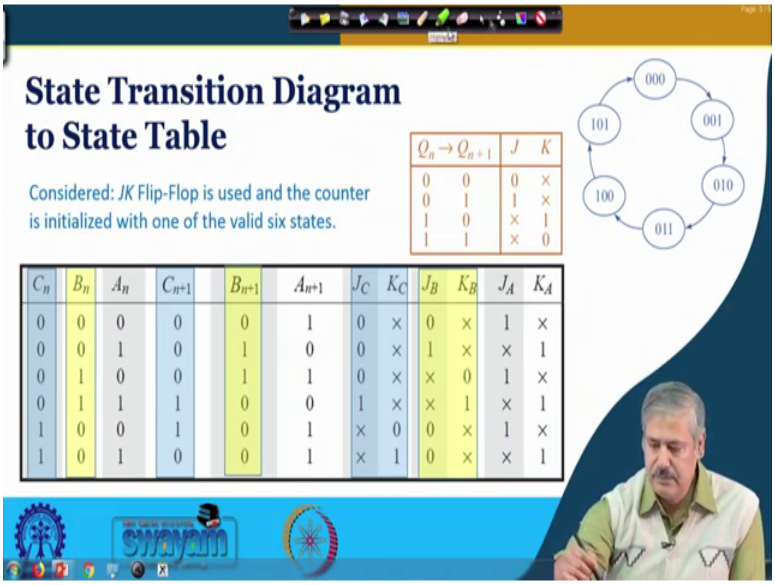
Digital and Electronic Circuits
Prof. Goutam Saha
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture – 45
Counter Design as a Synthesis Problem and Few Other Uses of Counter

Hello everybody, we are in the last lecture of this counter topic. So, in earlier lectures we have seen how to design Counter with different modular numbers using asynchronous reset and a preset. And in that we have seen that there is a small glitch that is possible because of the reset precedes going to the next state and asynchronous reset ensures all zero or a pre set value appearing in the same clock cycle.

So, in this particular class, we shall see how synchronous counter with different modular number can be designed and that design can be considered as a synthesis sequential logic circuit design problem. And we can we shall see that those kind of the glitch we will not occur, it does not required to go to next state. Straight away the next valid state in the sequence as per the modular number if n unique states are there those n unique states will be appearing one after another. And we shall also look at some specific application other than the ones that you have seen before ok.

(Refer Slide Time: 01:37)



State Transition Diagram to State Table

Considered: *JK* Flip-Flop is used and the counter is initialized with one of the valid six states.

Q_n	$\rightarrow Q_{n+1}$	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

C_n	B_n	A_n	C_{n+1}	B_{n+1}	A_{n+1}	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	0	1	0	0	×	1	×	×	1
0	1	0	0	1	1	0	×	×	0	1	×
0	1	1	1	0	0	1	×	×	1	×	1
1	0	0	1	0	1	×	0	0	×	1	×
1	0	1	0	0	1	×	1	0	×	×	1

So, we start with an example where we are taking mod 6 counter ok. So, in the mod 6 counter, we are having six unique states and we are considering a up counter for this discussion. And this counter has got the six states 000 001 01 that you can see 010 011 100 and 101 after that again it goes to 000. And we considered of all the flip flops J K to be the one by which we would like to design it right.

So, for J K flip flops in series see in this problem for design, we shall refer to its excitation table. So, this is the extension table right. So, if the flip flop current state is 0 and next state is also 0 as per this transition diagram, then at the input what should be present 0 cross 011 cross should be present 10 cross 1 should be present and 1 1 cross 0 should be present cross 0 means 1 of 00 or 10 either way it is fine ok.

One of these two combination either 00 or 10 that will ensure 11 transition 1 to 1 transition for J K flip flop ok. So, this we have already know from our earlier discussion on flip flop. Then we shall follow the standard synthesis process. So, once we have this state transition diagram, we shall go to this state table and we shall use the excitation table for getting different input combinations for different states.

