

**Switching Circuits and Logic Design**  
**Prof. Indranil Sengupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 46**  
**Design of Counters (Part – II)**

In this lecture, we continue our discussion on Counters. If you recall in the last lecture we talked about the design of binary ripple counters; both up counting and down counting modes, where the counters were counting modulo some power of 2. If it was a 3 bit counter it was module 8; module 8 means it counts from 0 to 7 and back to 0; modulo means divide by 8 and take the remainder. So, it will go up to 0 to 7.

Modulo 16 for a 4 bit counter, 0 to 15 and then back to 0 and so on. Now in this lecture to start with we shall be discussing the design of arbitrary modulus counter not necessarily to the power of 2.

(Refer Slide Time: 01:12)

**Design of Binary Ripple Counter of any Arbitrary Modulus**

- Suppose we want to design a *modulo-M counter*, for any arbitrary value of  $M$ .
  - Counter will count up from 0 to  $M-1$ , and then back to 0.
- Assume that the flip-flops have individual asynchronous  $CLR'$  inputs.
- Basic idea:
  - We first design a ripple counter with  $\lceil \log_2 M \rceil$  stages.  $\underline{\underline{3}}$   $\frac{M=14}{4}$
  - Then we design a gating circuit, which takes inputs from the counter outputs, and generates a 0 whenever the count value reaches  $M$ .
  - Connect the output of the gating circuit to the  $CLR'$  inputs of the flip-flops.

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

So, the second part of counter design; we start with the design of binary ripple counter of arbitrary modulus. So, what do mean by arbitrary modulus? We are saying that in general we are trying to design a modulo-M counter for any value of M.

Let us say M is 6; so if we have this kind of a scenario then the counter will count from 0 0 1 2 3 up to M minus 1 and then again back to 0. So, as I said modulo M means divide

by  $M$  and take the remainder. So, if  $M$  equal to 6; I am saying number of pulses that have come divide it by 6 and take the remainder. So, the value will be something from 0 to 5 that will be the final value in the counter right.

Now, let us look at the general design principle for such a counter. Now we make some assumptions here based on that we arrive at the design. The first thing is that we are trying to design an up counter, down counter can be design in a very similar way, but for the time being we are assuming that it is an up counter; it will count in the upward direction 0 1 2 3 4 up to  $M$  minus 1. Now we are assuming that the flip flops have individual asynchronous clear inputs, which is active low which means if the clear input is set to 0; all the flip flops will be reset to 0.

The idea is very simple. I do not want the counter to go to  $M$ ; after  $M$  minus 1 I want it to come back to 0. So, whenever it tries to reach  $M$ , I will forcibly bring the counter to 0; I will clear the flip flops, the idea is that; not 6 counter 0 1 2 3 4 5 as soon as it tries to become 6, I will activate the clear inputs they will all become 0 again ok; this is the basic idea.

Now the first thing is that for a modulo  $M$  counter we need  $\log M$  number of flip flops ceiling means the smallest integer that is greater than or equal to this number. Let us say if  $M$  equal to 6  $\log M$  will be how much?  $\log M$  will be 2 point something. So, ceiling will be 3; so, I will be needing 3 flip flops let us take  $M$  equal to 14; then  $\log M$  to the base 2 will be 3 point something. So, ceiling of that will be 4 I would need for flip flops.

So, here you take a decision how many flip flops are required then we use a simple gating circuit a simple circuit as I show you how to design it which will take inputs from the counter outputs and it will generate a 0 in the output whenever the count value reaches  $M$ ; that means, I will not allow the counter to reach  $M$ ; I will make the counter 0 immediately. So, whenever the output of this gating circuit is 0 that is connected to the clear inputs of the flip flop with flip flop will become 0 again.

(Refer Slide Time: 04:52)

**Example: Design of a modulo-6 binary ripple counter**

- Number of flip-flop stages required will be  $\lceil \log_2 6 \rceil = 3$
- The desired count sequence will be: 000, 001, 010, 011, 100, 101, 000, ...
  - From state 101, instead of going to the state 110, we have to bring the state back to 000.

