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

## Lecture – 41
## Asynchronous Counter

(Refer Slide Time: 00:27)

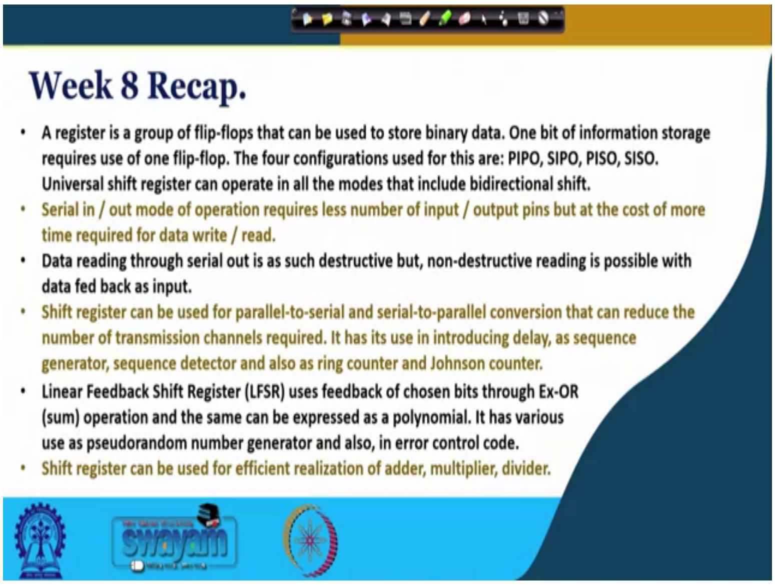
CONCEPTS COVERED

Concepts Covered:

- ❑ Week 8 Recap.
- ❑ Counter Basics
- ❑ Asynchronous Up Counter
- ❑ Asynchronous Down Counter
- ❑ Asynchronous Up – Down Counter

Hello everybody. We are in week-9 of this course. In this week, we shall discuss counter. So, we begin with a discussion on Asynchronous Counter, which we shall take up in this particular class. Before that we shall have a quick recap of what we discussed in week-8.

(Refer Slide Time: 00:33)



## Week 8 Recap.

- A register is a group of flip-flops that can be used to store binary data. One bit of information storage requires use of one flip-flop. The four configurations used for this are: PIPO, SIPO, PISO, SISO. Universal shift register can operate in all the modes that include bidirectional shift.
- Serial in / out mode of operation requires less number of input / output pins but at the cost of more time required for data write / read.
- Data reading through serial out is as such destructive but, non-destructive reading is possible with data fed back as input.
- Shift register can be used for parallel-to-serial and serial-to-parallel conversion that can reduce the number of transmission channels required. It has its use in introducing delay, as sequence generator, sequence detector and also as ring counter and Johnson counter.
- Linear Feedback Shift Register (LFSR) uses feedback of chosen bits through Ex-OR (sum) operation and the same can be expressed as a polynomial. It has various use as pseudorandom number generator and also, in error control code.
- Shift register can be used for efficient realization of adder, multiplier, divider.

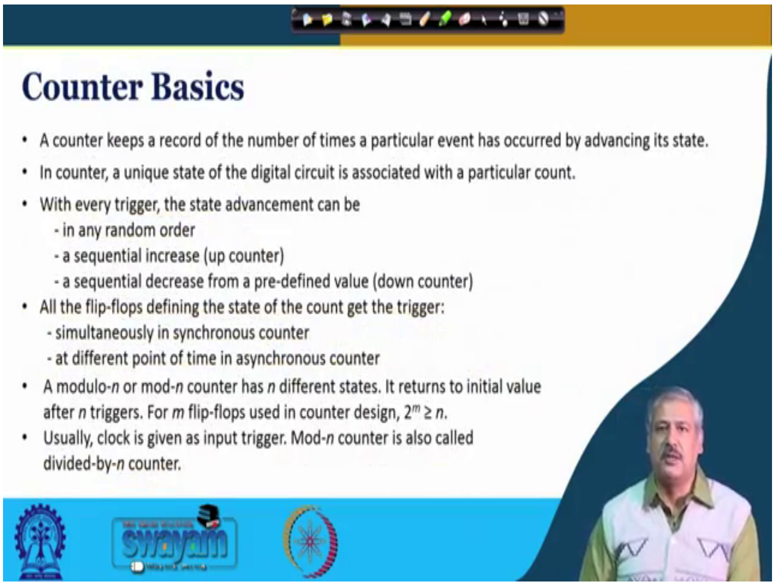
So, if you remember we discussed shift register, and there we noted that the number of bits in the register defines how many information can be stored, digital information can be stored. And we discussed various configuration of it parallel input parallel output, serial input parallel output, parallel input serial output, and serial input serial output.

And we noted that serially and serial out gives the option of having less number of input output pins, but the time required for data write and data read is more. And we also noted that for serial data out the way we read the data in such case as such is destructive. But, if we feed the data back, then it will be the data can be restored rewritten in the shift register.

And we saw a various application of shift register serial-to-parallel, parallel-to-serial conversion, and then using that to for a serial-to-serial communication which reduces the number of transmission channel required that is required. Then the sequence generator, sequence detector, ring counter or Johnson counter introducing delay of appropriate of required time.

