Switching Circuits and Logic Design

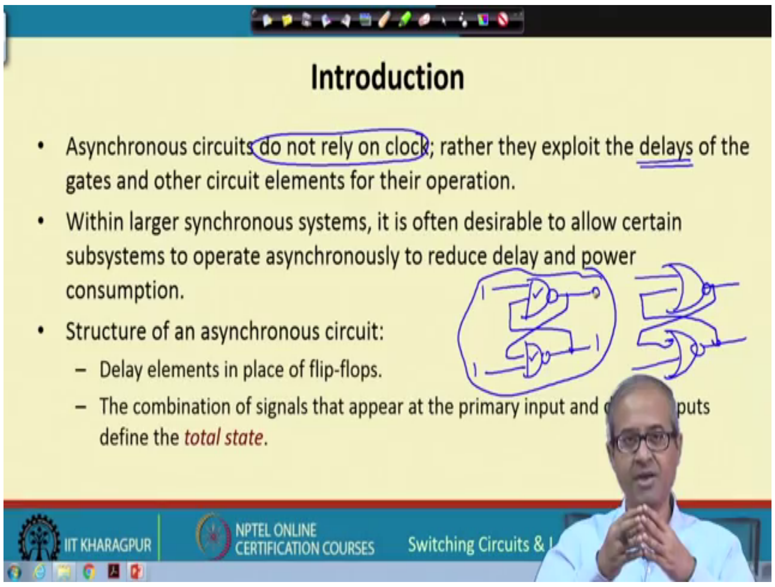
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 52
Asynchronous Sequential Circuits (Part I)

So, in this lecture, we start our discussion on Asynchronous Sequential Circuits. You see the kind of sequential circuits that we have discussed so far is basically synchronous in nature. What is meant by synchronous? There is a clock; all operations in the circuit are carried out in synchronism with the clock. When I say in synchronism, it means that there are flip flops which store the state of the circuit, the machine and with the clock this state changes occur in synchronism. But in an asynchronous circuit, there is no concept of flip flops or a clock. Everything happens with respect to the delays of the circuit element gates etcetera, right.

So, naturally speaking, if we have this kind of a scenario, there is no clock to synchronize, the design of such circuits is much more difficult in general. Let us look at some of the aspects of asynchronous sequential circuits, fine.

(Refer Slide Time: 01:31)



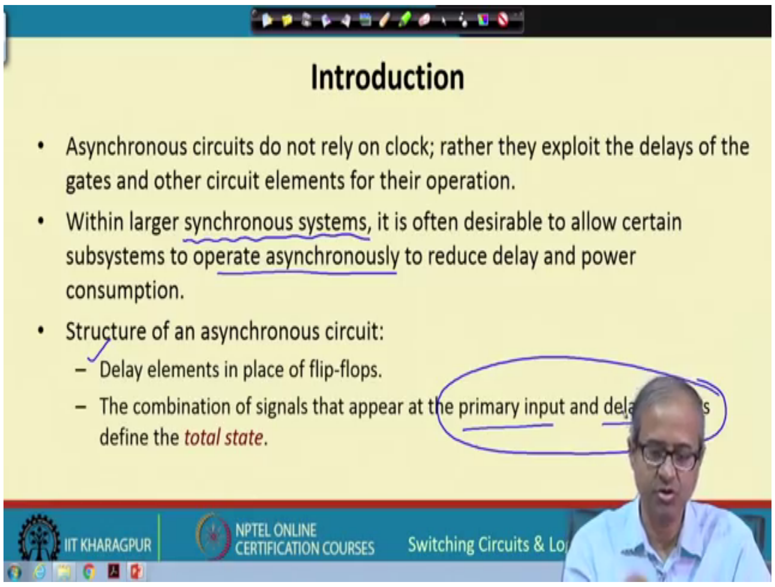
Introduction

- Asynchronous circuits do not rely on clock; rather they exploit the delays of the gates and other circuit elements for their operation.
- Within larger synchronous systems, it is often desirable to allow certain subsystems to operate asynchronously to reduce delay and power consumption.
- Structure of an asynchronous circuit:
  - Delay elements in place of flip-flops.
  - The combination of signals that appear at the primary input and outputs define the *total state*.

So, as I have said the asynchronous circuits in general, they do not rely on a clock. What do they rely on? They exploit the delays of the gates and other circuit element. Let us take an example. Well, one asynchronous circuit element you have already seen without well explicitly knowing about it. Well, you think of a one-bit latch. A one bit latch you can implement using cross coupled NAND gates or cross coupled NOR gates, both way you can implement, well.

While we are discussing flip flops, we saw this kind of one-bit latch designs. Now, you see this circuit I means in itself is a sequential circuit, because it can store some data. So, once I store some data 0 and 1, so, as long as I am applying 1 1 here, it remembers this value. Because 1 and 1 comes 1 1 is 0, it remains 0, 0 and 1, it is 1; it remains 1. So, it can memorize something which means it is a sequential circuit. But there are no flip flops, only gates. The way this 0 and 1 are remembered is based on the delay of these gates. So, after some delay this is computed again 1 is generated, after some delays this 0 is generated. So, this is a classic example of an asynchronous circuit where operation is determined by the delays of the circuit element here gates. Let us move on. But as it said design of asynchronous circuit is difficult.

