find out that the circuit is untestable, so you have to find out a there is inconsistency then you have to find out that there is no more options available correct. So, in this case everywhere you have to you can see this is the stuck at 0 I have to apply a 1. So, value of d will be there now, so what is the value of what is the value of D frontier for D over here this is the d front here because this is the only gate where fault effect can be propagated and we know that the output is the 1 and this the two inputs of the gates are x, x, x. So this one become your j frontier, so one is j frontier 2 is D frontier correct what we have done,

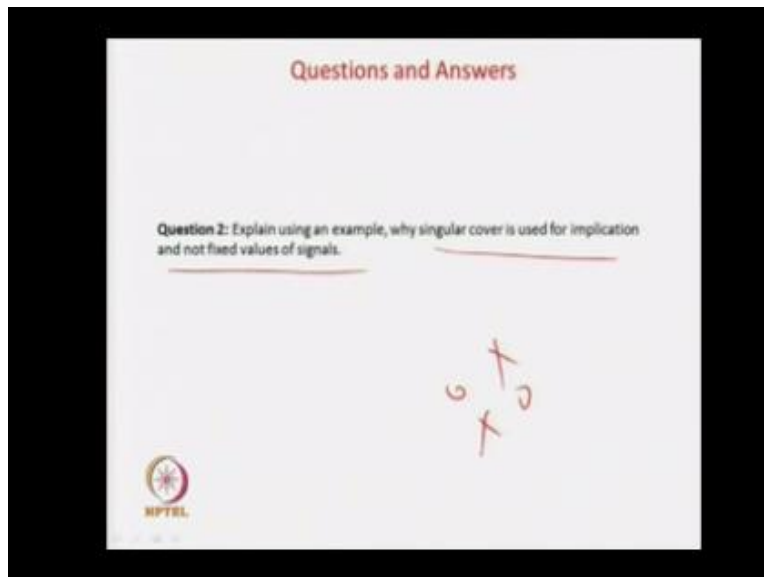
(Refer Slide Time: 01:00:29)



So now you requires a 1 over here, so by backward implications of the j frontier. So, we can say that a in 1 again by backward implications this FAN out, so you require a 1 over here that is the j, now if you want to propagate to the outputs, so this is D for the or gate. So, this one, so be a D, so once you propagate this value here. So, there is no more here D frontier that is very simple, now gains this you can think that two can be part of the j frontier because you know that the output is D and you required something some value here propagate this value. So, this obviously so that 0 is required to propagate the fault affects two and orb gate.

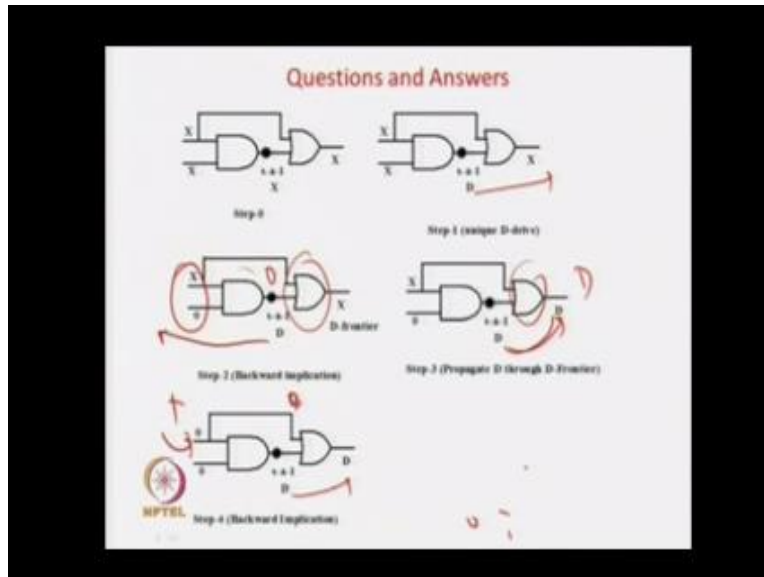
But now you see this is a 0, so than inconsistency over here. Now, there is a inconsistency of the stuck you have to find out very means you have to observe very carefully here that, because no choice that any point because from the here. There was a unique means, you have to observe very carefully here that there was no choice that any point because from the here. There was a unique need that is unique need drive, this as the unique need and no options I mean no point of time, what do you called backward implications of the j frontier, we have any options that is x 0 or 0 x or 1 0 or 1 everything was fixed. So, there is no more options, so you have to say that nothing but you have to say that fault propagations is stopped.

