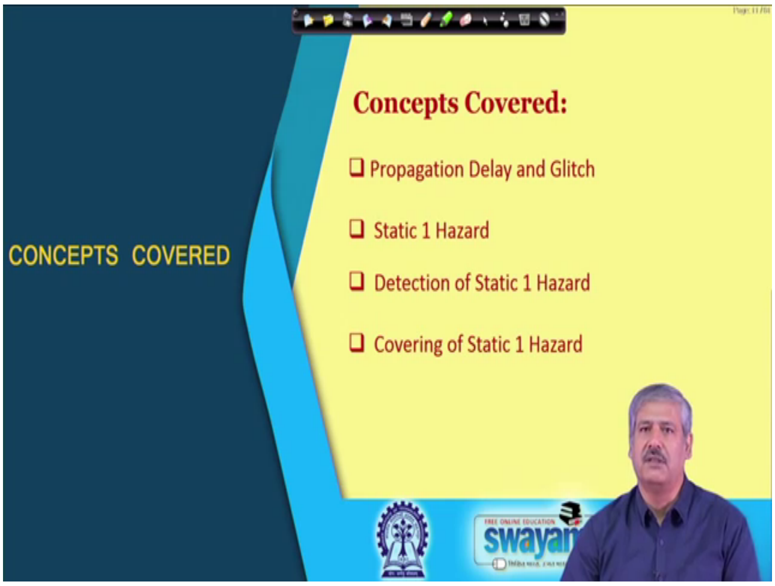
Digital Electronic Circuits

Prof. Goutam Saha
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 14
Static 1 Hazard

Hello everybody. In the last few classes, we were looking at minimization of circuit, a logic circuit and different methods, different ways. And in the last class also we defined certain cost, and we were looking for getting a lower cost in terms of hardware that is being used for realization.

(Refer Slide Time: 00:31)



CONCEPTS COVERED

Concepts Covered:

❑ Propagation Delay and Glitch

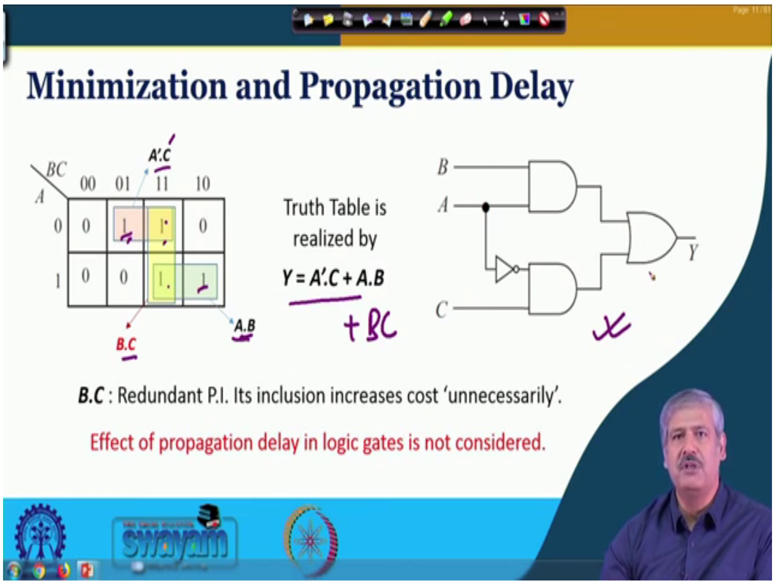
❑ Static 1 Hazard

❑ Detection of Static 1 Hazard

❑ Covering of Static 1 Hazard

In this particular class, we shall look at some of the practical issues associated with logic circuit implementation. And for that how we modify the circuit, which is which may be more costly in terms of the hardware use the way we have defined cost, but it saves the day, it helps in work coming some of the inherent problems associated with the logic circuit. And the concepts to be covered, which is related to propagation delay and the glitch that is found, and we shall do alone static 1 hazard in today's class.

(Refer Slide Time: 01:15)



Minimization and Propagation Delay

A \ BC	00	01	11	10
0	0	1	1	0
1	0	0	1	1

Truth Table is realized by

$Y = A'.C + A.B$ $+ BC$

B.C : Redundant P.I. Its inclusion increases cost 'unnecessarily'.