(Refer Slide Time: 03:43)



Introduction

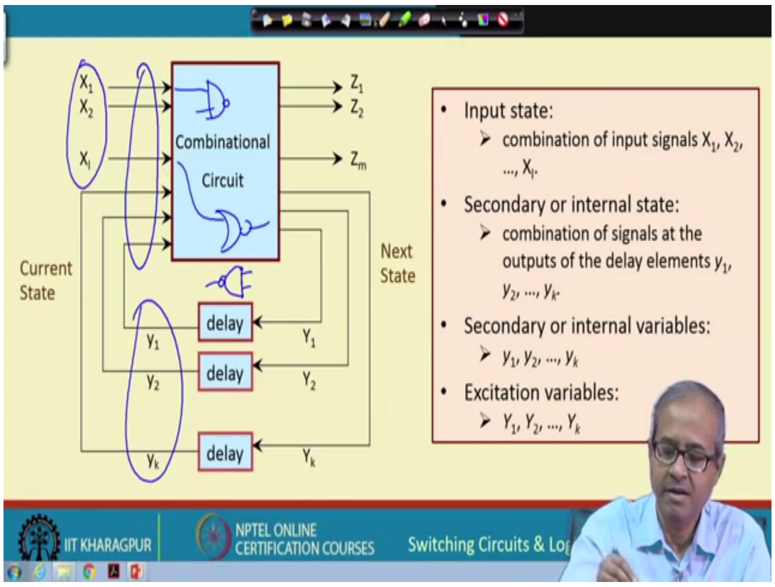
- Asynchronous circuits do not rely on clock; rather they exploit the delays of the gates and other circuit elements for their operation.
- Within larger synchronous systems, it is often desirable to allow certain subsystems to operate asynchronously to reduce delay and power consumption.
- Structure of an asynchronous circuit:
  - Delay elements in place of flip-flops.
  - The combination of signals that appear at the primary input and del s define the *total state*.

So, in general when we design large systems, they are synchronous in nature. But sometimes what you do? As part of a large systems smaller subsystems can be asynchronous. Just like in a flip flop, this cross coupled latches were there. The reason is that in general asynchronous circuits run faster because they do not have to wait for the clock to come. So, whenever the input changes depending on the gate delays, the output can be immediately computed, this is what is done..

So, if we have a large subsystem like this as I said certain smaller subsystems can be made to operate asynchronously, right. And if you look at the general structure of an asynchronous circuit, there are 2 differences you will find in general. There are delay elements in place of the flip flops.

So, we will come to this and the combination of the signals on the primary input and the delay outputs; just like in a synchronous sequential circuit, the flip flops determine the state, but here states of the primary inputs, as well as the delay outputs determine the state. This is called total state.

(Refer Slide Time: 05:13)

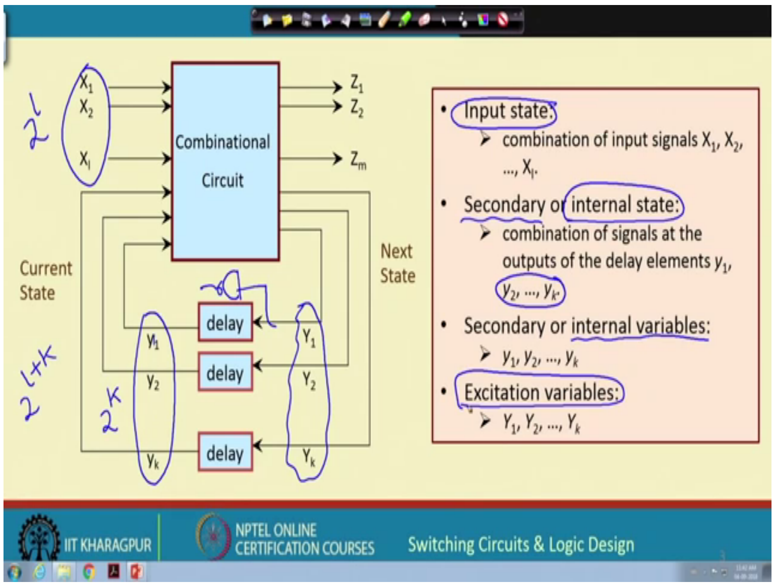


- Input state:
  - combination of input signals $X_1, X_2, \ldots, X_l$.
- Secondary or internal state:
  - combination of signals at the outputs of the delay elements $y_1, y_2, \ldots, y_k$.
- Secondary or internal variables:
  - $y_1, y_2, \ldots, y_k$
- Excitation variables:
  - $Y_1, Y_2, \ldots, Y_k$

Let us look into this, let us look at a structure diagram first, like this. This diagram you see on the left, this is very much similar to what we have seen for a synchronous sequential circuit. But the difference is that, in place of this delay elements you see out here there you had flip flops and the flip flops were triggered by a common clock signal. We said in synchronous circuits that whatever you are storing in the flip flops that determine the state of my machine.

But for an asynchronous circuit, because we do not have any separate flip flops. Everything is determined by the gates. This delays that we are talking about here, this delay can be a NAND gate, let us say delay is just a NAND gate a gate. These inputs are coming, the inputs are also going into some gate, right. So, there are such delay elements everywhere in the circuit. So, when you talk about the total state I talked about, so, it is not only the primary inputs, but also the output of the delay elements. The total inputs that are being applied to the combination circuit that determines the state of the system. Now, there are a few terminologies let us look at.