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

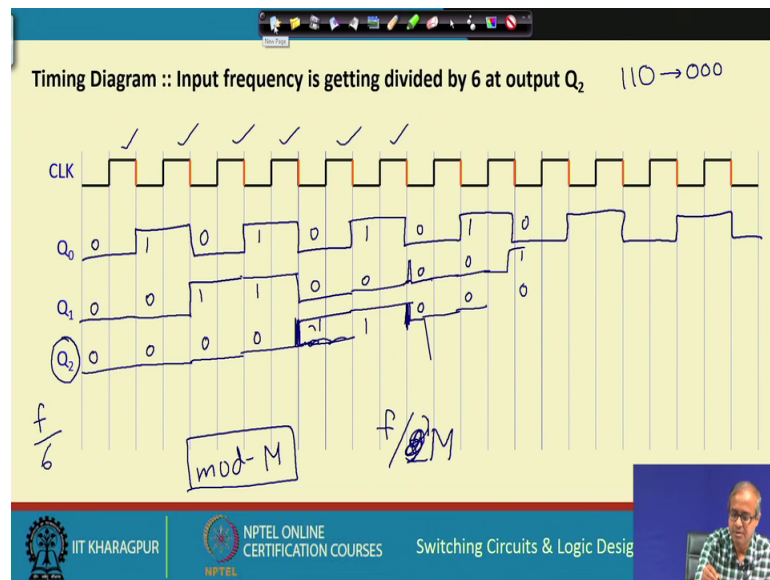
So, let us take an example of a modulo 6 counter; so, how to design it. So, for 6 number of flip flops as I said will be equal to 3 log 6 to the base 2 will be 2 point something, ceiling will be the next higher integer 3. Now for a mod 6 counter we want the counter to count in this sequence 0 1 2 3 4 5 and then back to 0; which means I am not allowing the natural next count which is 6; 1 1 0 to come.

So, whenever it tries to become 1 1 0; I will forcibly make it 0 again. So, how do I do it? This is what I mentioned, it can be done very simply like this you see you just ignore this NAND gate for the time being; the first 3 flip flop is just like a modulo 8 counter.

3 flip flops where the output of a flip flop is feeding in the clock input of the next flip flop; active load triggered. Now what this NAND gate is doing? NAND gate is trying to look for whether the count is reaching 1 2 0 or not 1 1 0 means what? This is 1, this is 1 and this is 0; so, you look at the ones the outputs which are supposed to be 1 in this invalid state; you feed them as the input of this NAND gate. So, whenever these 2 bits are trying to become 1 1; the output of the NAND gate will become 0. And this 0 will immediately clear the flip flops; so, the idea is that when you are in state 1 0 1.

Suppose you are in state 1 0 1 and you apply a clock it will temporarily go to 1 1 0 and immediately the counter will get cleared; this 1 one 0 will be a temporary state right. This is how we can design such a counter with arbitrary modulus right.

(Refer Slide Time: 07:26)



Now, let us work out for a divide by 6 counter as we have seen here. Now whenever there is 1 1 0; it will clear, just you remember this thing whenever the count value will be 1 1 0; it will be immediately clear to 0 0 0. Now let us look at this  $Q_0$ ;  $Q_0$  will be going on changing at every falling edge.

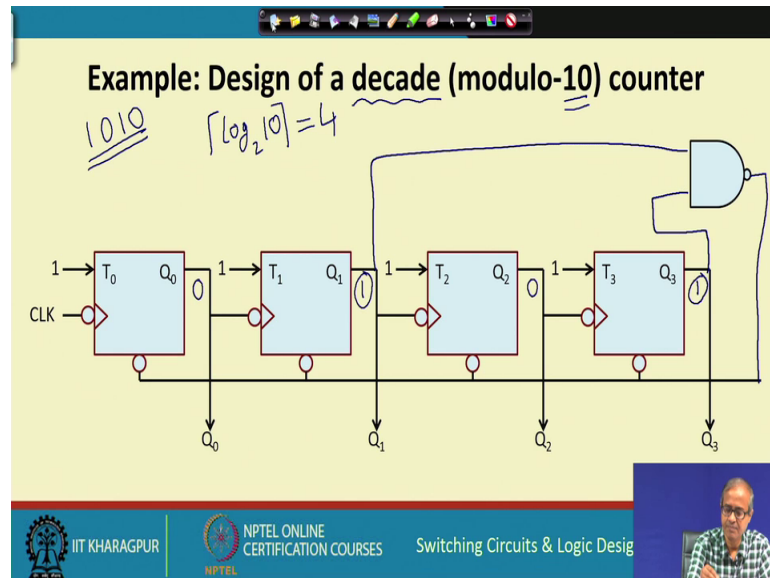
There is no issue with  $Q_0$ ; let us start with all 0. So, we have 0 0 0; so, the first clock comes it becomes 0 0 1, next clock comes there is a falling edge; this will change state it becomes 0 1 0 2. Next clock is comes this will continue 0 1 1 which is 3, next clock is comes there is again a falling edge this will change again sorry here there is a falling edge this will go up yes. So, this will be 1 0 0 this will become 4 then again there is a falling edge here this will not change, this will also not change 1 0 1 this is 5.

