

Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture – 31
Sequential Circuits (Contd.)

(Refer Slide Time: 00:21)

SR to D

D Input	Outputs Qp+1	S-R Inputs	
		S	R
0	0	0	X
0	1	0	0
1	0	1	0
1	1	1	0

K-maps

S = D

K-maps

R = \overline{D}

Logic Diagram

Next we will look into conversion from SR flip flop to D flip flop. So, here also for the since finally, we want D flip flop. So, we write down the operation for D flip flop that is if the D input is 0 and previous output is 0, then it is output Q_{p+1} should be 0. So, like that now for S and R, we write down the corresponding values that should come like for 0 to 0 this S value should be equal to 0, R value can be dont care. Now, 1 to 0 S value should be 0 and R value should be 1 and from 0 to 1, S value should be equal to 1 and R value should be equal to 0.

So, we have to be careful that we do not put 1 1 to SR flip flop. That is why though I could have written an X here, but you do not do that because that will be putting on that, that leads the chance that will be putting 1 1 on to SR. Just to avoid that so, it is hard coded to 0. Similarly, from 0 to 1 so, this is hard coded to 1 0 and this 0 is not, dont care, and this 1 1 so, this is I can put this R value as 0 and S value as do not care.

So, if you do a simplification and then, we find that between S and R, we need to put an inverter, so S is equal to D and R equal to D bar. So by putting this inverter so we can get this SR flip flop converted to D flip flop.

(Refer Slide Time: 01:51)

D to SR

Conversion Table

S-R Inputs		Outputs		D Input
S	R	Q _p	Q _{p+1}	
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	Invalid		Don't care
1	1	Invalid		Don't care

K-map

	RQ _p		00	01	11	10
S	0	1	0	1	0	0
1	1	1	X	X		

$D = S + \bar{R}Q_p$

Logic Diagram

The vice versa then the other way D to SR conversion; so, this is also similar, like what we do. We first write down the table for this SR, and then, for 0 0 can Q p equal to 0, then Q p plus 1 is also 0 0 0, then Q p is 1, then the Q p plus 1 is 1. So, that way we make this table, but this 1 1, so this is invalid and then, because for SR we are not going to put 1 1 into it. So, that is kept as invalid.

Now, we consider what is the, what should be the value of D like from 0 to 0. So, D is 0 from 1 to 1, D is 1 from 0 to 0, D is 0. It goes like this, then for this invalid cases, since we are not going to apply this pattern, so they are taken as do not care. So, make it Karnaugh map and try to see what is the expression for D and the expression for D becomes S plus R bar into KM. Basically this is Q p somehow not written properly that is that SR R bar S plus R bar Q p. So, R is inverted Q p is taken, they are ended here and they are odd with S and that is fake to the D flip flop.

(Refer Slide Time: 03:16)

JK to T

Conversion Table

T Input	Outputs		J-K Inputs	
	Q_p	Q_{p+1}	J	K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

K-maps

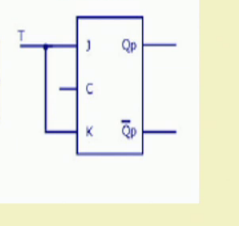
J=T


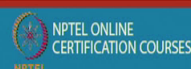
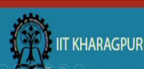
0	1
0	X

K=T

0	1
X	1

Logic Diagram





So, this way we can make conversion from D flip flop to SR flip flop. JK to T conversion, so, we already know that you for JK if you try this, J and K inputs together, so it will give me T flip flop. So, why does it happen like this? For T the behavior expected is if the T value is 0, output will not change. If the T value is 1 output will toggle, ok. So, for 0 if the previous value is 0, the next value is also 0. Similarly if the previous value is 1, the next value is 1. Whereas for either T input is 1 so, this Q value is if it was 0 now, it will become 1 and if it is 1 now, it will become 0. The T is 1 if Q_p is 1, then it will become 0.