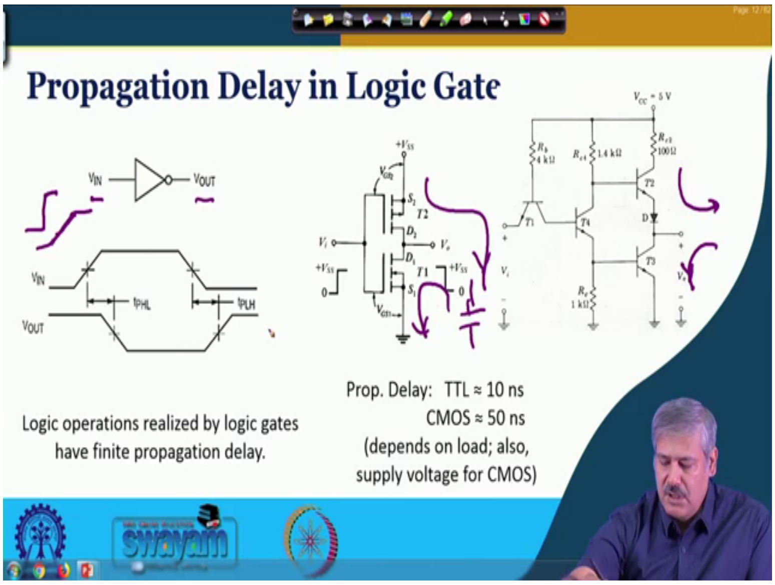
Effect of propagation delay in logic gates is not considered.

So, we look at again a very simple problem, the kind of problem we have seen before. So, this is a minimization of a truth table using Karnaugh map, where there are four min terms present. And this for this min term, this one and this one are covered by A prime C. This one and this one are covered by AB ok. And there is another way we can have this one and this one covered by BC right, but this two ones are already covered by AB and A prime C.

So, AB and A prime C are essential prime implicants essential, because they are covering either this one or the other one, which is not covered by anyone else. And in this case, BC becomes non-essential, so it is redundant ok. So, the circuit for the equation that we get for this truth table for minimized expression is A prime C plus AB. And this is how you realise a circuit ok? We could have a we could have obtained we could have a heard a BC also associated with no difference in meaning, but that would have unnecessarily increase the hardware cost ok, because BC means another and gate and that will get connected ok.

So, this is how we have seen the minimization problems so far and circuit realization. And in doing this what we have considered that I mean we did not consider the effect of propagation delay in the logic gates ok, this was not considered. We just went by the Boolean expression what will give us the minimized circuit ok. So, this is the background for with which we start today's class.

(Refer Slide Time: 03:22)



Propagation Delay in Logic Gate

Logic operations realized by logic gates have finite propagation delay.

Prop. Delay: TTL ≈ 10 ns
CMOS ≈ 50 ns
(depends on load; also, supply voltage for CMOS)

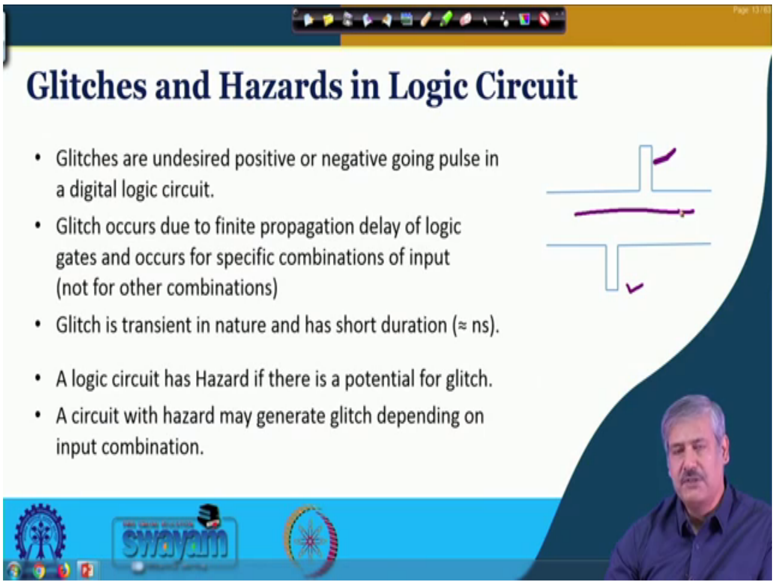
But, we note takes before that there is propagation delay in a particular logic gate, whatever logic gate we talk about ok. So, this propagation delay is associated with for example, now we are looking at it in terms of you know voltage that is presented ok. So, earlier we were writings 0, 1; 1, 0 and all which is fine, but if you look at input voltage and the output voltage, so if you were stretching it in time and all actually, we will see that it is not very sharp kind of you know rise, so there is a kind of you know finite rise time and finite fall time.