(Refer Slide Time: 06:53)

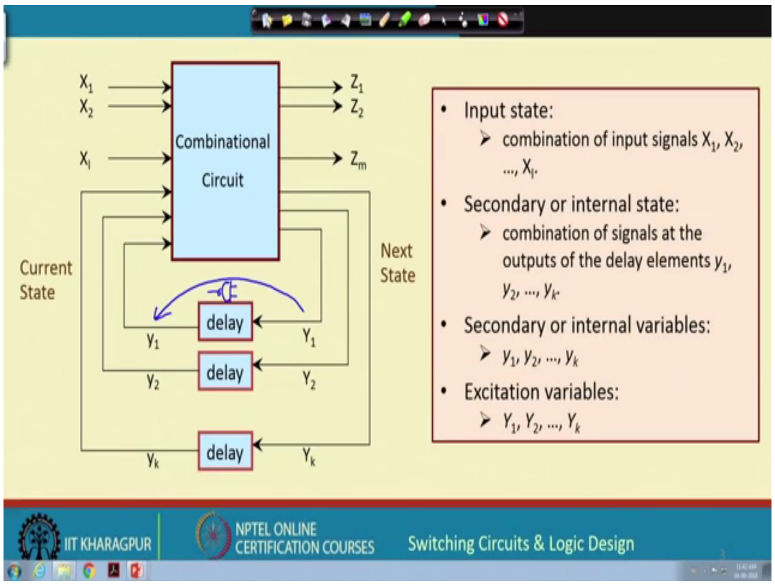


- Input state:
  - combination of input signals X_1, X_2, …, X_l.
- Secondary or internal state:
  - combination of signals at the outputs of the delay elements y_1, y_2, …, y_k.
- Secondary or internal variables:
  - $y_1, y_2, …, y_k$
- Excitation variables:
  - $Y_1, Y_2, …, Y_k$

There is something called input state. Let us say there are l input variables X 1, X 2 to X l. So, there can be 2 to the power l possible combinations. They determine something called input state. So, some input is applied to a combination circuit. Then you have some internal state which is sometimes also referred to as secondary state. This refers to the combination of the outputs of the delay elements, the small y 1 to small y k. There are k delay elements here, I am showing schematically. So, there can be 2 to the power k such internal states. And total states when you talk about, it will be 2 to the power l multiplied by 2 to the power k; 2 to the power l plus k..

Now, this y 1 to y k is sometimes also called secondary or internal variables. And the signals that are fed to the input of this delay element let us say gates as an example I said. These are referred to by capital Y 1 to capital Y k. These are referred to as excitation variables right ok. Now, the point to note here is that, you see for an asynchronous sequential circuit, there are no clocks. There is no signal coming from outside that tells you when this Y 1 will be copied to small y 1.

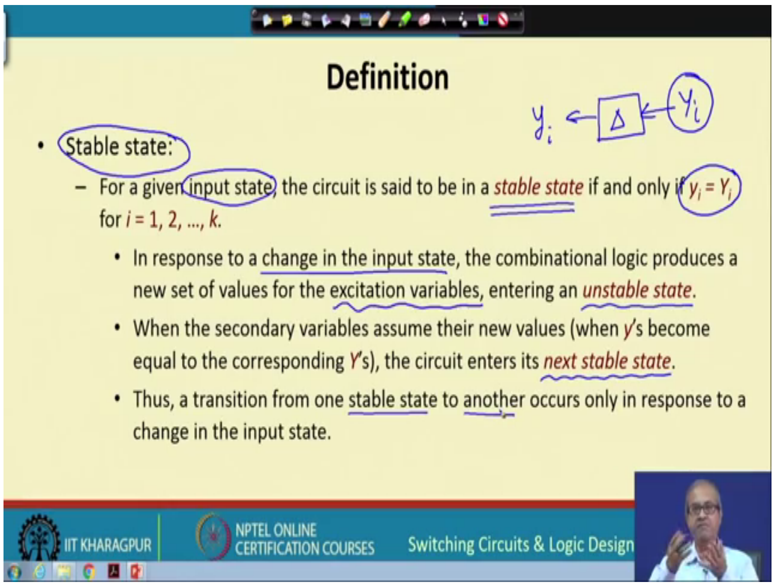
(Refer Slide Time: 08:35)



- Input state:
  - combination of input signals $X_1, X_2, \ldots, X_l$.
- Secondary or internal state:
  - combination of signals at the outputs of the delay elements $y_1, y_2, \ldots, y_k$.
- Secondary or internal variables:
  - $y_1, y_2, \ldots, y_k$
- Excitation variables:
  - $Y_1, Y_2, \ldots, Y_k$

For a synchronous circuit the clock did that. But here this will happen after the delay of the gate, I showed here a gate, let us take that same example. After this gate delay this will happen. So, when some input is applied the circuit will go through some transition region some temporary states before they get stabilized into y 1 to y k. There is a concept of stable state that comes into the picture here, ok.

(Refer Slide Time: 09:13)



$y_i \leftarrow \boxed{\Delta} \leftarrow Y_i$
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