And then we discussed at length linear feedback shift register, and its various use. And as pseudo random number generator, it has various use in from cryptography to many different places. And we also saw a cyclic redundancy check generation, and also detection any error in that code that is code word for any especially, it was useful for bust error checking, so that kind of use also we have seen. And finally, we saw use of shift register inefficient realization of adder comes subtractor multiplier divider ok.

(Refer Slide Time: 02:51)



## Counter Basics

- A counter keeps a record of the number of times a particular event has occurred by advancing its state.
- In counter, a unique state of the digital circuit is associated with a particular count.
- With every trigger, the state advancement can be
  - in any random order
  - a sequential increase (up counter)
  - a sequential decrease from a pre-defined value (down counter)
- All the flip-flops defining the state of the count get the trigger:
  - simultaneously in synchronous counter
  - at different point of time in asynchronous counter
- A modulo-*n* or mod-*n* counter has *n* different states. It returns to initial value after *n* triggers. For *m* flip-flops used in counter design, $2^m \geq n$.
- Usually, clock is given as input trigger. Mod-*n* counter is also called divided-by-*n* counter.

So, with this we discuss we start discussion on counter. So, counter what is a counter? So, counter keeps record of the number of times a particular event has occurred ok, so it keeps a count, so the way I mean the name itself can give us an idea. And to note how the how many such event has taken place. So, the counter circuit with the memory element in it, it advances its state one by one ok. And one particular state is associated with one specific count. So, there is a uniqueness in this particular thing. So, a specific set say state is associated with one count.

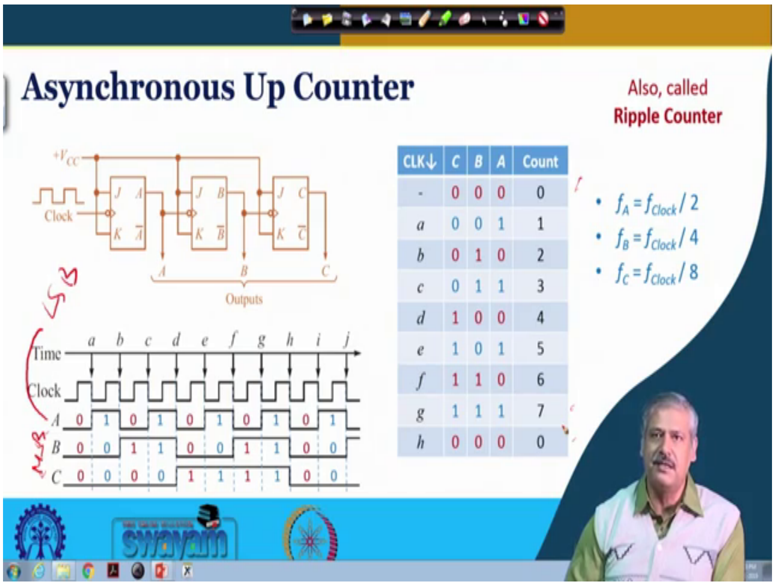
And if this advancing of state is in sequential order ok, so 0000 to 0001, then 0010 so on and so forth. Then if it is advancing it is increasing, then it is up counter. And if it is decreasing, the other way it happens it is called down counter ok, but as I said the basic idea is you need to have a unique state. So, it is not necessary that it is always need to be in that manner. So, it could be any random, but unique state, so but as many number of state has many count that is required here. So, in that is also possible in that case, it is random or irregular sequence will be there.

And the way the events triggered all the flip-flops that is there in the counter design in the make of the counter, it can happen that the trigger is going to each of the flip-flop. And simultaneously each of the flip-flop can get triggered, so that is called synchronous synchronously it is happening, so that is called synchronous counter. And if that if the trigger, the external trigger that is coming that is going to one of the flip-flop, then that flip-flop in turns in turn triggers the next and so on and so forth ok.

As if the effect of the one is going fast, you know to the other through in a ripple manner, so that is called ripple counter, also it is because how it is happening in different point of time, so that is that is why, it is called not exactly in a same synchronous with the external trigger. So, it is also called asynchronous counter ok. A modulo-n or modular mod-n encounter has got in different states ok, so that the number that is the weight it is associated.

And after in such trigger after the count of n, it comes back to its initial value. And if there are n flip-flops used for counter design, of course 2 to the power m need to be greater than or equal to m ok. And usually in most of the applications the discussion that we have clock is given as input trigger, but it could be some other signal also. And modern counter is also called divided-by-n counters, so we shall soon see that the output of the counter effectively will be having the rate of change of the signal, which is original signal that is the clock or the input trigger divided by the modular number ok.