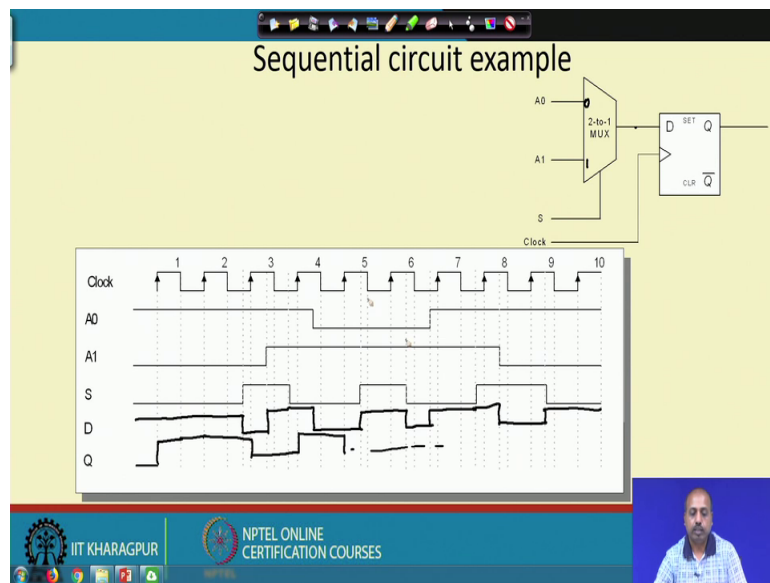
Now, for this J K what happens is that we have got this. So, for the JK so if you want to get this type of behavior, then from 0 0, this J input must be equal to 0, K input can be don't care for 1 1, K input must be equal to 0 because this J input being 1 will set it to 1, J input being 0, it will maintain the previous value.

So, both the cases, the value will be equal $Q_p + 1$ will be equal to 1. Similarly here if we are trying to get from 0 to 1, so J input must be made equal to 1, K input is don't care. So, if K input is 0 by virtue J equal to 1, so it will set the flip flop to 1. If K equal to 1 by virtue of toggling property of JK flip flop, it will toggle from 0 to 1. So, in either case, so this value will become equal to 1 and from 1 0 so, this J input can be don't care, where K input is equal to 1. So, it will be resetting the flip flop to 0. Now, if you draw the

Karnaugh map and then, we see that minimize it, then we transout that J equal to T and K equal to T.

So, both the cases so this T input should be connected to both J and K, so, that is the tying of this J and K inputs together and calling it as T input. So, this way we can convert 1 J K flip flop to T flip flop.

(Refer Slide Time: 05:29)



So, here is again another exercise that says that how can you, how this circuit will behave when you will be when you will set this behavior of this circuit. Now for that matter so you can do it like this. First of all, you can try to get what is the value of D here. So D, so your S signal is low here. So, as long as AC is let us say this is 0 and this is 1. So, when AC is low, A0 is selected. When AC is 1, A1 is selected.

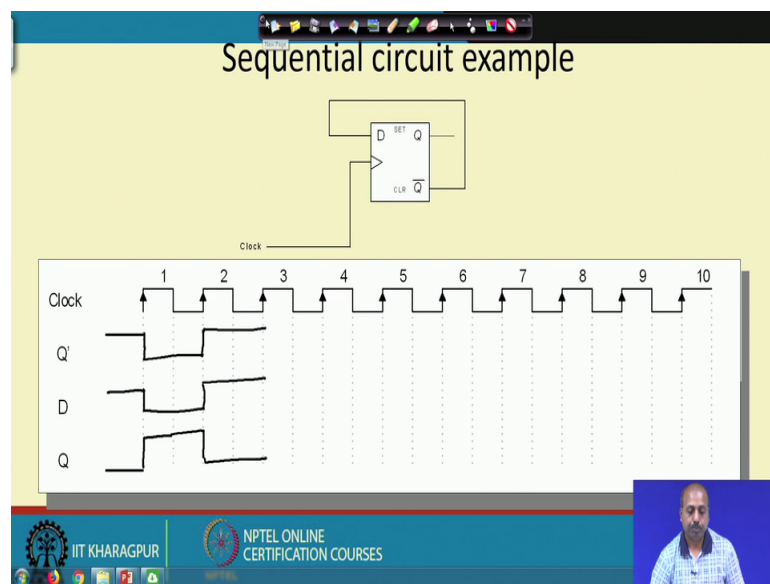
So, for this much time this D input is high and at this point, it is becoming low and then, it is becoming, it will remain low for this time. Then, A1 is becoming high, ok. So, it continues like this then AC is becoming low. So, A0 will get selected, but A0 is high here. So, it will continue like this and then, at this time it will be coming down and then, AC is high, AC is low for this much time. It will be like this then this A1 is high here. So, A1 is high here, A1 get selected and then, S is becoming low here and at this time you see that A0 is also low. So, A0 is low. So, it will come down and then, it will continue like this, then at this point it will be A0 is becoming high. So, it will become high. Now, again at this point AC is becoming high. So, A1 will get selected, A1 is high. So, at this

point it will come down and then, it will continue like this and then, AC is becoming low and A0 is high. So, it will go like this.