Definition

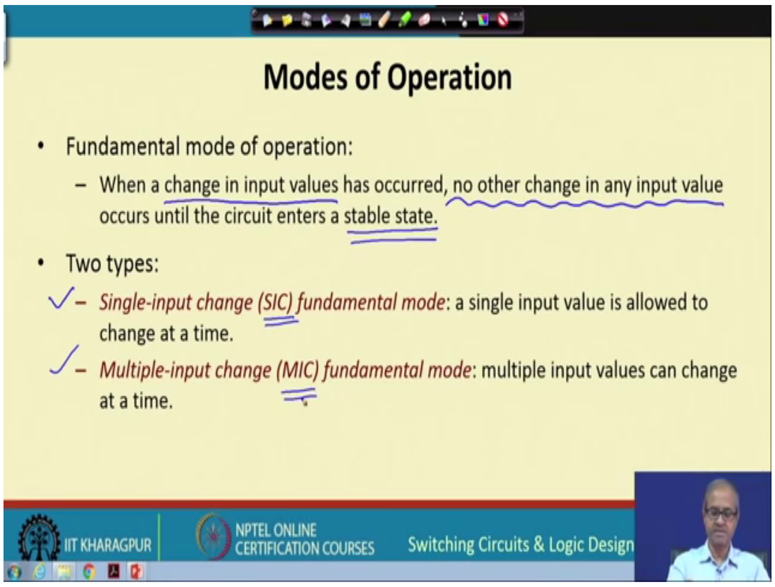
- Stable state:
  - For a given input state the circuit is said to be in a *stable state* if and only if $y_i = Y_i$ for $i = 1, 2, \ldots, k$.
    - In response to a change in the input state, the combinational logic produces a new set of values for the excitation variables, entering an *unstable state*.
    - When the secondary variables assume their new values (when y's become equal to the corresponding Y's), the circuit enters its *next stable state*.
    - Thus, a transition from one stable state to another occurs only in response to a change in the input state.

So, let us go into the definition. We define something called stable state. For some given input combinations, we say that this circuit in a is in a stable state when this capital Y's are all copied to the small y's for all the secondary state variables. Because you see the delay elements, each of the delay elements the input was capital Y i, the output was small y i..

So, it says that when all these outputs that you are computing based on a given input, they have been moved to y i, you call that the circuit is now in a stable state. But temporarily when there is a change in the input, the excitation variables, excitation variable means this capital of Yi's. They change, but those changes have not yet been reflected in small y i because there is a delay here that we refer to as an unstable state because the circuit may go through an unstable state before stabilizing..

And the next point is, when this capital Yi's are copied into y i we say that we have reached the next stable state. So, basically transitions take place from one stable state to another whenever the input changes. In a synchronous sequential circuit, state changes occur in synchronism with the clock. But in an asynchronous circuit, whenever the inputs change, the state changes will occur. For example, you think of that one bit latch. So, whenever the input I am applying 0 1 or 1 0 who's only then the output of the latch will change. So, when you apply 1 1 that is the stable state; no change, right.

(Refer Slide Time: 11:45)



Modes of Operation

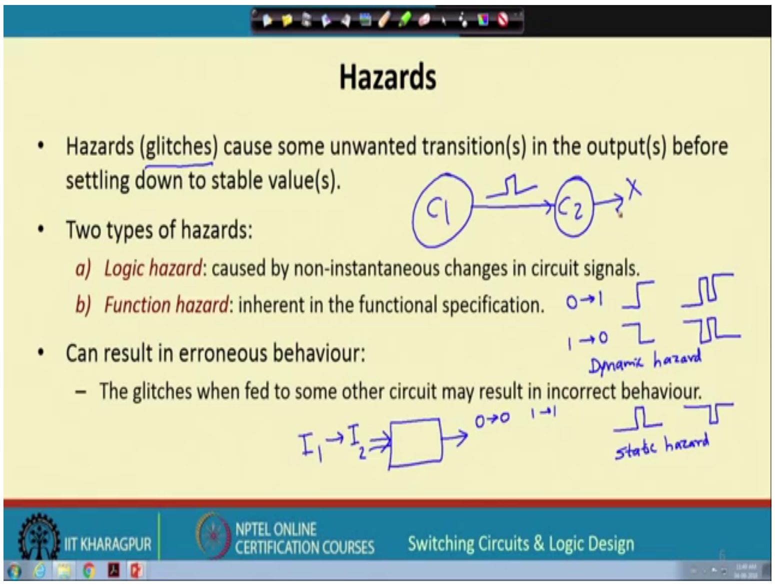
- Fundamental mode of operation:
  - When a change in input values has occurred, no other change in any input value occurs until the circuit enters a stable state.
- Two types:
  - *Single-input change (SIC) fundamental mode*: a single input value is allowed to change at a time.
  - *Multiple-input change (MIC) fundamental mode*: multiple input values can change at a time.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

Such an asynchronous circuit has something called a fundamental mode of operation that you refer to about. So, in the fundamental mode of operation what we assume is that when we have made some changes in the input variable, we must be very careful. No other changes in the input values are permissible before the circuit enters a stable state. Because here there is no concept of clock everything happens in synchronism with the delays or depending on the delays of the gates..

So, whenever there is a change in the input, you should allow for the circuit to stabilize before you apply the next input. So, you should not apply or change the inputs before the circuits have stabilized, right. And there can be 2 different kinds of changes that you can talk about, that one input changing at a time. This is called Single Input Change or SIC in short or in general you can have Multiple Input Change or MIC. These are the 2 fundamental modes of operation we talked about.

(Refer Slide Time: 13:07)



Hazards

- Hazards (glitches) cause some unwanted transition(s) in the output(s) before settling down to stable value(s).
- Two types of hazards:
  - *a) Logic hazard*: caused by non-instantaneous changes in circuit signals.
  - *b) Function hazard*: inherent in the functional specification.
- Can result in erroneous behaviour:
  - The glitches when fed to some other circuit may result in incorrect behaviour.