Now, here temporally it will try to become 6 because there is a falling edge here; it will go up this is already high. So, it is 1 1 0, but as soon as this tries to go up the counter will be reset; it will become 0 right. So, it will become 0 0 0 again and this will continue 0 0 0 again in the next clock, it will become 0 0 1, again it will become 0 1 0 and so on right.

Now the point to note is that if you look at the  $Q_2$  output you see it is starting from 0; it is going up here, it is again going down here. So, how many clock pulses are required, 1 2 3 4 5 6. So, after every 6 clock pulses this  $Q_2$  is completing 1 pulse which means I can say in  $Q_2$ ; the frequency will get divided by 6.

So, the basic observation is any mod M counter that we design if we take the last output in this case Q<sub>2</sub>. So, the input frequency will get divided by 2 to the power M not 2 to the power M; input frequency will be divided by M, it is a module M counter. In this case it was mod 6 counter, the frequency was getting divided by 6 right this time in diagram also shows that let us continue.

(Refer Slide Time: 11:35)

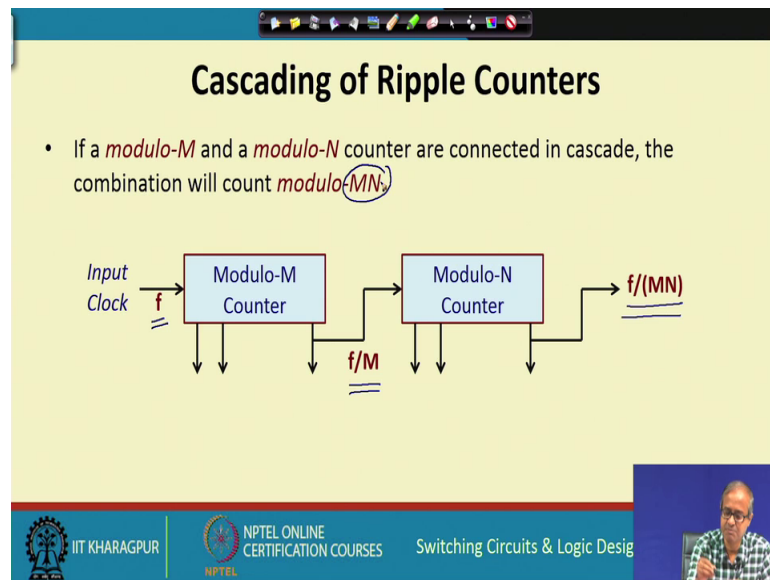


Let us now look at a very widely used kind of a mod M counter which is mod 10; this is also called a decade counter decade is nothing, but 10 this is sometimes called a decade counter.

Now, if M equal to 10 if you take log 10 to the base 2 and ceiling of it will be 3 point something. So, this is 4 you need 4 flip flops and 10 is what? 10 is 1 0 1 0. So, whenever you try to reach 1 0 1 0 you must reset it to 0; 1 0 1 0 means 1 0 1 0. So, Q<sub>3</sub> and Q<sub>1</sub> are trying to become 1; so, in this NAND gate you will have to connect the output Q<sub>3</sub> and connect the output Q<sub>1</sub>.

And the output of this NAND gate you will have to connect to the clear input this will make your decade counter. So, just by selectively connecting the outputs to this NAND gate you can design any arbitrary mod M counter. In this case I required mod 10; so, I connected the outputs corresponding to the number 10; 1 0 1 0, 1 0 1 0 1 is here 1 is here I connected like. So, I get a mod 10 counter right simple.

(Refer Slide Time: 13:19)



**Cascading of Ripple Counters**

- If a *modulo-M* and a *modulo-N* counter are connected in cascade, the combination will count *modulo(MN)*.

The diagram illustrates two counters in cascade. The first counter is labeled 'Modulo-M Counter' and receives an 'Input Clock' with frequency  $f$ . It has three output lines, with the bottom-most line labeled  $f/M$ . This  $f/M$  signal is connected to the clock input of the second counter, labeled 'Modulo-N Counter'. The second counter has three output lines, with the bottom-most line labeled  $f/(MN)$ . The slide footer includes the IIT KHARAGPUR logo, NPTEL ONLINE CERTIFICATION COURSES logo, and the text 'Switching Circuits & Logic Design'.