(Refer Slide Time: 01:01:38)



Then, it is stopped that is the only option you can say that is your fault only option you have to say that only the untestable because no more options, so no more backtracks. So, that is 1 example by the questioner you have to that is that went D algorithm say that it has fails to detect a fault. Next example, so the last question actually so explanation the example while singular cover is used for implications and not fixed value of signals that is x 0 that is x 0, x why do you want to keep.

(Refer Slide Time: 01:02:05)

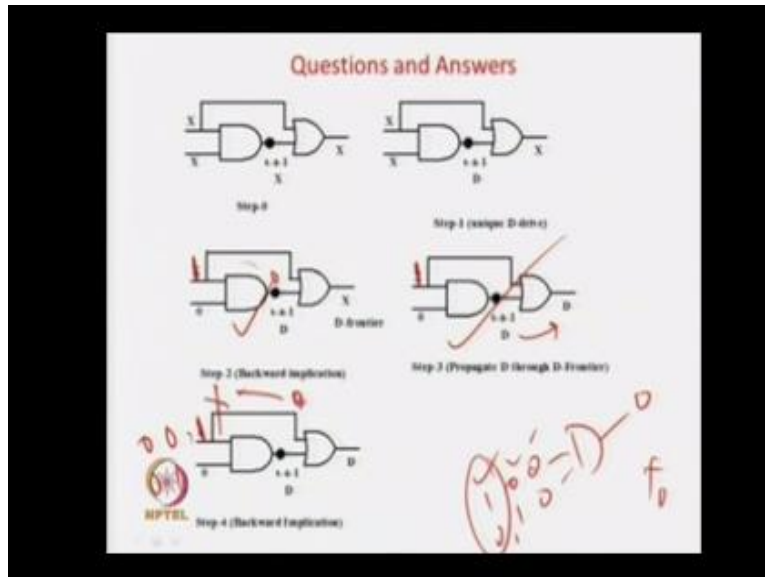


So, many x s in our circuit in D algorithm and not use any other steps what do you mean by any other steps like one because output of OR gate you could have used, sorry one in the OR gate. You could have used x and 1 x or 1, 1, but why you could have used 11 x, but why you are not using 1, 1, 1 why you are not fixing most of the values to keep flexibilities. So, elastic rate using an example say for example, this is your circuit, so you have to put a d over here. Now, the output this is the d next step is this one say unique d drive, so obviously this one will come your D frontier. So, the affect has to be propagated, so you have to propagate the affects faults to the values that is what.

So this is a d over here and if you have affect d over here, then what you require here you require answer of 0. So, this is the case first you have this d propagation, so let us see the step 2, so this D frontier, they affect the we have propagated, but also at the same time you have to propagate the value of a 0 over here. So, gain this two values has to be fixed, now the value has to be fixed means the value of the output of the AND gate is to be 0. This has to be derived by some backward propositions, so you can say that these are part of D frontier.

So this value has to be fixed. So, what is this value has to be fixed, so it is and gate output 0 you can take 0 x or x 0, so in this case I have taken x 0 and you can see that successfully. You will solve the problem already we have seen the x 0, so this is the case, now you can, so d is you know that the value of d can take that j D frontier. You just require a 0 over here, so it was x, so you could have put a x it was x and immediately you cut this x sorry it was x. So, immediately what you could have done, so this x is there if you convert, sorry it was x you convert the x through 0 and your job is done.

(Refer Slide Time: 01:04:01)



So, that is the problem is solved, but now has the person been selected for hart coated values like for the end gate output to be 0 IIT can be 0, 0, 0, 1 or 1, 0, so instead of using x 0. I could have used anyone of this, so for this 1 there is coating of some values, so let this person has use 1, 0 still your problem is solved you require a 0 over here. If I put a 1 0, here you problem is done, now you have propagate the values find D frontier has been propagated. So, this is 1 because he has 1 as not use this x 0 or 0 x concepts that are keeping flexibilities. So, here hart coated, so it was successfully here, but now you see it was a 1 because you have not used x.

So, there could have been many inconsistencies because you require a 0 over here that this is the one and you will declare that there is a big problem over here and that is of the circuit fault cannot be tested. Again, you have to try out with the other options like 0, 1 or 0, 1 all these are options here to try out, but by keeping a x all this problems are gone.

That is what the emphasis that D algorithm, you have to keep more number of x's to keep your options open. You have to deal, therefore inconsistency has to be possible, so that was about the lecture on ATPG or combinational circuits using D algorithm in the next section, in next lecture from the next class. We will study in details about sequential circuits, so till now all our discussions are limited or combinational circuits, so with this we close. 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