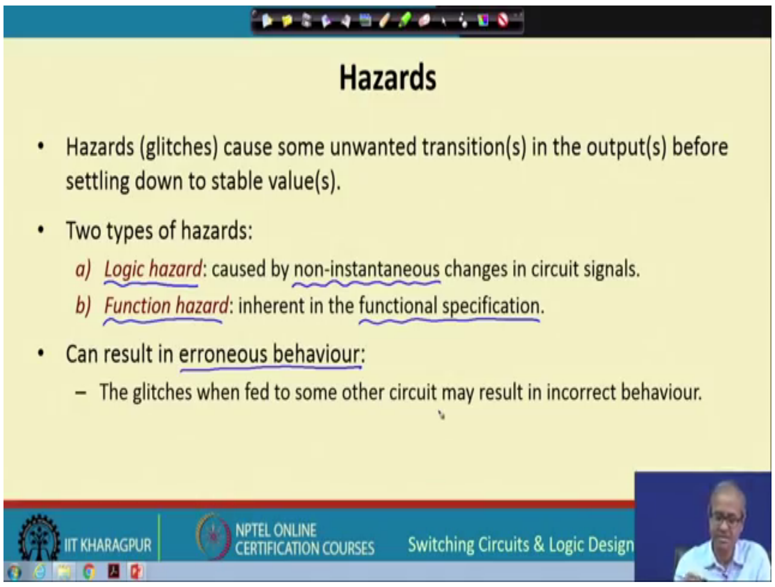
Now in asynchronous circuit, the most important thing that we talked about that you need to worry about is something called hazard. Say, hazards are also sometimes called glitches. You say let me just explain with the help of a general things. Suppose I have a circuit, I have a circuit, there are some inputs. So, I apply some input I 1, I change it to some other input I 2. Suppose, there is a output. For I 1, the output is supposed to be 0, for I 2 also the output is supposed to be 0. So, it is supposed to be a 0 to 0 transition means no change.

But actually because of the delays of the gates what you may see that the output is temporarily going into 1, this is a temporary state before stabilizing to 0. Or if it is 1 to , it was 1 remains 1, it may be temporarily going to 0 like this. This is what is called glitch and this whatever I mentioned here this is sometimes also referred to as static hazard..

Let us take another example, where the input was 0, let us suppose it is going to 1; for I 1 it is 0, I 2 it is 1. But instead of making a clean transition like this, it may so happen because of the delays that it makes a temporary transitions like this before going to 1. Similarly, for 1 to 0 instead of making a clean transition, it makes transition like this. These are also glitches; this is called dynamic hazard.

So, in sequential in this kind of asynchronous circuit design, we need to worry about these hazards because the reason is that if the output of a circuit is going to the input of another circuit. Suppose, I have a circuit here C 1, it is going to the input of another circuit C 2. Now, if there is a glitch like this here because of this glitch this C 2 can start malfunctioning and the output may be wrong, ok. So, it is always good to avoid such hazards when you are designing asynchronous circuits. There are 2 types of hazards we normally talk about.

(Refer Slide Time: 16:03)



Hazards

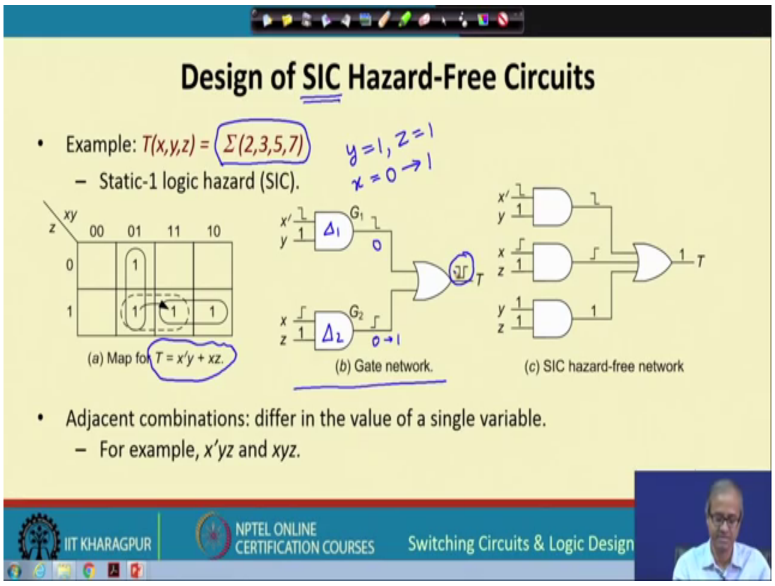
- Hazards (glitches) cause some unwanted transition(s) in the output(s) before settling down to stable value(s).
- Two types of hazards:
  - a) *Logic hazard*: caused by non-instantaneous changes in circuit signals.
  - b) *Function hazard*: inherent in the functional specification.
- Can result in erroneous behaviour:
  - The glitches when fed to some other circuit may result in incorrect behaviour.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

So, one is the example I talk I gave. This is called logic hazard. They are caused by changes in the circuit signal. Because of the delays in the gates, the changes will take some time they are non-instantaneous.

And in general there can be function hazards which do not that much depend on the delay of the gates, but they depend on the functional specification themselves. But hazards in general as I said can result in erroneous behavior; the glitches may be fed to some other circuit that can lead to errors, right, fine.

(Refer Slide Time: 16:45)



Design of SIC Hazard-Free Circuits

- Example: $T(x,y,z) = \Sigma(2,3,5,7)$
  - Static-1 logic hazard (SIC).

(a) Map for $T = x'y + xz$.

(b) Gate network.