(Refer Slide Time: 06:27)



Asynchronous Up Counter

Also, called Ripple Counter

| CLK↓ | C | B | A | Count |
|---|---|---|---|---|
| - | 0 | 0 | 0 | 0 |
| a | 0 | 0 | 1 | 1 |
| b | 0 | 1 | 0 | 2 |
| c | 0 | 1 | 1 | 3 |
| d | 1 | 0 | 0 | 4 |
| e | 1 | 0 | 1 | 5 |
| f | 1 | 1 | 0 | 6 |
| g | 1 | 1 | 1 | 7 |
| h | 0 | 0 | 0 | 0 |

- $f_A = f_{Clock}/2$
- $f_B = f_{Clock}/4$
- $f_C = f_{Clock}/8$

So, if this basic understanding of how counter works a counter is supposed to work, we look at asynchronous up counter, we begin our discussion with asynchronous up counter ok. So, for that we are having a circuit made up of JK flip-flop, we could have some other flip-flop as well, but JK flip-flop we have taken it up ok. So, what you see here that both the inputs of each of the JK flip-flop is connected to one tied to V CC that means 1.

So, each of the flip-flop when they get the trigger ok, the clock at the negative edge of it ok, this is negative edge triggered all of you can see that. So, it is supposed to toggle ok, so that is for sure. Now, let us see how things move. So, the clock is given here only to the first flip-flop right, as I was saying that is it is a synchronous. So, the clock is not fed to each of the flip-flop, only the first flip-flop is getting the clock right, then what will happen? After the negative edge, so initially all of the flip-flops are initialized with the values at 000 ok.

So, after the negative edge comes, as you see the timing diagram, it will toggle it will become 1 another negative edge comes ok, so this flip-flop will become 0 right. So, then again 1 0 1 0 1 0 and 1, so this is the way it will continue is it clear. So, this is how the first flip-flop will work right. We will discuss about second and third little later. So, the clock is directly connected the external trigger, which is getting counted how many clock cycles or how many negative edge of the clock occurring that is going directly to the flip-flop A. And the flip-flop A is struggling in each of the negative edge in each clock cycle right at this points a, b, c, d, e, the way you have seen it is it fine.

Now, what happens, this flip-flop B is getting a feel getting an idea about the triggering through A not directly from the clock. So, whenever A changes, A output A uncomplimented output A is connected as clock 2 flip-flop B. So, whenever A changes from high to low right, so a trigger comes that is something as a clock right. Now, for every two change in the clock, you can see a changes by 1, these two cycle these are the two cycles, then A changes by changes, there are two negative edges here, then there is one negative edge of the clock ok.

So, for every two such clock trigger, you get one trigger out over here for flip-flop B ok. So, it is coming via A right, it is coming as you know kind of the effect is rippling through flip-flop A to coming to flip-flop B right. So, at the negative edge of the A flip-flop output right, the B flip-flop will toggle that is the idea. So, a B is remaining 00 over here for this time instances. So, only at time instant B, when it gets a negative edge, it goes it toggles, so it goes from 0 to 1.

And how long you will need to remain in one? As long as next negative edge from A comes here, because A is fed as clock. So, next negative edge A comes at d, you can see here right. So, at the time from 11, it comes to 0. So, next negative edge will come at f, so it goes from 0 0 to 1 1, and this way it goes on is it fine ok. Now, let us look at C, what happens to C? C is getting triggered from B ok. So, similarly whenever B goes from high to low that time this C is triggered, and triggered in the sense it is both the inputs are 1, so it will toggle. So, let us see when B is going from high to low, so B is going from high to low at d. And again at h right, so these are the instances 1 to 0, and again 1 to 0.

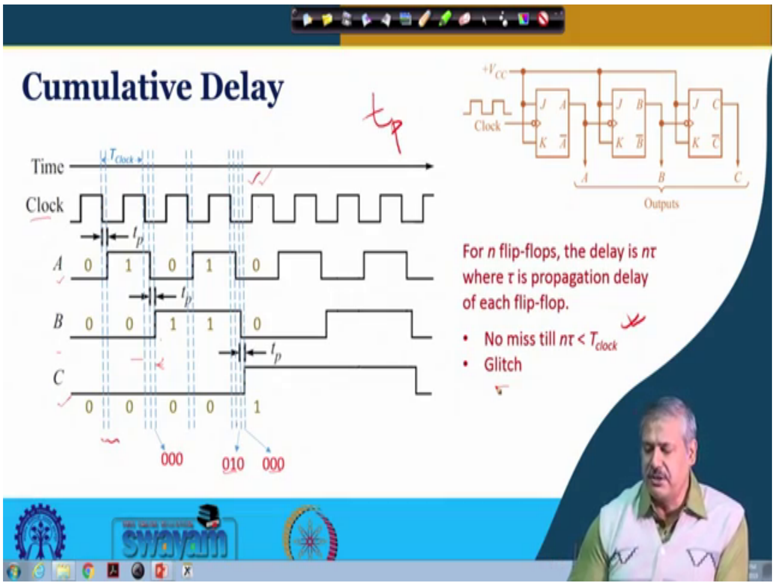
So, C is changing, C is remaining 0 0 0 0 ok, because it does not got any trigger right though flip-flop A has got triggered in every clock cycle, B has got triggered in every two clock cycle, C has not got triggered up to this only in the fourth clock cycle, you can see a, b, c, d right, it is getting a trigger ok. And at that time since it is toggling, it goes from 0 to 1. And it remains at 1, till next negative edge of B comes, and it goes from 1 to 0, and this way it continues is it fine. So, this is what is happening to the flip-flops right.