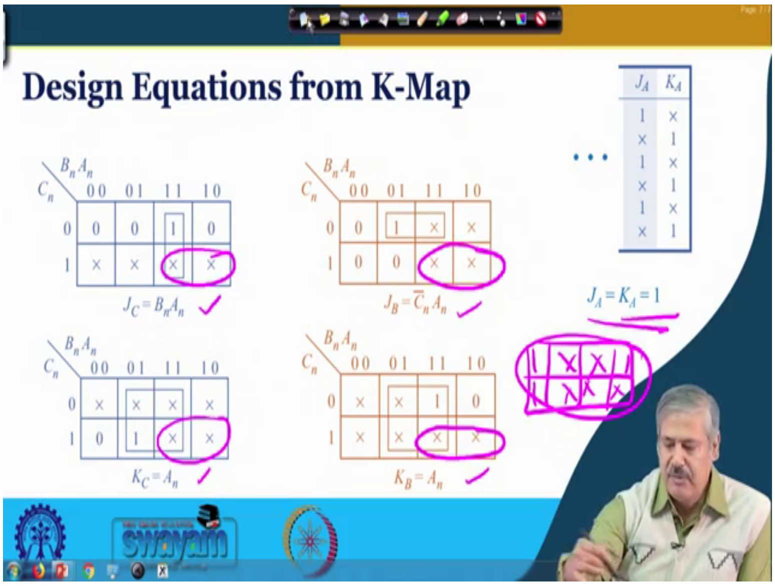
So, 0 0 0 is the initial state over here like right. So, for that the next state is 0 0 1. So, what will be the corresponding input for C n changing to C n plus 1 0 cross from here; B n is also changing to B n plus 1 is that is 0 to 0. So, 0 cross and A n changing from 0 to 1. So, it is 0 to 1 transition so, 1 cross is it clear. We are just recollecting and what we had done before in simple synthesis example in earlier classes, so, this is the first row. Similarly the second row after 0 0 1, it has to go to 0 1 0 ok.

So, this is 0 1 0 and corresponding transition 0 to 0 0 cross 0 to 1 1 cross and 1 to 0 1 to 0 you can see over here 1 to 0 cross 1; is it clear second row is also clear ok. So, third low third row 0 1 0 to 0 11. So, 0 0 0 0 cross 1 to 1 1 to 1, this line cross 0 and then 0 to 1 1 cross 0 to 1 1 cross ok. So, with this if you proceed, we will come to 1 0 1 right.

Now, after 1 0 1 in earlier design that was a module 8 counters at standard counter and it was going to 11 0 and we are doing a asynchronous reset. So, that 11 0 and 0 0 0 are occurring in the same clock cycle for a very small duration 11 0 was there. Now in this design after 1 0 1, we are going straight away to; straight a way to 0 0 0. Here it should be 0 right it should be 0 right. So, 1 to 0; it should be cross 1 so, that is your cross 1 0 to 0. So, this is your 0 cross and 1 to 0 1 to 0. So, this is your cross 1 is it clear, so, this was a type error is it fine.

1 to 0 right that is your I am doing it once more again. So, this will be; this will be 0 1 to 0, this is your transition cross 1. So, that is your cross 1 0 to 0 0 to 0. This is your 0 cross and 1 to 0 1 to 0 again cross 1. So, this is cross 1 is it clear. So, from 1 0, you are directly going to 0 0 0 right. So, this is the thing that you are doing in designing a modular number modular 6 counter using standard synthesis design problem ok. The way we solve it see sequential logic circuit design problem.

(Refer Slide Time: 07:01)



Design Equations from K-Map

$C_n \backslash B_nA_n$	00	01	11	10
0	0	0	1	0
1	×	×	×	×

$J_C = B_nA_n$ ✓

$C_n \backslash B_nA_n$	00	01	11	10
0	0	1	×	×
1	0	0	×	×

$J_B = \overline{C}_nA_n$ ✓

$C_n \backslash B_nA_n$	00	01	11	10
0	×	×	×	×
1	0	1	×	×

$K_C = A_n$ ✓

$C_n \backslash B_nA_n$	00	01	11	10
0	×	×	1	0
1	×	×	×	×

$K_B = A_n$ ✓

J_A	K_A
1	×
×	1
1	×
×	1
1	×
×	1

$J_A = K_A = 1$

1	X	X	1
1	X	X	X

Next what is the next talk? Next step so, next step is from those inputs that state table, we are writing the equation corresponding equation for individual inputs. So, J C K C J B and K B right and we have already noted in the previous case as J and K; if you look at the previous one ok.

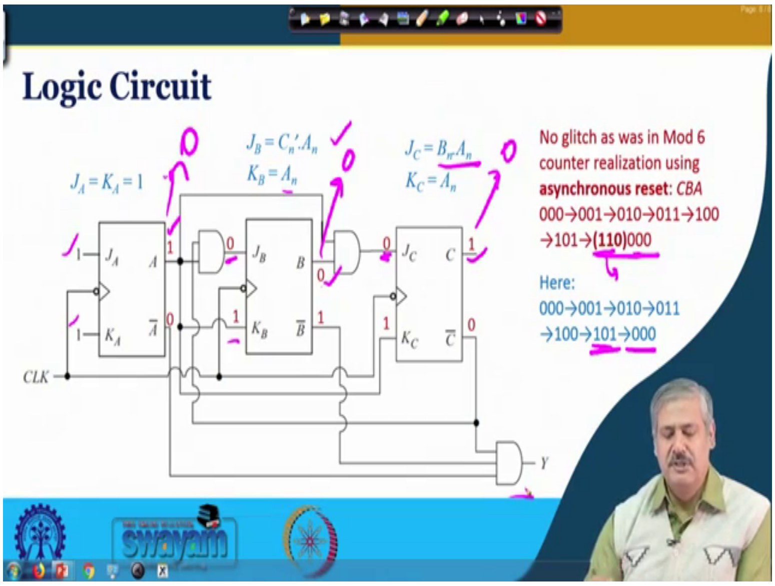
So, 1 cross 11 cross 11 and so on. So, that is the thing that is being represented here and rest of the things are taken from here to the next this corona map; this simplification. So, all of them are this 1 or cross so, we can consider all of them as 1, then it becomes the simplest possible. So, that way you can consider each 1 of them as 1 ok.