So, in the input side the time the 50 percent of when it reaches the 50 percent level as a inverter. And corresponding output that falls, output will go down go low right, when it reaches the 50 percent value. This is the propagation delay, which is at the output which is high to low ok. This is propagation delay at the output for output going high to low. And similarly, the opposite is low to high that is input is at you know 50 percent level, when it is going form high to low from that output is going from low to high, when it reaches the 50 percent level. So, this is propagation delay at the output low to high ok.

So, and this is because of you know usually, they will be little bit different from one another. Because, if you look at the CMOS, there is a capacitancy we had noted before load capacitance right. So, this charging path, so this is the charging path and this charging path is like this right. So, these paths are different.

Similarly, in this case this is one path and this is another path right. So, there is a small difference between these two values ok, but average propagation delay is taken is the you know we add them up, and then divide by 2 ok. And for TTL, it is of the order of the you know 10 nanoseconds, CMOS for same you know 5 voltage circuits like that of TTL, it is of the order of 50 nanosecond. But depends on load and also for CMOS if you increase the voltage level say two say 10 volt instead of 5 volt, it will become say 30 nanosecond of that order ok, but that is a finite propagation delay which is to be noted in this particular discussion ok.

(Refer Slide Time: 06:06)



Glitches and Hazards in Logic Circuit

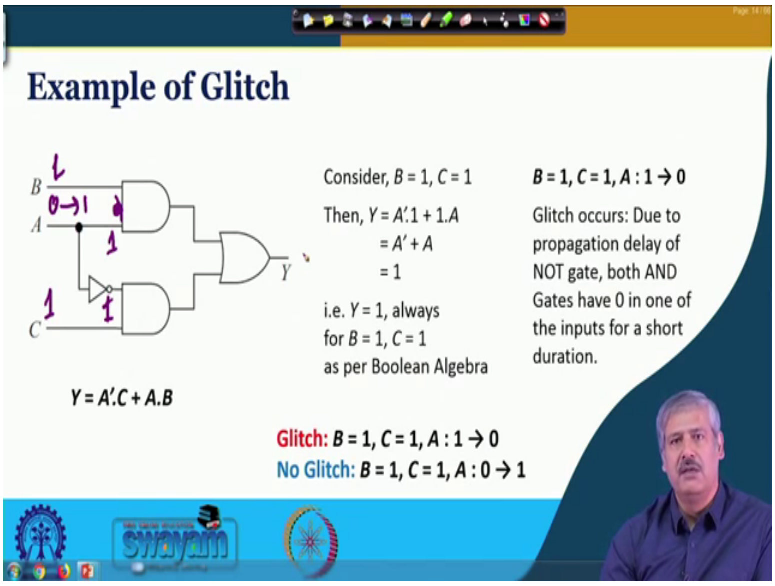
- Glitches are undesired positive or negative going pulse in a digital logic circuit.
- Glitch occurs due to finite propagation delay of logic gates and occurs for specific combinations of input (not for other combinations)
- Glitch is transient in nature and has short duration (≈ ns).
- A logic circuit has Hazard if there is a potential for glitch.
- A circuit with hazard may generate glitch depending on input combination.

Now, what is its effect because of this is what happens, because of this we will find in the circuit there may be glitches ok. And so we will see how it occurs, but before that let us define what is glitch. Glitches are undesired positive or negative going pulses. Undesired means, this is positive going pulses and this is negative going pulses. Undesired means it was not supposed to go, so it was supposed to be you know straight like this right, but then there is some such thing happening.

And glitch occurs due to finite propagation delay of logic gates and occurs for specific combination of input. So, for certain specific combination of input only it will happen, it is not occurring all the time. For another combination, it will not it may not occur ok. And glitch is transient in nature; it is there and for a very short duration of the order of the propagation delay. So, if something in the input is getting changed, so momentarily it will occur, and then it will disappear, in the stable state it will not be there ok.

And the other definition which we need to take note of is called hazard. A logic circuit has a hazard, if there is a potential for glitch. So, circuit which is hazard free, there is no such potential ok. There is no such combination at the input some changes for which glitch will occur, so that is a completely hazard free circuit ok. And a circuit with a hazard may generate glitch, if those specific input condition or the pattern is presented ok, so this is we take note of right.