Now, by this what have you achieved in terms of you know count. So, if you now look at the values of C, B, A, if you read it in this manner C MSB, B C is your MSB, and A is your LSB, if you read it this way. So, 0 0 0, 0 0 1, you can read in each of this clock cycle right up to a. From a to b, it is 0 0 1; b to c, it is 0 1 0; c to d, it is 0 1 1 ok.

So, these are the you know clock cycles. And 0 1 1, 1 0 0, 1 0 1, 1 1 0, 1 1 1 right, what is it in binary 0 to 7 you have come, then again it is 0 0 0 ok. So, you have got these states if you put it in the in the tabular form 0, 1, 2, 3, 4, 5, 6, 7, and again it is coming to 0 right. So, what is it, it is nothing but up count the number is sequentially increasing right. And it is a modulo 8 counter, because 8 distinct states up there.

Now, if you look at the output of flip-flop A right, you can see for every two changes in clock one changes happening here. So, clock cycle the period of time period of A one period of A is just half of sorry twice that of the clock period right two clock period we are consuming in one cycle of A. So, frequency wise for every two, you know cycle one cycle of A is seen, so the frequency of A is half of it, then half of the clock. Frequency of B, one-fourth, and frequency of C that is the last flip-flop over here, you are having a frequency which is divided by 8. So, you can see that divided by 8, it is a modulo 8 counter, because 8 sets are there. So, you can get up to I mean if you from the last output a divided by 8 count output.

(Refer Slide Time: 14:05)



Cumulative Delay

For $n$ flip-flops, the delay is $n\tau$ where $\tau$ is propagation delay of each flip-flop.

- No miss till $n\tau < T_{clock}$
- Glitch

So, far so good we have got a up counter modulo 8 up counter by a very simple mechanism the output of one is fed as clock to the next flip-flop, and each flip-flop is toggled in. Now, in these what was not visible in the previous case that is the timing diagram, so that we are trying to show by stretching the diagram in somewhat ok, and introducing the time delay associated with each flip-flop transition ok.

So, let us see what is it, so this is the clock right. So, clock has got a negative edge over here. Now, flip-flop A is changing, but it will take a propagation delay which is defined here as t suffix p right. So, after that time only, A will go up. Similarly, we will go down here after that propagation time of propagation delay t p right is it fine, so that will happen in every cases.

Now, flip-flop B. So, flip-flop B is not fed by the clock directly, it is fed by the flip-flop A right. So, when flip-flop A changes, after that one more propagation delay, which is required inside flip-flop B for its output to change that will be there. So, from the clock clocking instant over here the negative edge, B changes change of B will take place change of A required that is one propagation delay plus change of B that is required.

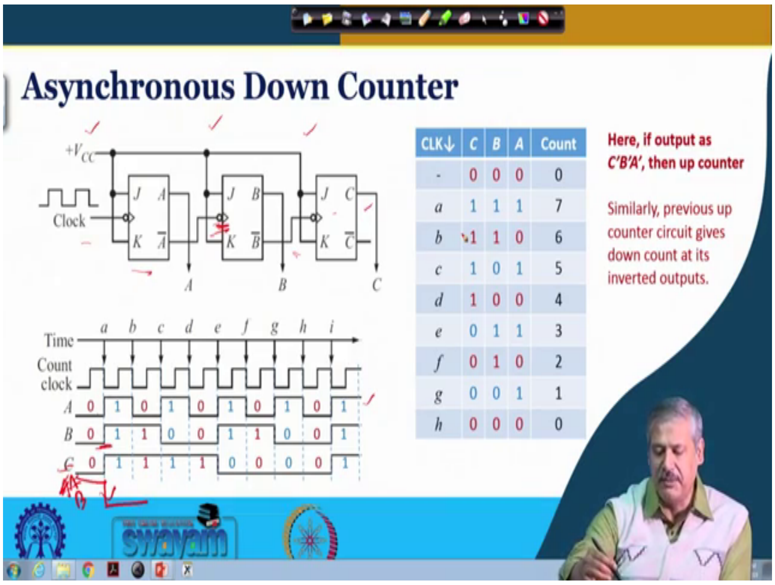
So, two propagation delay will be there within this, so that is what is shown here. So, two propagation delay will get involved. And for C, what will happen? C is not again fed by clock triggered by clock, it is triggered by B. So, B is taking two propagation delay from the clocking instant, so it will take another propagation delay. So, three propagation delay will come into picture, is it clear.

So, the way this way you can see in this arrangement, the propagation delay are cumulative right. And if you are you know having a very fast clocking right, then this propagation delay, when they add it up for an n-bit counter right, and becomes comparable with the time period of this particular clock ok, then there can be problem ok.

So, these addition, when it goes extend beyond this time period, then what will happen, within this time period over here never this 1 0 0 will occur. So, for example over this is the way you see 0 1 1, then 1 0 0 right. So, this is the corresponding clock cycle ok. So, 0, 1, 2, 3, 4, this is the fourth clock cycles after you know starting from 0 ok. So, 1, 2, 3, 4 right, so this is the fourth clock edges have come right. But, if it is extends beyond this, so fifth clock edge will come and 1 0 0 is not shown, so that is what is something comes as a cause of concern. So, this is one issue with the with this kind of you know arrangement right.