So, in the corresponding table, you can understand that this is 1 cross 1 cross then 1 cross and this two are 11 0 and 11 1 we can consider them as cross. So, all of them can be put together. So, this is your 1 fine. For other cases, we are just substituting the values taking the values from those columns and remember this 11 0 and 111 those were not there. Those last two possible with three flip flops eight possible states are there. So, 110 and

111 we did not considered right. It will not it is not a valid state is not it. So, we considered for those two cases cross. So, this is cross cross; that means, it does not matter because it is supposed it is considered that it will never go to those states.

So, it is initialized it one of the valid state this counter is initialised with one of the valid state. Later on we shall see if we it is not the case then what will happen so that we shall take up lit later so over here ok. So, accordingly we do not have any compulsion of if there was a 1 or something 0 considering treating it accordingly ok. Now we can consider them as do not care and depending on that we will get a minimise circuit ok. So, these are the corresponding equation right and once this equations are obtained from there, we can get the circuit right; is it clear.

(Refer Slide Time: 09:30)



Logic Circuit

$J_A = K_A = 1$

$J_B = C_n'.A_n$

$K_B = A_n$

$J_C = B_n.A_n$

$K_C = A_n$

No glitch as was in Mod 6 counter realization using **asynchronous reset**: *CBA*

000→001→010→011→100 →101→(110)000

Here:

000→001→010→011 →100→101→000

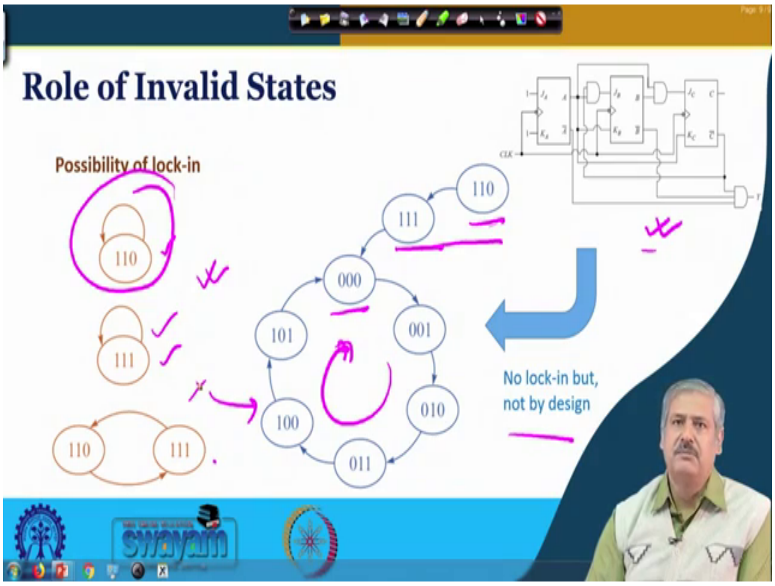
So, this is the standard sequential synthesis logic synthesis problem. We look at how this particular circuit how this particular circuit behaves when 1 0 1 comes here in over here and then with the clock trigger how it goes to 0 0 0 without any glitch or so, alright. So, 1 0 1 so this is 1 right and this is at 0 and this is at 1 right.

And when this is 1 and this is 0 and this is 1 so, what are the corresponding inputs of this flip flop? So, J K is always 1 that you have noted is not it and this J B is we have found from the previous corona map is simplification is C n bar A n. So, accordingly the circuit has been drawn ok. So, C n bar because it is 0 so, together it is 0 ok. And what is A n? A n is 1 1. So, this is 0 1 fine. What about J C K C?

So, J C is B n A n right so, A n is 1, but B n is 0. So, this input is 0 right. And what about K C? K C is A n A n is coming over this is 1 right. So, for a flip flop you have got 11 for B flip flop, you have got 0 1 at the input and for C flip flop, you have got 0 1 at the input right. Now when the clock trigger comes what happen? Now when the clock trigger comes, you will have 11 it toggles. So, it will become 0 from 1 to 0 0 1 for this J K flip flop 0 1, if it is the input then it will be 0 only right and here also it is 0.

So, it will be 0 only. So, with the clock edge, it all of them go to 0 0 0 unlike the asynchronous reset based modular 6 counter where we have got 11 0 and at that times with this value it was getting reset and the output was obtained. I mean the next state was obtained as 0 0 0 for longer duration of the time. Is it fine right? And this is the decoding of the counter. So, this part is a understood right.