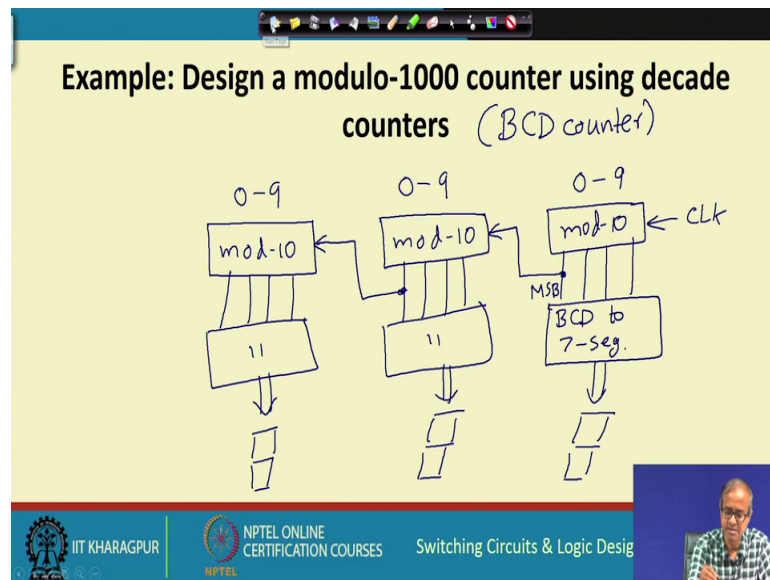
Now, next point is how do I cascade counters and what will happen if I cascade 2 counters? Cascading counter means connecting them like this suppose I have a mod M counter; the last stay last means the most significant bit that is connected to the input clock of the next counter.

If I have a mod M counter or an mod N counter the point to remember is that you already know; I have seen I have shown for a mod 6 counter that if I have a mod M counter; if I input clock has a frequency  $f$ , then in the more significant bit output the frequency will get divided by M. But if this signal is now fed to the clock of another counter mod N this will also do a frequency division by N. So, in the final output the frequency will be divided by MN; so, you should remember this.

If counters are cascaded then the modulo will be the product of the individual modulus mod M mod N; if you connect them in a cascade it will become mod MN modulo MN counter right. Now this principal can be used in the design of arbitrary kind of counters; let us take some specific examples.

Let us see I want a modulo 1000 counter like we have so many kind of equipments where some digits are being displayed in decimal, like a frequency counter or any kind of digital displays. Numbers are normally displayed in decimal; so, how do I implement a 3 digit counter in that case? Because numbers I mean decimal 0 to 9 and then again back to 0 suppose in a 3 digit I need a modulo 1000 counter.

(Refer Slide Time: 15:32)



So, what I do? We use 3 mod 10 counters; mod 10, mod 10, mod 10 we will here I am showing from the other side.

Suppose my input clock is coming like this mod 10 counter will be having 4 outputs suppose this is the most significant bit this is MSB. This MSB is connected to the clock of this is again MSB this is connected to the clock of this. So, if I have a circuit like this then this will be a modulo 1000 counter where your counting in decimal; that means, this is a you can say this is the BCD counter.

Because each of these stages will be counting from 0 to 9 and then again back to 0 0 to 9 0 to 9; so, this is a binary coded decimal BCD counter; this is not a binary 1000 this is BCD 1000 counter. Now if I want to display them in digit what I can do? You know BCD to 7 segment decoder we discussed earlier. So, you can use that kind of a decoder circuit here BCD to 7 segment decoder. So, we have this kind of a circuit and this decoder circuit you can use to drive some 7 segment display units. So, in this way you can have a 3 digit display; where it will be counting the number of clock pulses in decimal.

Right this kind of an application is quite common in many equipments we have this kind of circuit; internally they are implemented like this. There are a number of decade counters decade counters are actually counting in BCD essentially there cascaded together and you can directly drive the digits right; so this is one example. Let us take

another very interesting example which you see everywhere a digital clock; now in a digital clock first let us see what kind of output we get.