Another is what you can see after 0 0 0, this is 0 0 1, after that 0 1 0 is supposed to come in up counter. But, momentarily you can see for a small duration 0 0 0 is there, because A has changed, but B is taking one more propagation delay to change ok. Similarly, over here after 0 1 1, you are expecting 1 0 0 0 to come 1 0 0 all right. But, momentarily you can see that 0 1 0 is there, and then 0 0 0 is there right, after that 1 0 0 is coming right. So, this can cause glitch in the circuit ok, so this is something which one need to be careful about ok. And we shall see solutions and many other things before later, but this is something which we taken not in asynchronous counter in this discussion this part of the discussion.

(Refer Slide Time: 18:37)



Asynchronous Down Counter

| CLK↓ | C | B | A | Count |
|---|---|---|---|---|
| - | 0 | 0 | 0 | 0 |
| a | 1 | 1 | 1 | 7 |
| b | 1 | 1 | 0 | 6 |
| c | 1 | 0 | 1 | 5 |
| d | 1 | 0 | 0 | 4 |
| e | 0 | 1 | 1 | 3 |
| f | 0 | 1 | 0 | 2 |
| g | 0 | 0 | 1 | 1 |
| h | 0 | 0 | 0 | 0 |

Here, if output as C'B'A', then up counter

Similarly, previous up counter circuit gives down count at its inverted outputs.

Now, we look at asynchronous down counter ok. So, a synchronous up counter we have seen, so a synchronous down counter again we look at modulated counter. These are these are three flip-flops up to with these three 2 to the power 3 maximum modulo 8 counter is possible. So, in this what is done, you can see the circuit input side remains the same, its flip-flop will toggle ok. Clock is given to the first flip-flop right.

And over here the instead of A, A bar is fed as clock to the next flip-flop B. B bar is fed as clock to the next flip-flop that is C is it fine ok. So, by this what happens, so initial state is 0 0 0, and a will toggle in a every in a clocking and negative edge of the clock that is what sure, this is the first flip-flop right, so that is 0 1 0 1 that is what you see for A, which is which is similar to what you had seen before.

Now, because of A bar with fed as clock what happens, so this will change at negative edge right. So, A is going from in this particular place, A is going from high to low ok, sorry A is going from here low to high. So, what will happen to A bar, A bar will go from high to low, is not it. If you plot A bar alongside ok, A is going from high to low. So, A bar will go from A is going from low to high, so A will go from high to low. So, there is a negative edge. There is a negative edge over here which will trigger it, trigger it means it will toggle, so B will go from then 0 to 1 is it fine.

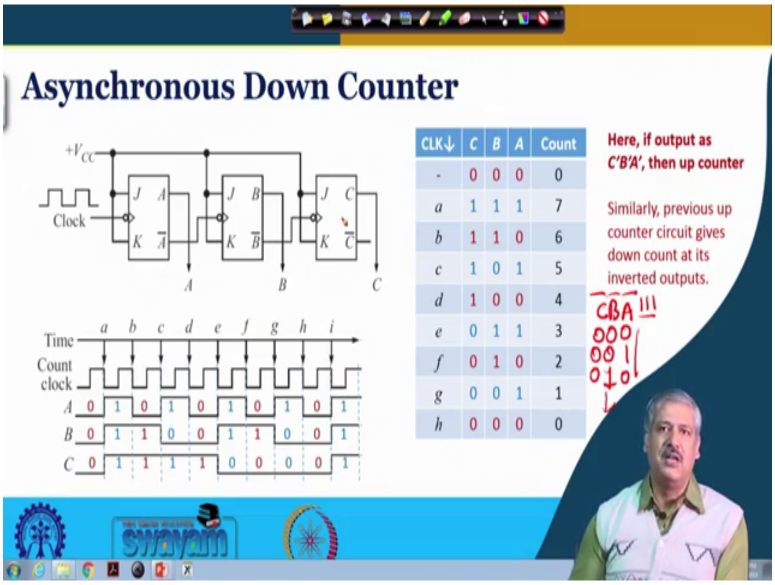
Now, when B goes from 0 to 1 B goes from 0 to 1 right, B bar what happens to B bar? B bar will go so that was A bar before, now B bar will go from high to low 1 to 0, so that will also trigger that will trigger C. So, C will go from 0 to 1 is it clear, because it is just opposite of it right. Whenever there is a positive edge for A ok, there is a negative edge for A bar. Similarly, for B bar and C bar is not fed anywhere, so it would have been for C bar also ok, if there was a flip-flop D right.

So, at positive edges of A, B, actually transition are happening ok, because for the subsequent flip flops, because A bar and B bar are fed as clock. So, these positive edge is over here, so 0 0 0, it becomes 1 1 1 right. Next these A is changing right, and A is changing over here, it is going from high to low. And it is high to low means, A bar is going from low to high, so no change right understood.

Again A is going from low to high here, so output will go for A bar will go from high to low. So, here A bar is going from high to low, so there is a change in B, so it is coming from going from 1 to 0. So, this is the way, it continues. So, what you can see 1 1 1, after that 1 1 0, after that 1 0 1 ok. Again B goes from low to high here, so high to low will be in this particular place.