(c) SIC hazard-free network

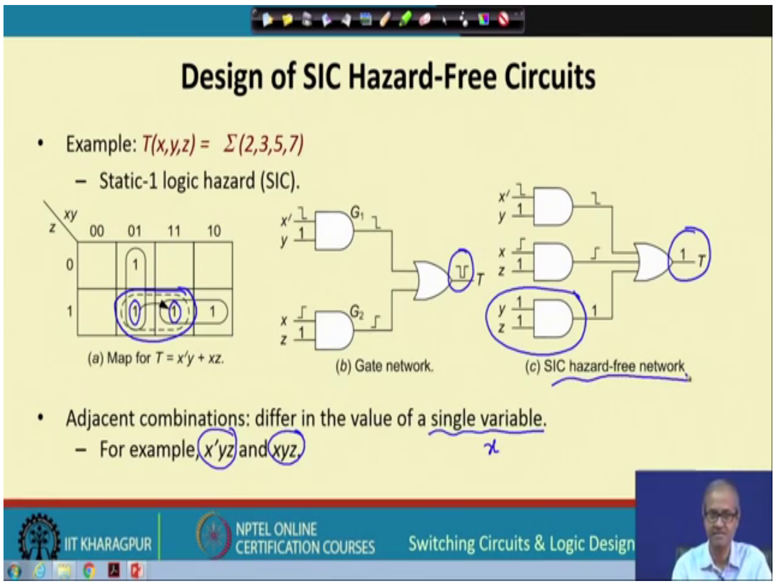
- Adjacent combinations: differ in the value of a single variable.
  - For example, $x'yz$ and xyz.

Let us now take an example. Let us look into the design of single input change SIC hazard free circuit. Let us take an example, this function T in sum up products sum of min terms representation; 2, 3, 5, 7. Now in a Karnaugh map, the 4 min terms are shown here; 2, 3, 5 and 7. No, these are not exactly 2, 3, 5, 7; anyway 2 is 010, this is 2, this is 3, 5 is this, ok. This is 5, this is a 7 ok, alright.

Now, suppose I minimize this function like this, the 2 cubes are shown by solid lines and the minimized form of the expression is this. So, I can implement this like this, the first get implements x bar y. I feed x bar in the first input y here. The second input I apply x and z. There is OR. Now let us suppose, let us take a scenario, that I have applied y equal to 1 z equal to 1. This is fixed and x was 0 and I make a single input change, I make x equal to 1. So, when I make x equal to 0 to 1; that means, x changes from 0 to 1 and x bar changes from1 to 0. Now, you see the delay of these 2 gates can be different, right.

Suppose, delay of gate 2 is more than G 1, so, this transition occurs first. G 1 output becomes 0 first, but you see this is at AND gate this is delayed. So, the output will become 1 after some delay. So, there will be a small time where this G 2 will be 1 then it will become 0. So, there is will be a small period when both G 1 and G 2 will be 0 and 0. That will result in this static hazard as I have shown here. This will happen if the delay of G 2 is greater than delay of G 1. I suggest you draw the timing diagram and verify that a glitch like this actually happens right.

(Refer Slide Time: 20:01)



Design of SIC Hazard-Free Circuits

- Example: *T(x,y,z) = Σ(2,3,5,7)*
  - Static-1 logic hazard (SIC).

(a) Map for $T = x'y + xz$. (b) Gate network. (c) SIC hazard-free network

- Adjacent combinations: differ in the value of a single variable.
  - For example, $x'yz$ and xyz.

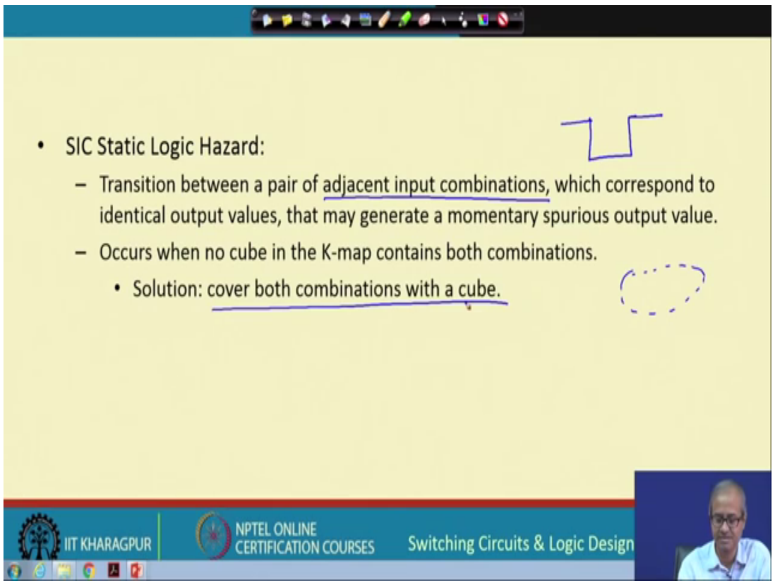
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES Switching Circuits & Logic Design

Now, coming back to this Karnaugh map, this glitch happens because we are changing a single variable; here it is x and there are 2 adjacent combinations we can identify x bar y z and x y z. You see looking at the cube, you look at these 2 1, you look at this. You look at this these 2 correspond to x bar y z and x y z. In the 2 cubes there is one adjacent cube. And whenever this X changes, temporarily you make a change this arrow is shown this temporary transition is taking place..

Now, in order to avoid this hazard, this kind of a glitch, what you do? You look at this kind of adjacent cubes and you see that well, there are 2 adjacent cubes, you also include this cube in your implementation. Well, this may not be minimal, you are adding one more gate alright, but you are avoiding hazard.

So, if you add this get this means what; y z. So, you add this third gate here. And what you get? You can check that here even if there are delays in these gates, the output will be a perfect one; no glitch, this is a hazard free circuit ok.