(Refer Slide Time: 12:12)



Role of Invalid States

Possibility of lock-in

No lock-in but, not by design

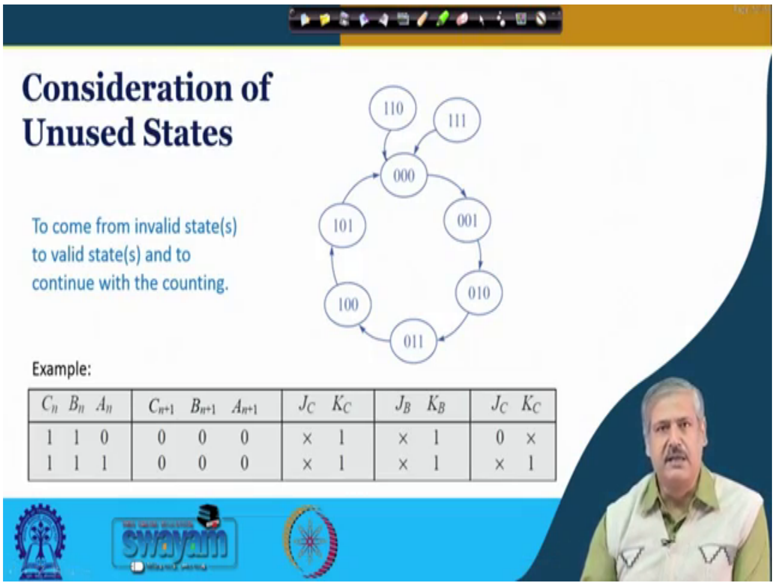
Now we look into the case for the two states 11 0 and 11 1 which we did not actually consider in our design process right. So, in this particular case, this is the circuit diagram we are having. Suppose it is you know initialised with or because of some noise or some charge it goes to one of those two states. So, if it goes to 11 1 11 1 and then, you can consider the you know calculate the output input for each of the flip flops right and with a next clock trigger, you can see that it is coming to 0 0 0.

And then we clocking, it will keep circulating like this as per this state transition diagram and mod 6 counter will be there after that. And had it been initialised or it had gone to the other invalid state which is 11 0 with a clock trigger. It would have been come to 111 and then to 0 0 0 ok.

So, this is what happens for this particular circuit, but this is happening just like that. But there could be have been another I mean there could have been other possible cases where if it is 110 with clocking right it remains at 11 0 only right. With 111 it is remaining at 1111 or 110. It is going to 11 or with from 11 1 it is coming back to 11 0. That means, it is getting trapped into the into the invalid states only and it is never coming back to coming to one of the valid states ok.

That also could have been a possibility because we got this way that it is not getting trapped ok, but it was not by design; not by design right. Just saw see that it has it has happened in that manner. But in another design there remains a possibility that it could be locked in one of these cases or if there is a modular 5 counter there could be three invalid states it could be locked in one of those 3 states also or you know keep circulating there. So, this is something which we should think of avoiding in a counter design right.

(Refer Slide Time: 14:39)



Consideration of Unused States

To come from invalid state(s) to valid state(s) and to continue with the counting.

Example:

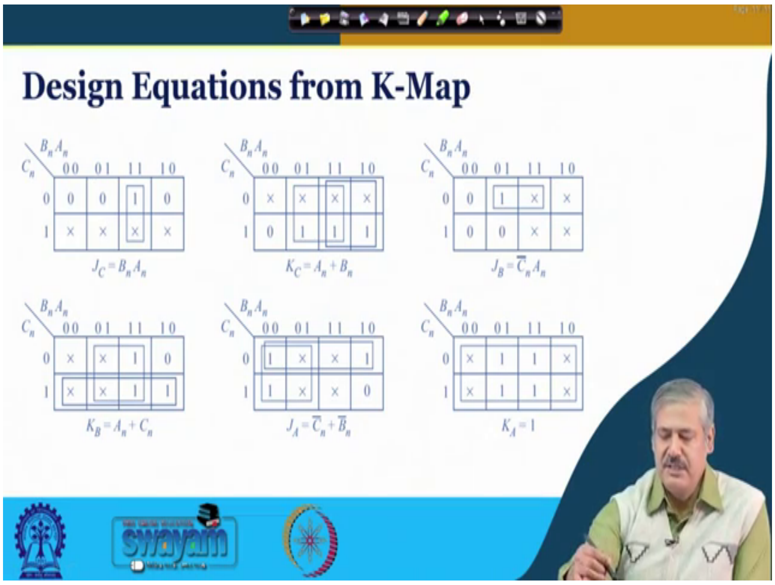
C_n	B_n	A_n	C_{n+1}	B_{n+1}	A_{n+1}	J_C	K_C	J_B	K_B	J_C	K_C
1	1	0	0	0	0	×	1	×	1	0	×
1	1	1	0	0	0	×	1	×	1	×	1

So, for that what you need to do? We need to explicitly considered then in the design process right. And how you can consider? So, there could be multiple ways. So, in this example what we see is that for one of for any of these invalid states if it goes 110 or 1 111. It the next clock trigger it comes to one of the valid state which is the 0 0 0.