(Refer Slide Time: 07:56)



Example of Glitch

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

Consider, $B = 1, C = 1$

Then, $Y = A'.1 + 1.A$
$= A' + A$
$= 1$

i.e. $Y = 1$, always
for $B = 1, C = 1$
as per Boolean Algebra

$B = 1, C = 1, A : 1 \rightarrow 0$

Glitch occurs: Due to propagation delay of NOT gate, both AND Gates have 0 in one of the inputs for a short duration.

Glitch: $B = 1, C = 1, A : 1 \rightarrow 0$
No Glitch: $B = 1, C = 1, A : 0 \rightarrow 1$

And then we shall look into a examples. So, we go back to our or this previous circuit only, the circuit with which is started, this is the one ok. And this was our reduced circuit reduced you know relationship optimized obtain up right, and this corresponding circuit. Now, we consider a specific case say B is equal to 1, C is equal to 1 ok, so at that time what will happen, what is the value of the output B is equal to 1, and C is equal to 1 right.

So, in the truth table sorry in the truth table if you look at it, you will find that the B is equal to 1, C is equal to 1 is one ok, and it is obtained from the equation also. If you just substitute it here, so A prime 1 1 A prime right, so this is the cases you get a A prime plus A and this is 1, so that means y will always be 1 as per the Boolean algebra right, so this is the case.

Now, you consider that B is at 1, so this is at 1. C is at 1 right, and A goes from 1 to 0 so A goes from 1 to 0 right. So, when A was 1 in the stable condition before this change takes place, so this NOT gate output was 0. Now, when A A goes from 1 to 0, so immediately this input is become becoming 0 at that time point. But, this NOT gate output will not go to 0 will not go to 1 immediately because of the propagation delay just now we had considered in the previous slide. So, it will go to 1 after some time right.

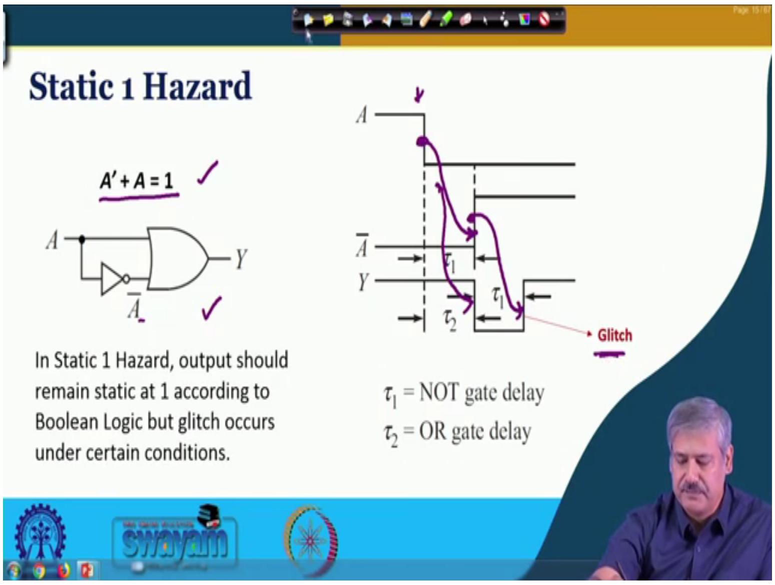
So, for that amount of period for that amount of period for which this NOT gate output you know goes from output goes from logic low to high ok. So, till that particular time that particular delay output propagation delay low to high at that time this remains at 0. So, this AND gate output as a 0, after the of course the propagation delay, it will get the 0, and this will this is already in 0 all right.

So, what we see that OR gate both the outputs are 0, even for small duration after that it will become 1. So, for that small duration after again it is own propagation delay the OR gate will generate A 0, which was otherwise 1 ok, so that is the after that propagation delay of course, it becomes 1 and then there is no issue. One of the OR gate input is one except for that transition period, is it clear.

So, this is the case when it is happening, when we see that a glitch is occurring a for that with this OR gate output is monetarily going to low ok. But, if B is equal to 1, C is equal to 1, and A goes from 0 to 1 will there be any issue, will there be any glitch? So, this is 0 so this is 0. And this is 1 means this is 0 means, this was already in 1, then change takes place. So, this 0 becomes 1 right all immediately.