(Refer Slide Time: 18:24)

**Example: Design a digital clock**

|   |   |   |
|---|---|---|
| $\begin{array}{ c } \hline \square \\ \hline \square \\ \hline \end{array}$ | $\begin{array}{ c } \hline \square \\ \hline \square \\ \hline \end{array}$ | $\begin{array}{ c } \hline \square \\ \hline \square \\ \hline \end{array}$ |
| Hour  | Minute  | Second  |
| 0-23  | 0-59  | 0-59  |
| (mod-24)  | (mod-60)  | (mod-60)  |

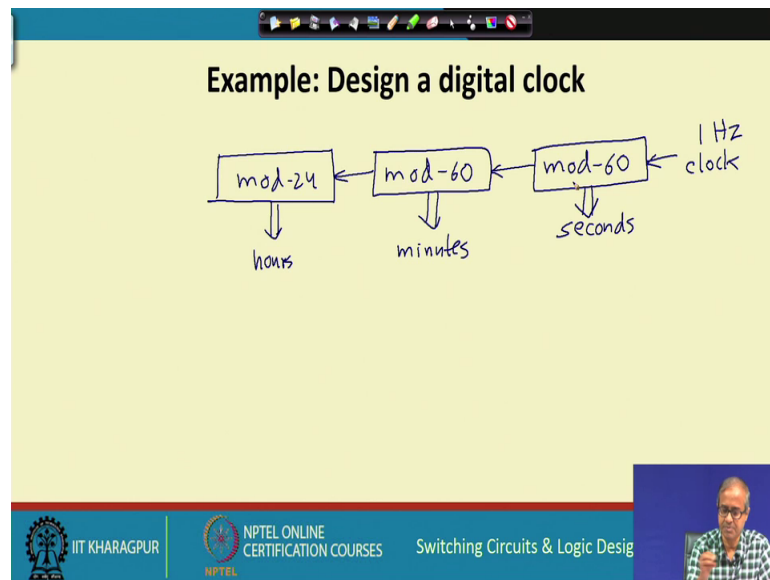
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

We have 2 digits that indicate our 2 digits that indicate minutes and again 2 digits that indicate seconds. But the point to notice that the way this counting goes on are not the same for the 3 different parts.

In second your counting from 0 to 59 and then back to 0 which means it is a mod 60 counter, for minute also your counting from 0 to 59; this is also a mod 60 counter. And hour let us assume that it is a 24 hour format; so it will count from 0 to 23; it will be a mod 24 counter. And again because the numbers are all displayed in decimal, they will have to be designed using decade counter or BCD counters only because we want the display in decimal not in binary; this is the basic requirement right.

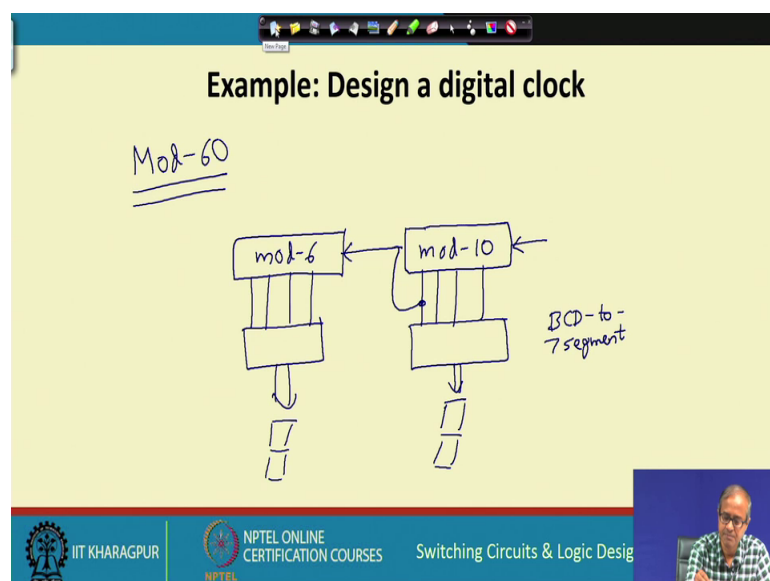


(Refer Slide Time: 19:58)



So, to do this what we have is that basically I will be using a cascade of 3 counters. As I said; so, again I am showing from the right side first modulo 60, then modulo 60, then modulo 24. Now in input I am applying a 1 hertz clock 1 hertz clock means 1 pulse every second; it will be counting the number of seconds. So, the this output will give you seconds, this output will give you minutes and this output will give you hours; this is the overall block diagram that it is. So, I need to design mod 60 counters 2 of them, I need to design a mod 24 counter one of this; let us see how I can design this counter separately.