After that it will continue to count you know six the way, it was designed I mean it is already designed alright. So, for this, then this two in used state which we did not considered in the earlier design and there was a do not care option in the corona map based simplification of the design equation. So, they now get into picture right.

So, 110 you are getting 00 0 and corresponding C B a flip flop inputs are like this and 111 it is also 000 and these are the corresponding inputs from the J K flip flop excitation table right. Now other than those six, this two also are getting considered right. So, to explicitly get around or avoid that getting trapped into the what is that called unused states right.

(Refer Slide Time: 16:14)



Design Equations from K-Map

$C_n \backslash B_nA_n$	00	01	11	10
0	0	0	1	0
1	×	×	×	×

$J_C = B_nA_n$

$C_n \backslash B_nA_n$	00	01	11	10
0	×	×	×	×
1	0	1	1	1

$K_C = A_n + B_n$

$C_n \backslash B_nA_n$	00	01	11	10
0	0	1	×	×
1	0	0	×	×

$J_B = \overline{C}_nA_n$

$C_n \backslash B_nA_n$	00	01	11	10
0	×	×	1	0
1	×	×	1	1

$K_B = A_n + C_n$

$C_n \backslash B_nA_n$	00	01	11	10
0	1	×	×	1
1	1	×	×	0

$J_A = \overline{C}_n + \overline{B}_n$

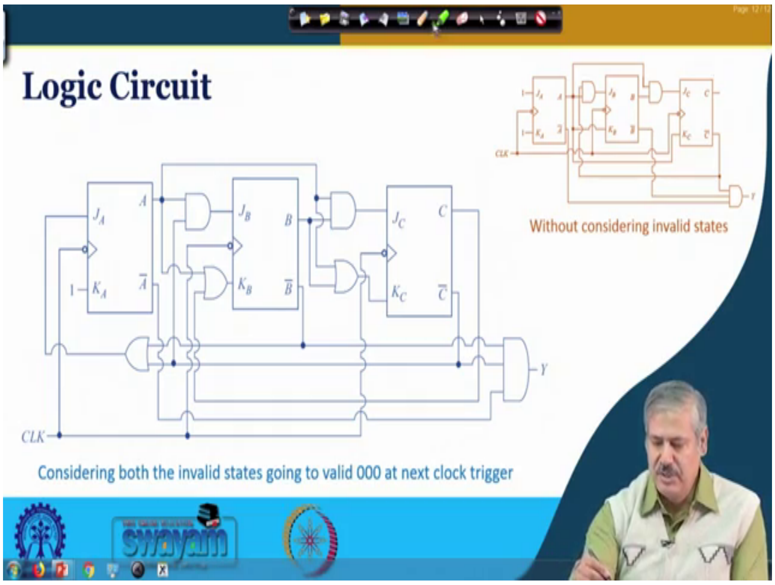
$C_n \backslash B_nA_n$	00	01	11	10
0	×	1	1	×
1	×	1	1	×

$K_A = 1$

So, now the design equation becomes a little bit more complex. In the sense now this earlier this two cases; it was always do not care. So, we had better fix flexibility in deciding the equations. So, corresponding equations have now become like this all right. So, you can see that earlier this equations right for example, J A; J A was all 1 J was 1 because this was a do not care right. Now there is a 0 over here ok.

You cannot treat this as a 1. So, accordingly it has become C n bar plus B n. So, we are sacrificing some of these design flexibility and how that becoming little bit more complex, but the benefit is that this there is no lock kind of situation ok, so, no getting trapped into the unused state.

(Refer Slide Time: 17:25)



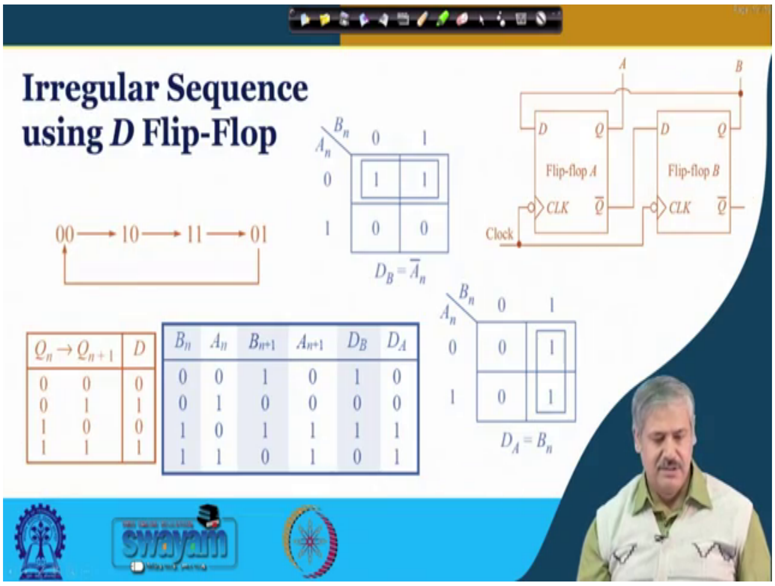
Logic Circuit

Without considering invalid states

Considering both the invalid states going to valid 000 at next clock trigger

So, that is what you can see that this is the this was the earlier circuit and now with those design equations for inputs, this is the modified circuit ok. So, at the invalid states going to valid 0 0 0 at next clock trigger. And this is this is what is expected also is not it right fine.