But, this 1 is held, 1 will go to 0 after the propagation delay right, but by that (Refer Time: 11:55) this AND gate output both of them are 1, so this is 1, so OR gate will be held at 1 there is no issue, so that is what is known that means, what was talked about as the specific input combination only the glitch will occur. This circuit is having a hazard this is what hazard free circuit, because there is a potential glitch, but glitch will occur only when a specific combination is presented ok.

(Refer Slide Time: 12:21)



Static 1 Hazard

$A' + A = 1$

In Static 1 Hazard, output should remain static at 1 according to Boolean Logic but glitch occurs under certain conditions.

τ_1 = NOT gate delay

τ_2 = OR gate delay

Now, we define static 1 hazard, what is it? Static 1 hazard is the case when output should remain output is to remain at static 1, static at 1 according to Boolean logic algebra the expression, but glitch occurs under certain condition. So, essentially what we had seen that somehow the circuit boils down to a condition, which is after you know considering all the different inputs, you know the values of those inputs an expression something like this ok, so that is that gives rise to the static 1 hazard ok.

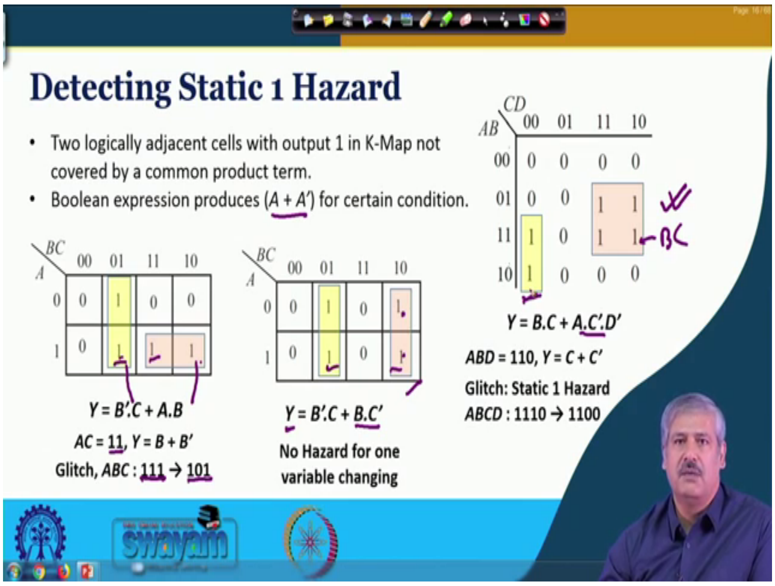
If the somehow this kind of expression is there, the and the corresponding realization if you look at it, if you are having it, in terms you know circuit is realization, so this is the essence of it, finally you know it boils down to some a reduced circuit like this ok. There may be additional gates and all, so those delays will be considered, but this is the basis of the static 1 hazard that is occurring ok.

So, how again we whatever we just mentioned, if we try to put it in the form of diagram for this particular circuit. So, this is A right going from 1 to 0 over here at this time point, at this time point ok, and because of this A bar will go high after the corresponding delay propagation delay logic low to high right.

So, from this point to this point, there is a delay right. And momentarily you see at this point of time, this and this both are low ok, so because of which OR gate output after it is own propagation delay, which is high to low right given by tau to over here ok. So, it will go to low, because both of them are low at this point, so this is going to low ok.

So, tau 1 and tau 2 could be different, but as I said in the same logic family we usually take them as you know same tau and the low to high, and high to low also can be different, but more or less for this you know depiction, we can show them as same. And after that this propagation delay this goes to high, this output will go to high, and because of which again OR gate output will go to 1 right after it is for until remain low for that period only is it clear ok. So, this is how the glitch is occurring.

(Refer Slide Time: 15:34)



Detecting Static 1 Hazard

- Two logically adjacent cells with output 1 in K-Map not covered by a common product term.
- Boolean expression produces $(\underline{A + A'})$ for certain condition.