(Refer Slide Time: 21:35)

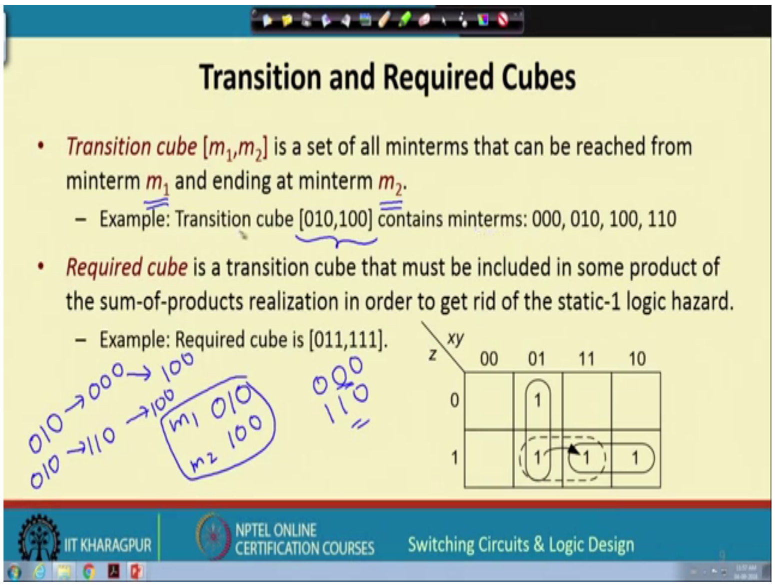


- SIC Static Logic Hazard:
  - Transition between a pair of adjacent input combinations, which correspond to identical output values, that may generate a momentary spurious output value.
  - Occurs when no cube in the K-map contains both combinations.
    - Solution: cover both combinations with a cube.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

Talking about static logic hazard, as I said in the example, transition between a pair of adjacent input combinations which correspond to identical output values in the example it was 1, it remained 1. It may temporarily generate a spurious output value glitch. And this occurs as I just now said with respect to Karnaugh map. When no cube in the Karnaugh map contains both combination which was shown in the dotted way. Solution is to cover both combination which a cube which we actually did, right. And in the final solution, there is no hazard.

(Refer Slide Time: 22:23)



Transition and Required Cubes

- *Transition cube* $[m_1,m_2]$ is a set of all minterms that can be reached from minterm m_1 and ending at minterm m_2.
  - Example: Transition cube [010,100] contains minterms: 000, 010, 100, 110
- *Required cube* is a transition cube that must be included in some product of the sum-of-products realization in order to get rid of the static-1 logic hazard.
  - Example: Required cube is [011,111].

z \ xy	00	01	11	10
0		1		
1		1	1	1

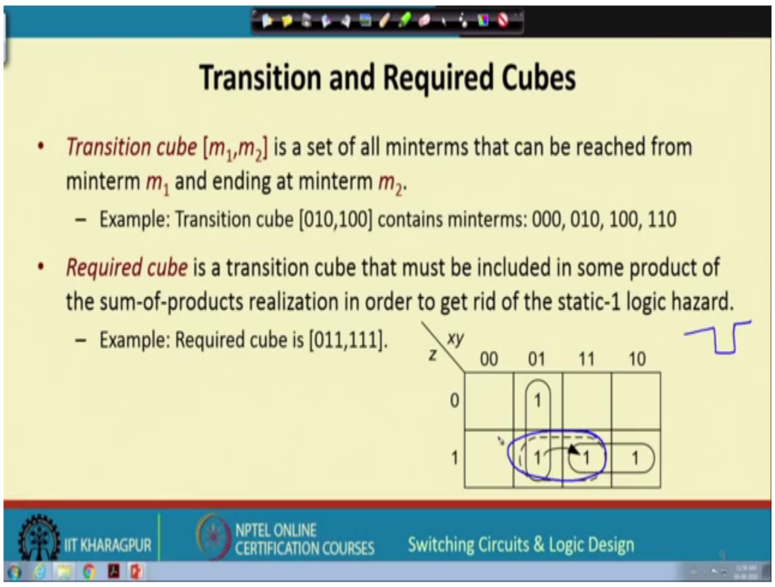
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, let us continue with the discussion. There is something called transition cube and required cube. Let us see transition cube is defined with respect to 2 min terms m 1 and m 2. This refers to set of all min terms that can be reached from m 1 and up to m 2. Let us take an example..

Suppose I have 010 and 100, this is m 1 and m 2 is 100. So, I want to go from 010 to 100, but I can change only one inputs at a time. So, there can be min terms like that in addition to 00 and 100, there can be 010, 010 is there, there can be 000 and there can be 110. You see to change 010 to 100 I can have one possibilities, 010 to I can go to 000 first, ok. Then I can go to 100 or I can start with 010. I can go to 110 first, then I can go to 100 single input transitions.

So, I try to find out all such possible means additional min terms that will be part of the transition cube. And required cube is a special kind of a transition cube that must be included in some product term to get rid of static logic hazard. Static 1 means it was 1, this is static 1.