(Refer Slide Time: 17:55)



Irregular Sequence using D Flip-Flop

$A_n \backslash B_n$	0	1
0	1	1
1	0	0

$D_B = \overline{A}_n$

$A_n \backslash B_n$	0	1
0	0	1
1	0	1

$D_A = B_n$

Q_n →	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

B_n	A_n	B_{n+1}	A_{n+1}	D_B	D_A
0	0	1	0	1	0
0	1	0	0	0	0
1	0	1	1	1	1
1	1	0	1	0	1

So, now, let us look at another aspect of the design. So, you had been discussing this up counter. Again similarly down counter is possible and it is also possible that we can have these counting states. They need not be regular in the in the sense that it is increasing model you know always the number is increasing one after another by a value of 1 or getting decremented by 1 right. So, it is not necessary that it had need to be like that. We just need a as many number of unique states ok. And that can also be considered in the design process right.

So, here is one such you know modular for counter where the numbers are not 0 0 0 11 0 11, again 0 0 it is not like that what is it? 0 0 after that 1 0 then 11 0 1 and then 0 0. One good thing about it what you can see here that only 1 it is changing which could be useful in some consideration in some cases right.

So, it in to counting states one after another one bit is changing its value all right, but otherwise the significance the corresponding decimal equivalent 0 2 1 1 again 0. So, in that sense it is irregular, it is not 0 1 2 3 0 or the down count the way we have seen that 3 2 1 0 and again 3 is not like that right. so, that sense.

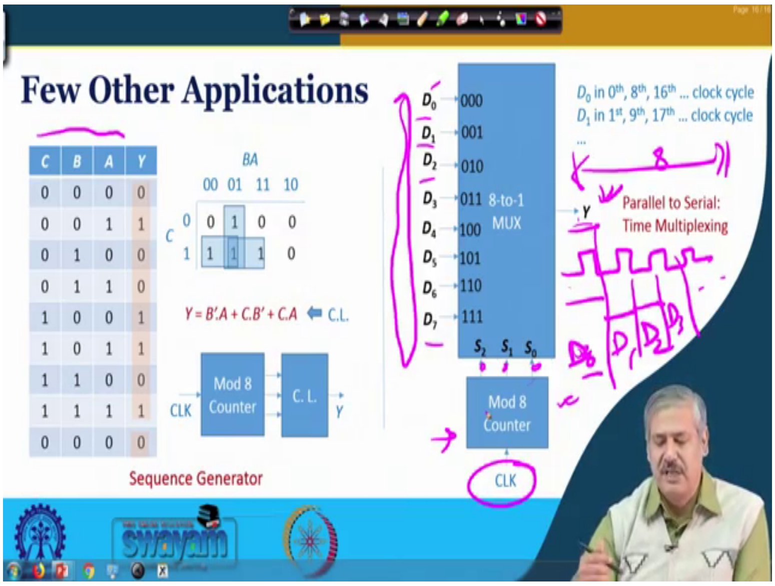
And what else is there in this particular example? So, we had used J K flip flop right earlier, now we are considering d flip flop. So, with d flip flop also you can design the counter. So, let us see how. Since it is d flip flop so, d flip flop excitation table will be put to you. So, this is 0 to 0, then 0 should be input 0 1 1 should be here input 1 0 0 should be at the input and 11 1 should be at the input is it fine. So, this is the way the excitation table of d flip flop we have seen in earlier classes related to flip flop alright.

So, now 0 0 to 1 0 0 0 to 1 0, then 0 1; if it is 0 1, then it goes to 0 0; if it is 1 0, it goes to 11 and if it is 11, it goes to 0 1. So, you need to be take a note of this thing that it is not coming one after another 0 0 to 0 1 and it is just incrementing in that manner.

So, you need to be careful about these aspect in the design process not to make mistake when we map it to corona map right. And then D flip flop s is whatever is the next value that is the input ok, so, d 1 0 1 0 so, 1 0 1 0 so, 0 0 1 1 so, 0 0 1 1. So, accordingly you have got your this corona map entry and these are the equations is it fine right. So, with this equations, we can get the circuit mint right and the circuit will we have like a mod 4 counter with counting states 0 0 1 0 1 1 0 1 again 0 0 fine right.