A \ BC	00	01	11	10
0	0	1	0	0
1	0	1	1	1

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

$AC = \underline{11}, Y = B + B'$

Glitch, $ABC : \underline{111} \rightarrow \underline{101}$

A \ BC	00	01	11	10
0	0	1	0	1
1	0	1	0	1

$Y = B'.C + \underline{B.C'}$

No Hazard for one variable changing

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	1	0	1	1 ← BC
10	1	0	0	0

$Y = B.C + A.\underline{C'.D'}$

$ABD = 110, Y = C + C'$

Glitch: Static 1 Hazard

$ABCD : 1110 \rightarrow 1100$

Now, how we can detect you know static 1 hazard, so is there any rule for that ok. So, yes there is a rule, there is a method by which we can detect static 1 hazard. And later on we shall see if we can cover those hazards, we can make the circuit hazard free ok, but how we can do that? So, for that we have already noted the Boolean expression, finally you know reduces to produce a sum you know A plus A prime kind of thing for certain condition certain specific condition of the input. The other important thing that we can look at even you know without going into a optimized expression or so, just by looking at the Karnaugh map, we can figure out the way we form the groups whether there is a potential you know glitch there or not ok. So, how we can find it out?

So, let us look at this example, similar example similar to previous example ok. So, these are the two 1s and these are the two 1s, so obviously we are forming a group like this, these two prime implicants ok. So, is there a potential hazard yes, the rule says if the two logically adjacent cells with output one in Karnaugh map is not covered by a common product term ok. So, this one and this one are adjacent. They are not covered by one common product term, it is covered by this is this group is one generating one product term, this group is generating another product term is not it ok. So, it is having a potential static 1 I mean it is having a static 1 hazard potential glitch under certain condition it might occur.

So, let us see how it occurs. So, we can write from this Y is equal to B prime C, this one is B prime C, and this one is AB right. This group is AB, and this is B prime C right. Now, when AC is equal to 11, A and C are 11. We get Y is equal Y is B plus B prime, this is a case of you know static 1 hazard the kind of thing that we have seen. So, if the input is 111, and then from that it goes to 101 that means, B goes from 1 to 0 the way we have seen before, then a glitch negative going pulse will be there ok, so that is what we see from this particular case. We can identify right, this is not covered ok.

Let us look at this example. So, this is an example where you know there are two such you know prime implicants B prime C and BC prime ok. So, is there any hazard here? So, there is no logically adjacent cells which is not having a common terms. So, this is these two are logically adjacent a common product term, these two are logically adjacent common product term ok. And these two are not logically adjacent say this one, this one not logically adjacent. So, they are not need not be covered ok.

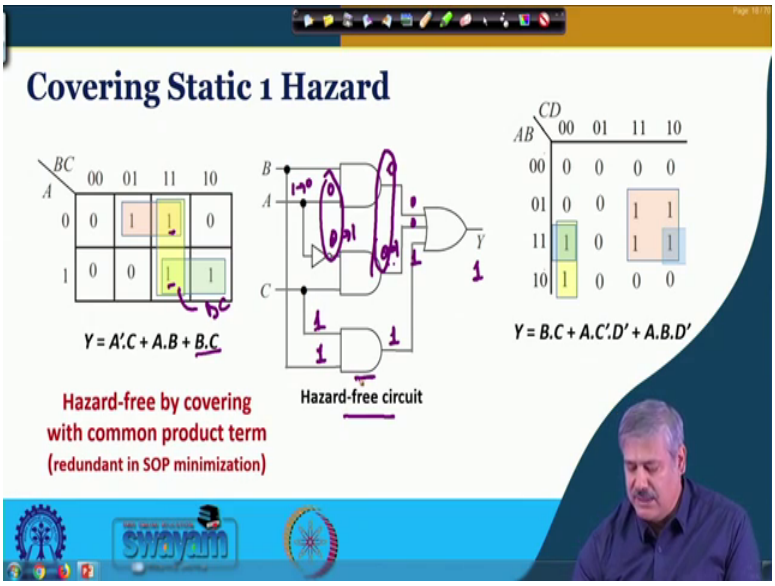
So, there is no such you know common term is required. So, this is not having any hazard ok, because we are in all these cases we are considering only one variable changing right. And so, if you see that, if you consider say B is equal to 0 right, then this term will not be there, it is only B prime C, so it will follow if C changes with appropriate propagation delay output will change accordingly ok. So, there is no such hazard.

So, these were three variable problem examples, we look at a four variable example. So, this is the corresponding Karnaugh map of a given truth table ok. So, the way we optimize get a optimized expression for this circuit is a group of four over here, and a group of two right. So, this is generating you ABC right, this is generating BC and this is generating A, C prime, D prime this term right ok. So, you just add them and you get the minimized SOP realization, does it have a static 1 hazard.