So, C changes over here ok. And C does not change, when B bar goes from high to low, because B bar is what is fed here unlike in the previous case. So, this way you can see 1 1 1, this way if you read MSB to LSB C to A 1 1 0, 1 0 1, 1 0 0, 0 1 1 ok, and these are the counting states right is it fine. And it is up to going up to 0 0 0 right, 0 0 1, and then it is 0 0 0, after that again 1 1 1 will get loaded right. So, it is mod 8 down counter.

(Refer Slide Time: 23:47)



Asynchronous Down Counter

| CLK↓ | C | B | A | Count |
|---|---|---|---|---|
| - | 0 | 0 | 0 | 0 |
| a | 1 | 1 | 1 | 7 |
| b | 1 | 1 | 0 | 6 |
| c | 1 | 0 | 1 | 5 |
| d | 1 | 0 | 0 | 4 |
| e | 0 | 1 | 1 | 3 |
| f | 0 | 1 | 0 | 2 |
| g | 0 | 0 | 1 | 1 |
| h | 0 | 0 | 0 | 0 |

Here, if output as C'B'A', then up counter

Similarly, previous up counter circuit gives down count at its inverted outputs.

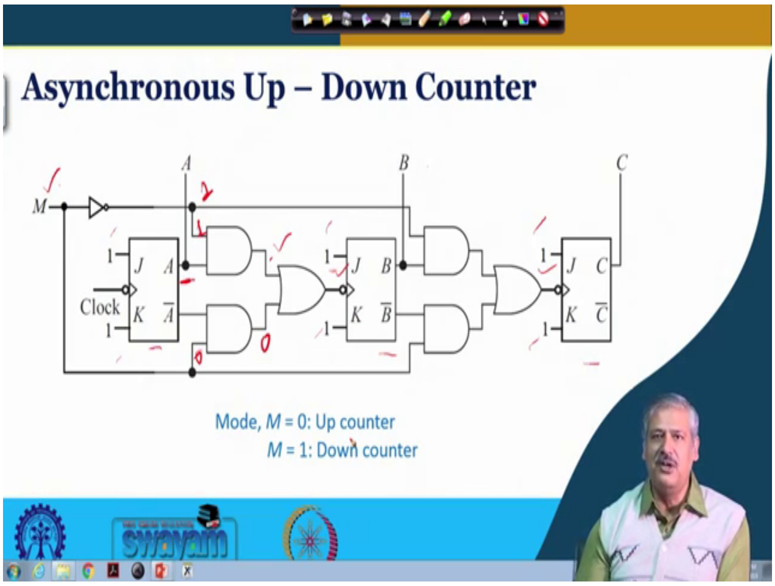
$\overline{C}\overline{B}\overline{A}$
000
001
010

Now, one important thing to be noted here; so, you have taken out put A, B, C right. Now, if inverted output is also available, please understand this part. If this counter circuit also allows you inverted output to the outside world that means, A bar, B bar, and C bar ok. So, what the reading for C bar B bar A bar would have been in these cases, and you know identify.

So, for example this was 0 0 0 right, after that it has gone to 1 1 1 ok. So, 1 1 1, and then 1 1 1 inverted it is 0 0 0 only ok, so after that it is 1 1 0. So, 1 1 0 is your what invert it is 0 0 1, then 1 0 1. So, one is one inverted will be 0, C bar will be 0, this is 1, this is 1 0 1 1 0, and this is 0. So, just invert of this. And initially when it was 1 0 0 0, it would have been initially it was 1 1 1 we are awaiting that. So, this way it will continue.

So, what is it, it is up counter only up counting. So, the inverted outputs complemented outputs will if available will give you up count while the standard one uncomplimented outputs are giving you down counter for A bar and B bar fed as clock to the subsequent flip flops right. So, similarly in the previous arrangement the up counted, we had seen had with the option of taking the output from C bar B bar A bar, then output there would have shown down count ok. So, this we again take a note.

(Refer Slide Time: 25:31)



Asynchronous Up – Down Counter

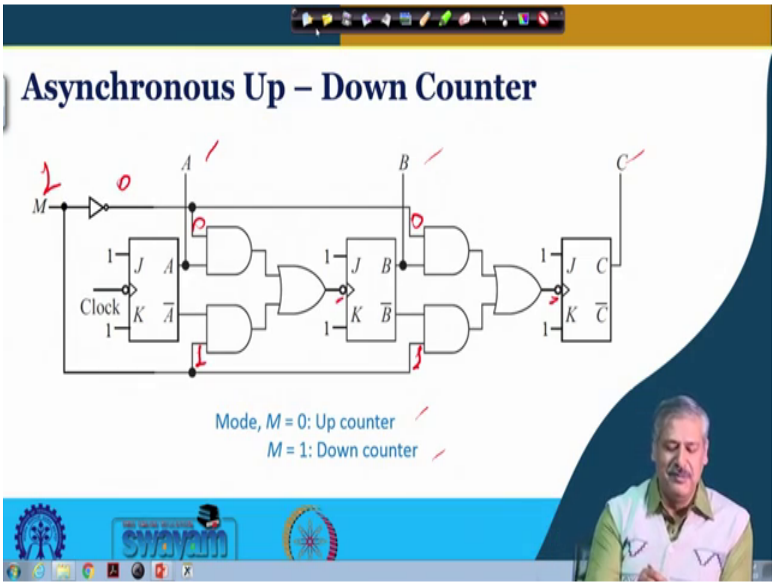
Mode, $M = 0$: Up counter
$M = 1$: Down counter