So, these are example with mod 4, it can be done for mod 8, mod 6, mod 4, 5 whatever right and whenever we are using a module number which is not integer for up to with flip flops that can have some invalid states unused states. It is better that we take care of it, but putting them into the design consideration to avoid any kind of locking.

(Refer Slide Time: 22:01)



Few Other Applications

C	B	A	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1
0	0	0	0

C \ BA	00	01	11	10
0	0	1	0	0
1	1	1	1	0

$Y = B'.A + C.B' + C.A$ ⬅ C.L.

Sequence Generator

D_0 in 0th, 8th, 16th ... clock cycle
D_1 in 1st, 9th, 17th ... clock cycle
...

Parallel to Serial:
Time Multiplexing

Now, we had seen you know counters getting used for you know of course, the counting number of pulses and all and then digital clock where the seconds minutes and hours you know this count was getting incremented using different kind of you know counter.

So, mod 6, mod 10 together giving seconds and also 2 minutes and then mod 12 for the hour count all those things we have seen before as use of counter at. This is the standard those are the standard uses right. At there can be few other uses of counter where we can see that counter can be used for in this example sequence generator.

So, how is it. So, the counter sequence generator we had seen before using shift register mostly shift register and linear feedback shift register also alright and that is also an version of a shift resistor. And here in as a counter so, what you do? Say we are looking for a 8 bit long sequence which gets repeated right.

So, we are then having a mod 8 counter with us. So, this can be extended to 16 bit long sequence. So, then will be having a mod 16 counter. So, 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 11 0 11 1 and then again 0 0 0 is coming right. And for each of this count as you know the clocking occurs so, whatever the sequence output sequence generator as a sequence generator the output that you consider in the right hand side.

So, the say we want to generate 0 1 0 0 1 1 0 1 again 0 0 0 1 0 0 11 0 or that is repetitively. So, now, these becomes a comeotrologic you know circuit problem. So, if the count is 0 0 0, the counting state the output will be 0. If the counting state is 0 0 1, output will be 1 0 1 0 output will be 0. And this counter is kept keeping you know keeping keep repeating itself. So, this 0 will again come over here when 0 0 0 comes this one will come again here when the counting state is 0 0 1. So, this is the wave map it right.

So, we have to ensure a logical connection between Y and C B A right. So, C B A is independent variable and Y is the dependent variable. So, for which we are having a trying to get a relationship ok. So, C B A 0 1 and B A is 0 0 0 1 1 1 and 1 0. So, you just put it 0 1 0 0 0 1 0 0 1 1 0 1 1 1 0 1 right and then we write the corona map equation right. So, B prime means C B prime plus C A. So, this is the combinatorial logic that we will get right. So, how then this circuit will be get?

So, clock will be given which is generating the sequence right to a that clock goes to a mod 8 counter right. So, that has got three states C B A right which is now fed to the combinatorial logic within this combinatorial logic box, we have got this equation alright. So, basically these you can take in complemented output of if it is available or inverter, then AND gate and OR gate ok.

Together you will get it so, this is finally, is it fine. right As I said if it is mod 16, then we have to have you know four bit counter any other depending on the counting sequence whether its 12 bit sequence gen sequence we want or to generate repetitively. So, it will be a mod 12 counter right.

That is the way we have qto move and another example with which we end to this class is that what you see over here is a 8 2 1 multiplexer ok. Again this example can be extended to 16 to 1 and cases and this 8 to 1 multiplexer will be having 3 select inputs right and multiplexer does what many to one. So, depending on select inputs 1 of the input get connected to the output is not it; so, that is the idea. So, if you place 0 0 0 D naught will be connected right. So, with 0 0 0 0 over here right D naught and Y is connected the rest of the things are not contributing to the Y.

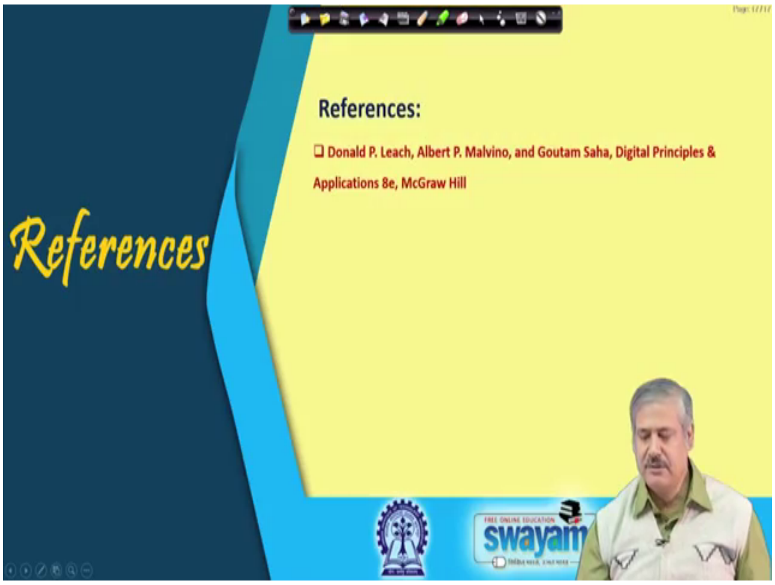
So, that is the way multiplexer works right. Now you can see there this input is selection inputs are fed from the from a mod 8 counter alright which is your C B A; C B A you have seen here and a clock is fed to this counter, so, 0 0 0 0 0 1 and so on and so forth so, this is the clock right. So, when the clock is giving a count say 0 0 0 the first count first clock right so, then d naught whatever is the d naught value that will go over here right. Next is 0 0 1 whatever is the D 1 value that will go here. If D 1 value is 0, it will go as 0 next clock 0 1 0 whatever is the d two value ok.