Let us look at if there is a logically adjacent one, so this one, this one and this one, they are logically adjacent, which are not having a common product term right. So, there is a potential static 1 hazard. And when does it occur or which combination, so it occurs this is the thing, so ABD. So, we can see that AB 11 and between these two what is common D with 0, so ABD-110. So, this is common between these two 1s right; and C going from 1 to 0 C going from 1 to 0, because we know that from high to low, then there is you know a static 1 hazard.

So, in that case ABD; ABD remaining A and B remaining at 1, and D remaining at 0, these are the cases. And C going from 1 to 0, we are having a glitch ok. And also from Boolean expression also we can find it out right. So, this is just examining that whether you know this rule prevails or not, and we can find out the condition also by this way the glitch will occur, and so this is how we can detect static 1 hazard, but knowing that there is a static 1 hazard that is not enough

(Refer Slide Time: 21:59)



Covering Static 1 Hazard

A \ BC	00	01	11	10
0	0	1	1	0
1	0	0	1	1

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

Hazard-free by covering with common product term (redundant in SOP minimization)

Hazard-free circuit

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	1	0	1	1
10	1	0	0	0

$Y = B.C + A.C'.D' + A.B.D'$

. We need to see that we are able to cover it ok, we are able to cover it and the circuit does not have hazard, because those negative going pulse or positive going pulse, I mean in this case negative going pulse for static 1 hazard might cause some you know trigger certain events, which is undesirable ok.

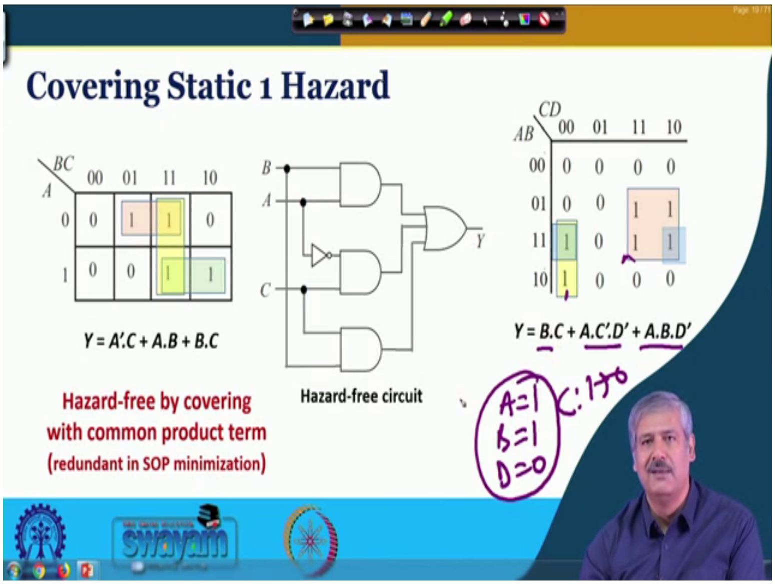
So, in the example that we have taken before right; so since it is caused by the I mean during the detection process, we have seen that it is caused by the logically adjacent terms not coming under one you know one common you know product term that is the cause of it. So, the solution is to have a common product term that covers logically adjacent ones in the output of a function in a Karnaugh map ok, so that is the solution. So, in this case the example that we started with right; so this one and this one did not have the logically you know have a one common product term.

So, now we look at one common product term that can be there. So, this common product term is nothing but your BC ok, which we saw there in the first place, but we abandoned, because of higher cost that was getting associated with it, I mean when we include it that is a redundant term right. But, in this case when you associate it ok, the hazard gets covered and we can get a hazard-free-circuit ok. So, this B and C this was the earlier circuit, now this B and C is coming over here. And how is it helping? So, B and C 11, then only the hazard I mean the glitch was there right. So, B and C 1 and 1, what is happening this is 1, this is 1, so the output is held at 1 the OR gate any output is 1 is output is 1 right.

So, even if A changes from 1 to 0, and you know there are some changes happening over here, some changes happening here, momentarily they are becoming 0, this is becoming 0 for some time, this is becoming 0 for some time ok. So, 1 to 0 A 1 to 0, so this has become 0, it is still 0 right, it is taking some time to go to 1.

So, this is the time we are talking about when both the outputs are 0 right, it is taking some time to go to 1 ok. But, these two 0's even if it is there, this 1 over here is suppressing the glitch that could have had been place unless a could have been there at this output is not there ok, this additional circuit ok. So, yes it is more costly, but it is reducing removing the hazard would not costly in terms of you know ordered cost, but it saves some other cost in terms of performance, where this glitches are the can cause damage ok. So, this is the thing that we have to keep in our mind.