Now, having understood how up counter and down counter works in asynchronous counter ok; so, can you have one circuit ok, which can both work as up counter and down counter. And depending on our requirement, we can make it what like a up counter, and for another use, we can make it what like a down counter. So, here is a circuit. So, the flip-flops A, B, C are there, the inputs are 1 1 that means, for every clocking available here, it will trigger ok, it will sorry toggle ok, it will go to the other value. If it is 0, it will go to 1 1, it is it will go to 0 ok.

Now, we have put an additional input M, so M is can be considered as mode control input. So, M is 0 what happens, let us see for this particular circuit right. So, M is 0, you can see M here, this is two AND gate and one OR gate. So, it is more likely to I mean it is the way the 2-to-1 multiplex circuit you seen, but in a different context right.

So, M is equal to 0, this means this is equal to 1 right, this is equal to 1, and this is equal to 0. So, 0 means, these AND gate output is 0, irrespective of what is there in the other output ok, because for AND gate 0 is a forcing input. And since, it is 1. For these AND gate, whatever it changes in A that will go here right. And for OR gate 0 is a non-forcing input. So, whatever change is occurring, it will come here so, effectively M is equal to 0, A is coming here. Similarly, B is coming here right. And if A comes and B comes, the way we have seen the circuit, it is up counter only.

(Refer Slide Time: 27:25)



Asynchronous Up – Down Counter

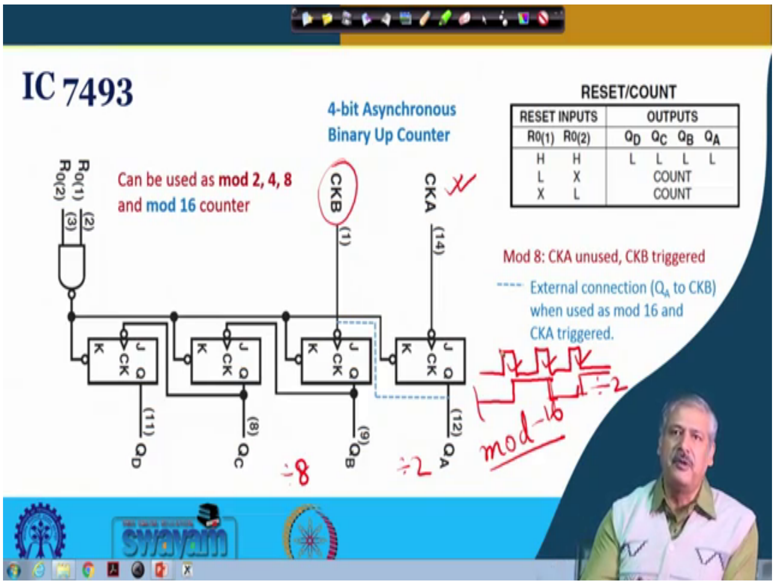
Mode, $M = 0$: Up counter

$M = 1$: Down counter

Instead of this if M is equal to 1, this is 0, so this is 0, and this is 1 right, so this is 0, and this is 1. So, what is happening at that time? A bar is going to the clock of B flip-flop, and B bar is going to the clock of C flip-flop right. So, what is this arrangement, if you are taking output from A, B, C, the uncomplimented you know output of each of the flip-flop ok, so then it is acting like a down counter.

So, M is equal to 0 up counter, M is equal to 1 down counter, this is the way we can make it work in one circuit both the things are there right. So, while it is counting it as counted up to a particular level value, then you are changing M, then again it will start counting in the opposite direction. So, it has count say counted in the up count mode 0, 1, 2, 3, 4, 5 ok, and then you have made it 1, so it will start down counting ok.

(Refer Slide Time: 28:29)



IC 7493

4-bit Asynchronous Binary Up Counter

Can be used as mod 2, 4, 8 and mod 16 counter

RESET/COUNT

| RESET INPUTS | | OUTPUTS | | | |
|---|---|---|---|---|---|
| R0(1) | R0(2) | $Q_D$ | $Q_C$ | $Q_B$ | $Q_A$ |
| H | H | L | L | L | L |
| L | X | COUNT | | | |
| X | L | COUNT | | | |

Mod 8: CKA unused, CKB triggered

---- External connection ($Q_A$ to CKB) when used as mod 16 and CKA triggered.

So, finally we would like to see one standard IC, which gives this asynchronous up count ok. So, IC 7493 is one such IC, which has got actually if you look at the inside the circuit, a mod 8 count counter over here, and the mod 2 counter ok, two such units are there in this particular right IC. So, this is clock B, and this is clock A right.