If it is 0, it will going 0. So, then 7 11 1 d 81 will go here again 0 0 0 d naught will come over here is it clear, is this is understood? So, this parallel set of data. Now is placed serially in a consecutive clock period right by this arrangement. Now if somebody read this one this y in synchronised with this clock and this count value that is there. So, when this value is 0 0 0 in the first clock cycle it will read D naught then he will read D 1.

So, this is D naught read this is D 1 read this is D 2 read and this is D 3 read and so on and so forth and then D 7. So, what was parallel now you an see it is in serial banner you are able to read it and it is time multiplex means it is appearing in different point of time. So, this is was one full 8 clock cycle within which this is getting ride. So, this is in a way you can say it is a parallel to serial conversion is happening and the data is able to be read through a time multiplex manner. So, this is another interesting example of you know using mod eight counter in combination with a multiplexers.

So, if it is a 16 to 1 you know 16 such parallel data read to be read by 1 output line in different point of time, then this would be 16 to 1 multiplexer and then it will be a a mod 16 counter is not it. So, you just need extension of that.

(Refer Slide Time: 29:43)

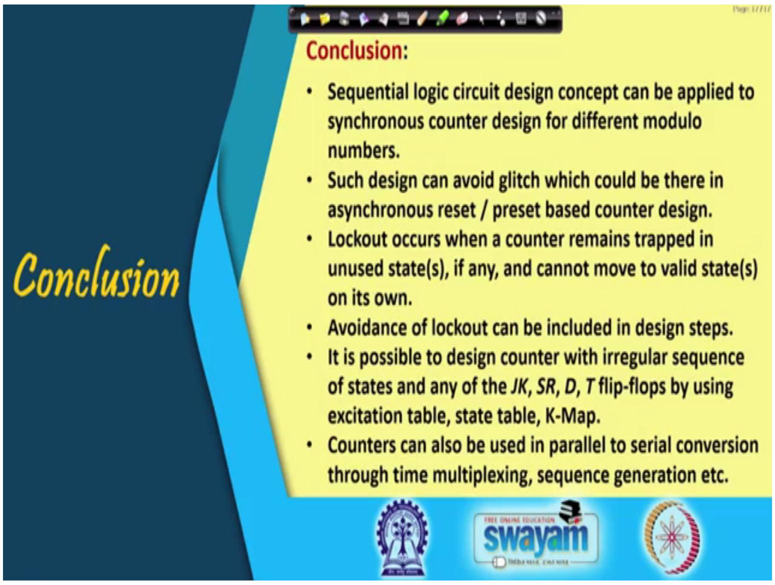


References:

- Donald P. Leach, Albert P. Malvino, and Goutam Saha, Digital Principles & Applications 8e, McGraw Hill

So, with this we come to the conclusion.

(Refer Slide Time: 29:44)



Conclusion:

- Sequential logic circuit design concept can be applied to synchronous counter design for different modulo numbers.
- Such design can avoid glitch which could be there in asynchronous reset / preset based counter design.
- Lockout occurs when a counter remains trapped in unused state(s), if any, and cannot move to valid state(s) on its own.
- Avoidance of lockout can be included in design steps.
- It is possible to design counter with irregular sequence of states and any of the *JK*, *SR*, *D*, *T* flip-flops by using excitation table, state table, K-Map.
- Counters can also be used in parallel to serial conversion through time multiplexing, sequence generation etc.

So, what we have understood is that sequential logic circuit design concept can be applied to synchronous counter design for different modular numbers and such design can avoid the glitch which otherwise is same for asynchronous reset based counter design. Lock out is possible when un in states are not considered and the counter remaining trapped in that ok.

So, it is better to avoid it by explicitly con considering it in the design steps and of course, J K flip flop, T flip flop we have seen before and also SR flip flop for counter design, but we are also know that D flip flop also can be used right for design. And the corresponding excitation table can be used for this design process. And the counters can have real set of applications we want that to thing that parallel to serial conversion. Together used together with a multiplexer or sequence generation where a combinatorial logic was used in addition ok. So, with this we conclude.

Thank you.