So, this is the behavior of D line. Now, from this behavior, you can very easily draw Q line behavior for the D flip flop. So, because at this point your this Q is high. So, this will be high and then, at this clock edge is basically you have to see. So, this is high then again at this clock edge it is low. So, it will be coming down here and then, again at this clock edge, it is high. So, it will be going high next clock edge, the signal is low, the signal D is low. So, this way you can draw the rest of the part.

So, for any sequential circuit tracing, first you determine the value at the input of the flip flops and then, based on the clock you just try to draw it. So, that is the technique for doing it.

(Refer Slide Time: 08:28)



So, this is another example where this Q bar line has been fed back to D. Now, see this. So, this clock is initially, so Q bar line is initially high. So, when Q line is initially low so, you can say that D line is initially high, ok. Now, since this is a D flip flop then this Q bar line is fed back. So, this Q value, so this is getting high at D. So, at the clock edge, this value will be going up then and as soon as this goes up, these goes down, ok. Then, it continues like this till the next clock.

Now, at the next clock what is the value of D? So, that is equal to the value of Q bar, ok. So, this whenever this Q bar has become low, this D value has also become low. So, at this point this Q bar value is low. So, D value is low and this Q will be coming down because it is coming as low here. So, it will be coming down and that it will continue for this much of time and this then this Q bar line so, that is the compliment of this thing Q will continue till this and D line since it is equal to Q bar, it will go like this.

So, this way you can draw the circuit, you can complete, you can draw the timing diagram and complete the portion. So, I am leaving it for you as a practice.

(Refer Slide Time: 10:08)

Registers

- A collection of 2 or more D flip flops with a common clock
- Registers are often used to store a collection of related bits (e.g. a byte of data in a computer)

Often not only the clock but also the other control signals (clear and preset) are shared

Next will be looking into another sequential element which is known as register, so far we have seen flip flop and in a flip flop, you can see that we can store only 1 bit of information. If the clock signal is not given, then if I have already D type of flip flop, then it will remember what was the last value of D given to it.

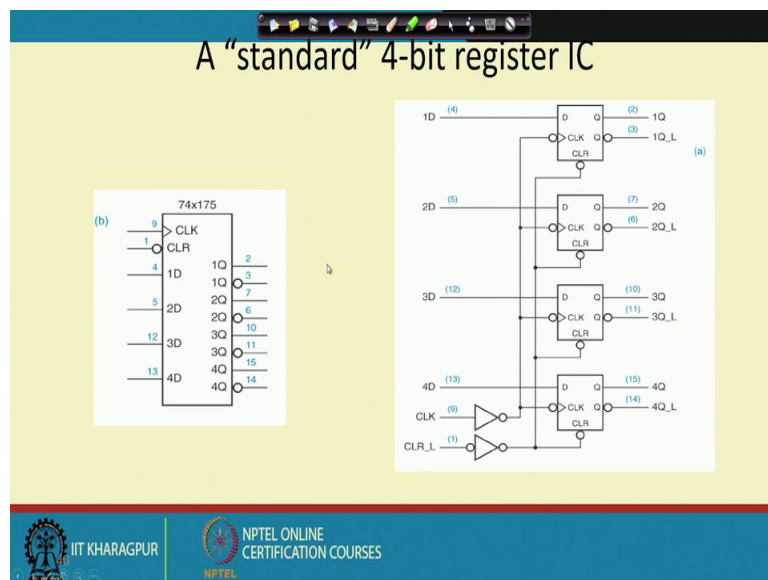
So, that is why it may be considered as 1 bit register. Now, if I want to remember multiple bits, instead of single bit if I want to remember multiple bits, then what is needed to be done is that we should have more than 1 D flip flop. We should have a collection of two or more D flip flops with a common clock and this type of structures, they are known as registers.

So, registers are often used to store a collection of related bits for example, a byte of data in a computer. So, like here this particular structure you see it, we have got 4 D flip flops and the clock signal is common to all of them. So, at the positive edge of the clock whatever value is coming on to IN1 will be stored into the first flip flop. Similarly, whatever coming IN2 will be coming in the second flip flop, IN3 will be coming the third flip flop and IN4 will come to the fourth flip flop.

So, I can say that on the positive edge of the clock the 4 bit pattern that I am giving here will be stored in this flip flops and they will be available on this outlines and also, there is a common clear line connected to all of them. All the flip flops they are connected using this clear line. If the clear is equal to clear is set or is active, then all the flip flops content will be cleared. There may be some preset control lines also that is not shown in this diagram. So, that may preset all the flip flops to 1. So, this type of structures are very common inside computers and they are known as the registers.

So, next we will be looking into this construction of the registers to using these flip flops.