If you do not bother about clock A right, if you just feed the outside clock two clock B, then what will happen? This circuit will behave like a mod 8 up counter, you can see this Q B is going to the flip-flop Q C is going to the flip flop. So, it is not complemented output, it is uncomplimentary output is going. So, it will be behaving like a up counter all right, so this part is mod 8 up counter, and this part is mod 2 counter right.

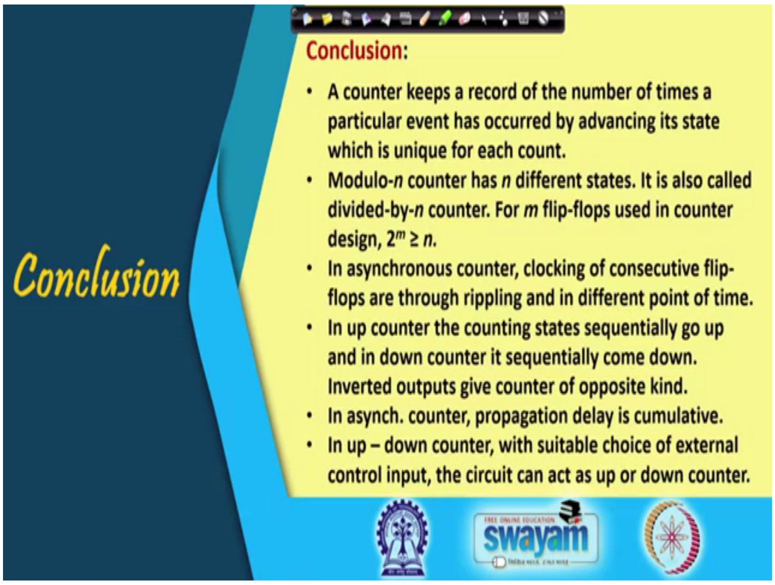
Now, instead of Q D, if you are just looking at Q B and Q C, then these are mod 2 and mod 4 right, you will get a count of that thing you have seen that before that it is a divided by 2 over here, divided by 4 over here, and divided by 8 over here ok, so that is the 1. But, as a whole three flip-flops are there, 8 unique states increasing order. So, it is a mod 8 up counter, if you take it as a whole right.

Now, if you make an if you make an external connection from Q A to clock B over here right, and feed the clock feed the clock here external clock here, then what will happen? So, now forget about this one it is not there, it is directly connected. So, instead of three flip-flops, now we are having four flip-flops. So, in the earlier circuit timing diagram, we stopped at A, B, and C. Then there will be a D coming over there right, so that D will change for every negative edge of C ok. And that will one full cycle of it will occur for 8 plus 8, 16 cycles of the clock right. So, it will become a mod 16 counter ok, because every here when you are feeding the clock, so if you cannot if you remember the circuit. So, this is your clock, so negative edge ok.

So, the Q A is changing over here, and then again over here, and then again over here right. So, this is halfed, so this is divided by 2, and this is divided by 8 right, so together it is divided by 16. So, modulo 16 count that is what we will get right, is it clear. So, extending what we had discussed before, for three flip-flops to four flips-flip flops-right. Additionally, here you can see there are two other inputs which are R 0 1 and R 0 2. If both of them are high both of them are high, then what will happen? This is NAND gate, so it will become low.

And you can see that there will be a synchronous clear synchronous reset right ok, it is required for initial addition for some other purposes that we shall discuss later. And any one of them low ok, then the normal count operation will take place up count operation will take place right. And why 2, why not 1, 1 would have been sufficient more and that will be revealed in the subsequent discussion on the counter ok. There is a specific reason to put this thing some such thing ok, which is helpful in certain context ok.

(Refer Slide Time: 32:53)



Conclusion

Conclusion:

- A counter keeps a record of the number of times a particular event has occurred by advancing its state which is unique for each count.
- Modulo-*n* counter has *n* different states. It is also called divided-by-*n* counter. For *m* flip-flops used in counter design, $2^m \geq n$.
- In asynchronous counter, clocking of consecutive flip-flops are through rippling and in different point of time.
- In up counter the counting states sequentially go up and in down counter it sequentially come down. Inverted outputs give counter of opposite kind.
- In asynch. counter, propagation delay is cumulative.
- In up – down counter, with suitable choice of external control input, the circuit can act as up or down counter.

So, to summarize counter keeps a record of the number of times a particular event has occurred by advancing its state which is unique for each count. And for the modulo-n counter, n different states are there. And if m flip-flops are there, then with those m flip-flops, we can go up to 2 to the power m modulo number. Asynchronous counter, the clapping of consecutive flip-flops are not by not from the same clock or the external trigger, it is done through a rippling effect from one flip-flop to another.

In up counter the counting states sequentially go up, and in down counter it is sequentially come down ok. And if you have the option of taking the output from the inverted outputs, then the counter of opposite kind can be available. And in asynchronous counter important thing is that you have seen the propagation delay is cumulative for which the is I mean in this cumulative delay is of the order of the clock time period, then a count can get missed. And also it can it has, it can give glitch in between values may come up ok, which can cause problem. And up-down counter can be made we where one same circuit can give both up count as well as down by suitable choice of controlling.

Thank you.