(Refer Slide Time: 21:17)

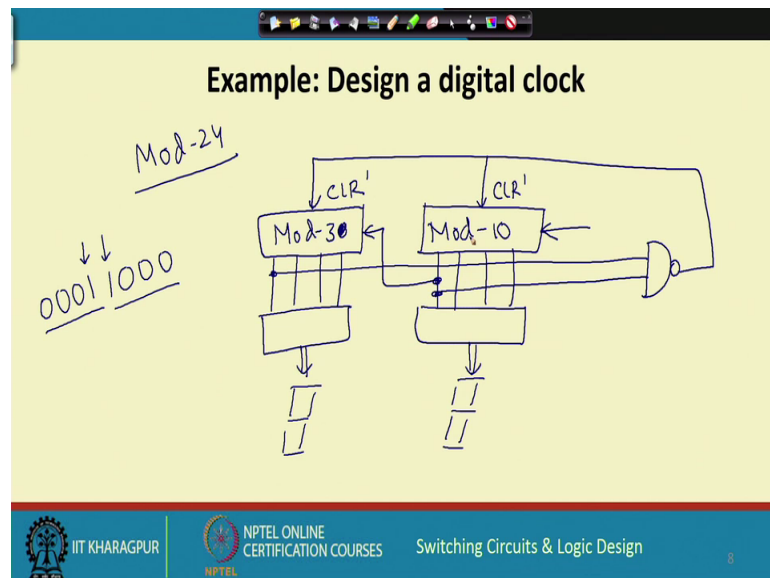


First we talk about modulo 60 counter design, because you are designing using BCD. So, what we do? You use a mod 10 counter and mod 6 counter you see I need 60. So, if you multiplied 10 by 6 it become 60. So, I said mod M and mod N counter if you cascade together it will become mod MN.

So, I require mod 60; I take a mod 10 I take a mod 6 because 10 into 6 is 60 this will serve my purpose. So, I connect them in cascade connect them in cascade in basically if this mod 10 has 4 outputs; the most significant bit is connected here. And mod 10 will be fed with the input clock, mod 6 will also be having 4 outputs and here I will be using this BCD to 7 segment decoder circuit.

And to the output of it 1 digit will come here and to the output of it 1 digital will come here. So, the mod 60 counter I can design like this mod 60 modulo. So, this will be a mod 60 counter; this will be a driven by these are BCD to 7 segment decoder driver. And finally, they will be driving the 7 segment displays right. So, I will be requiring 2 such not 60 counters then I need a mod 24 counter.

(Refer Slide Time: 23:11)



So, how do I design a mod 24 counter? Because you see mod 24 is an odd number I cannot generate 24 by multiplying 10 with something ok. So, I will take the next higher; so what do you do? I take a mod 10 counter you see I have to use mod 10 because I am counting decimal and next one I take mod 3. So, it becomes mod 30 mod 3 sorry mod 3

next higher and the input clock is coming here mod 10 in the most significant bit is fed as here and this will be mod 3; 4 bit output and generating.

Now, on this part this side it is similar BCD 7 segment decoder driver and 2 displays. Now the point to notice that I need mod 24 counter, but this is a mod 30 counter ok. Now 24 means what? If you write down the number 24 in binary; 24 means 16 plus 8 means 1 0 0 0 is 8 and 0 0 0 1 is 16 this is 24.

So, you see in 24 these 2 bits are 1; so again similarly I use a NAND gate; more significant bit of the first stage which is this; this I connect as one of the input and least significant bit of the next one, this one. And this I use to connect to the clear input of all the flip flops clear bar; clear bar.

So, whenever it becomes 24 I reset it to 0; so, this will be counting from 0 to 23 right. So, in this way I can design a mod 24 counter; so you see in this way I can design arbitrary counters and for many practical application as I have shown; I need to count in decimal so that I can display the results in decimal. I need decade counters are BCD counters they are also sometimes called for this purpose. So, I have taken 2 examples of a mod 1000 counter and also that our digital clock which is very common to illustrate this process.

So, with this we come to the end of our discussion on the registers and counters. Now I reemphasize registration counters are very important building blocks in system design; whenever we design complex systems you need to design registers, you need to use counters. Depending on the situation you will have design or register or a counter with a specific functionality; based on whatever we have discussed, I believe we will now been a position to design any kind of a register or counter as per the requirement or the need.

Thank you.