(Refer Slide Time: 25:55)



Covering Static 1 Hazard

A \ BC	00	01	11	10
0	0	1	1	0
1	0	0	1	1

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

Hazard-free circuit

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	1	0	1	1
10	1	0	0	0

$Y = \underline{B.C} + \underline{A.C'.D'} + \underline{A.B.D'}$

Hazard-free by covering with common product term (redundant in SOP minimization)

A=1
B=1
D=0
C: 1→0

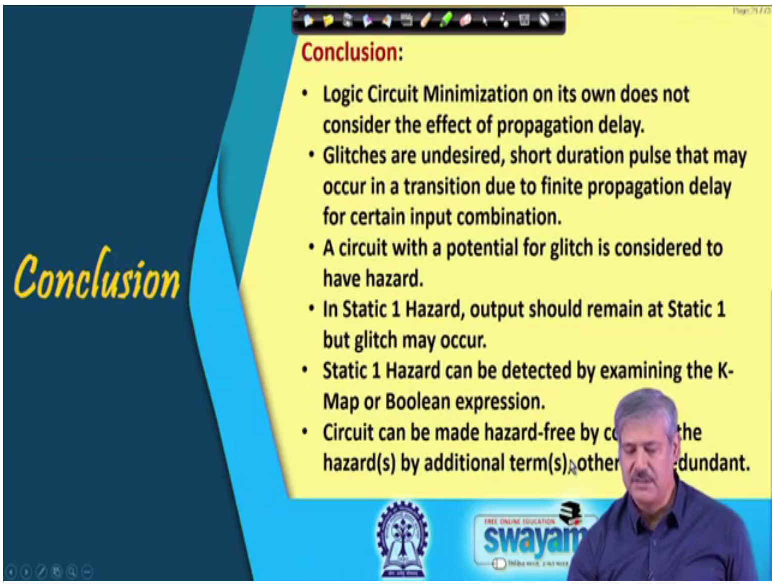
Finally, we look at the other example we had discussed before, so this was the circuit. And two these were the two terms this was one term and this has another term BC and A, C prime, D prime all right. And this is the so called redundant term right.

But, this generates a product term, which connects which is common across logically adjacent ones ok, which is otherwise would not be connected, so that is the ABD prime. And this ABD prime if you remember that was the condition at the input side, when A is 1, B is 1, and d is equal to 0, and C was changing from 1 to 0 the glitch was occurring ok.

So, by having this three and the cases, I mean ABD prime connected here for this particular cases, when the glitch could have been there putting such additional a three input AND gate, and connecting to a OR gate right, you are suppressing that glitch ok. So, this is what we keep in mind. And also we take note of that it is not that the circuit may have only one hazard, they can be multiple hazard also I mean multiple such possible cases all right.

For example, if in this particular example if we just simply modify like this is 1 and say this is 1 right, then minimized representation will be like this one and the other one. Now, to make it hazard free, so there are now two potential if you look at it, there is other two such potential glitches ok. One is this is the one that we have already seen, another is this particular case ok. These two are logically adjacent, but not covered by any common term, common product term in the most minimized you know case. So, to cover this we need to generate this term and add it here for you know that particular minimized expression.

(Refer Slide Time: 28:16)



Conclusion:

- Logic Circuit Minimization on its own does not consider the effect of propagation delay.
- Glitches are undesired, short duration pulse that may occur in a transition due to finite propagation delay for certain input combination.
- A circuit with a potential for glitch is considered to have hazard.
- In Static 1 Hazard, output should remain at Static 1 but glitch may occur.
- Static 1 Hazard can be detected by examining the K-Map or Boolean expression.
- Circuit can be made hazard-free by co the hazard(s) by additional term(s) other dundant.

So, this is we take note of. And with this we summarise what we discussed today that logic circuit minimization on its own does not consider the effect of propagation delay. Glitches are undesired, short duration pulse that may occur in a transition due to finite propagation delay for certain input combination. A circuit with a potential for glitch is considered to have hazard.

In static 1 hazard, output should remain at static 1, but glitch may occur for that specific combination. Static 1 hazard can be detected by examining the K-map or the Boolean expression. K-map comes first. So we can look at the K-map and form that we can look at these logically adjacent ones or logically adjacent ones are covered by one common product term or not. And circuit can be made hazard-free by covering hazards by additional terms, which are otherwise considered redundant ok.

Thank you.