(Refer Slide Time: 24:15)



Transition and Required Cubes

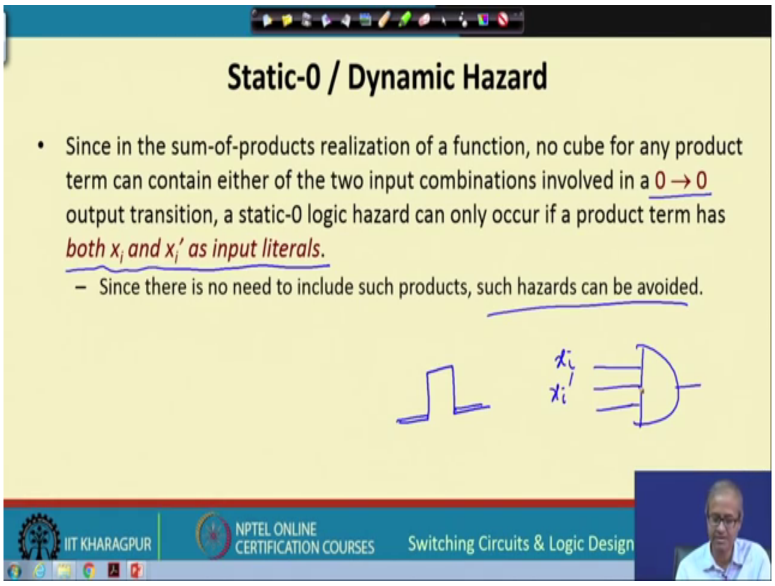
- *Transition cube* $[m_1,m_2]$ is a set of all minterms that can be reached from minterm m_1 and ending at minterm m_2.
  - Example: Transition cube [010,100] contains minterms: 000, 010, 100, 110
- *Required cube* is a transition cube that must be included in some product of the sum-of-products realization in order to get rid of the static-1 logic hazard.
  - Example: Required cube is [011,111].

z \ xy	00	01	11	10
0		1		
1		1	1	1

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

Now here the required cube like this one I have shown, this is 011 to 101, this is 011, this is 111. This is just an example of a required cube, ok.

(Refer Slide Time: 24:37)



Static-0 / Dynamic Hazard

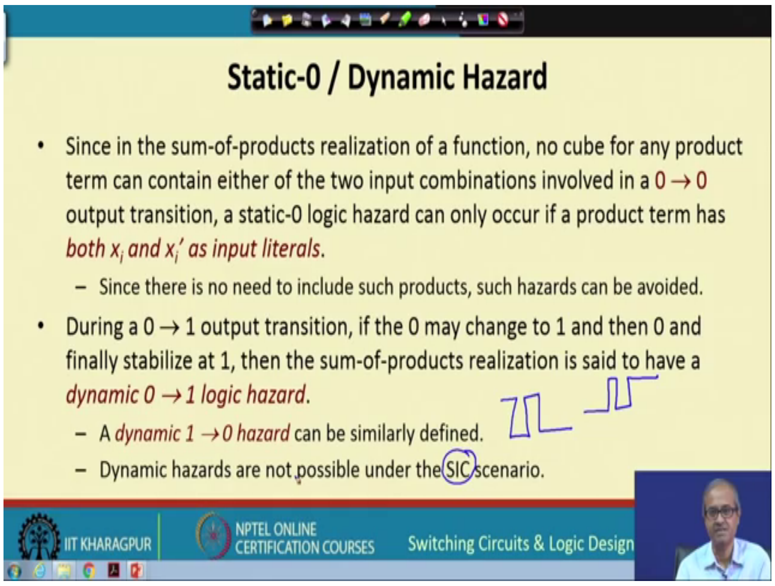
- Since in the sum-of-products realization of a function, no cube for any product term can contain either of the two input combinations involved in a $0 \rightarrow 0$ output transition, a static-0 logic hazard can only occur if a product term has *both x_i and x_i' as input literals.*
  - Since there is no need to include such products, such hazards can be avoided.

x_i
x_i'
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

And the last thing static 0 or dynamic hazard both static 0 means, you see it was 0, it remains 0. Now, you see in a sum of products realization of a function we do not keep track of the min terms for if the function output is 0. So, there are no gates which are generated for this. That is why for static 0 we need not have to worry about it. Such hazards can be avoided right. There is no cube for any product term that will lead to this 0 0, because 0 we do not have in any of the cubes, only 1's..

Only in a very with hypothetical case, if we have a gate where I apply x i as one of the input and also x i bar as one of the inputs, there is no reason why I should apply like this. Only in such cases this kind of an hazard can be there, but otherwise it is not there. The only situation is both x i and x i bar can be the input literals of one of the cubes. But normally we do not do that, there is no reason to do that; that is why such hazards can be avoided.

(Refer Slide Time: 25:55)



Static-0 / Dynamic Hazard

- Since in the sum-of-products realization of a function, no cube for any product term can contain either of the two input combinations involved in a $0 \rightarrow 0$ output transition, a static-0 logic hazard can only occur if a product term has *both x_i and x_i' as input literals.*
  - Since there is no need to include such products, such hazards can be avoided.
- During a $0 \rightarrow 1$ output transition, if the 0 may change to 1 and then 0 and finally stabilize at 1, then the sum-of-products realization is said to have a *dynamic $0 \rightarrow 1$ logic hazard.*
  - A *dynamic $1 \rightarrow 0$ hazard* can be similarly defined.
  - Dynamic hazards are not possible under the SIC scenario.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

And during a 0 to 1 transition, 0 to 1 I said that there can be dynamic hazard there can be a temporary change. Like this, this is called a dynamic 0 to 1 logic hazard and dynamic 1 to 1 also I mentioned earlier, this kind of a transition. And for single input change scenario we can never have dynamic hazards. This is the point to note. Only when one input changes at a time, you can only have single input static hazards..

So, with this we come to the end of this lecture. We have tried to give you a very basic overview about the concept of hazards in asynchronous circuits and some of the simpler kind of hazards, how they can be avoided. So, we shall be continuing our discussion in the next lecture.

Thank you.