(Refer Slide Time: 12:20)



So, this is a standard 4 bit register IC 74 175. So, there are 4 bit means, there should be four such flip flops inside and each flip flop has got 1D input and 1Q output 1Q bar output, fine. So, it is written as 1D 1Q 1Q and there is a bubble shown here. That means, the pin number 2 is Q line and pin number 3 is the Q bar line for flip flop 1.

Similarly, these 2D means this is the D line for the second flip flop pin number 5 and its outputs are 2Q and 2Q bar. So, it is second flip flops Q line is 7 and Q bar line is 6. So, this way this chip is there and this clear and clock. So, these are common for all the flip flops. If the clear signal is made 0, so all the Q values will be made equal to 0 and Q bar line will be made equal to 1 and then, this the clock signal is required and this symbol shows that it is a positive edge triggered flip flop.

So, internal diagram is like this. So, you have got this 4 flip flops, and this individual flip flops so, they are having this D flip this, D line clock line and the clear line and there are outputs like Q and Q bar. So, this way so this standard 4 bit register IC there. So, you can use it for design.

(Refer Slide Time: 13:56)

A "standard" 8-bit register IC

74x273	
11	CLK
1	CLR
3	1D
4	2D
7	3D
8	4D
13	5D
14	6D
17	7D
18	8D
2	1Q
5	2Q
6	3Q
9	4Q
12	5Q
15	6Q
16	7Q
19	8Q

$3 \times 8 = 24$
 $V_{cc} - 1$
 $GND - 1$
 $\frac{2}{28}$

(20)

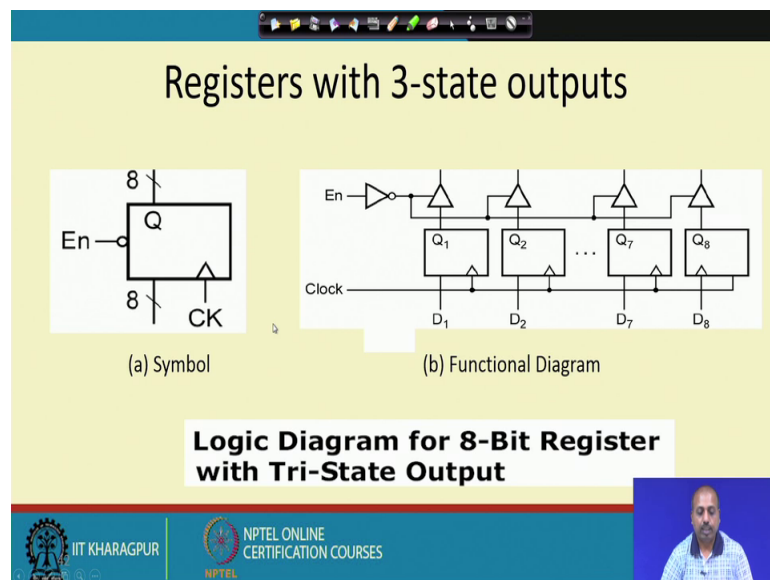
You can have 74 273, this is another IC, another integrated circuit chip where we have got this 8 such flip flops, so, 1D 2D 3D 4D and then this 1Q 2Q. So, interestingly you see that Q bar lines are not taken out ok.

So, Q bar lines are not taken out because for D flip flop. So, if you put both Q and Q bar line for 8D flip flops, so we will be needing how many pins? So, each flip flop requires 3 pins; 1D 1Q and 1Q bar. So, 3 into 8, so 24 pins will be needed there itself plus there is a supply voltage V cc. So, that is one pin, then the ground pin, ground lines that will be 1 pin and this clock and clear. So, there are 2 more pins. So, 24 plus 4, 28 pin, but this 74 273, it has got lesser than 24-28 pins. So, this you can see here up to a pin number 19.

So, 1 2 3 4 5 6 7 8 9 and 10 is not shown here, 10 is actually by the supply line, the V_{cc} line and there is another ground line.

So, total it will have 20 pins in it. So, to fit that design into 20 pins, what has been done, the Q bar lines have been cut. So, Q bar lines are not provided. So, if externally you really need Q bar line in your design, you can put an inverter and do it. So, what I mean is that in, so you have got this D line and this Q lines. So, Q bar pins are not taken out. So, if you need externally Q bar, so put an inverter here and take the Q bar from there ok, but in most of the cases, we do not need both Q and Q bar lines. So, that is why it is many chips will not have this explicit Q bar lines in them.

(Refer Slide Time: 16:00)



Some of the registers, they have got 3-state output. So, 3-state output means that the output can be 0. It can be 1 or it can be in a high impedance state that is neither 0 nor 1. So, it is like this. So, there is an enable line. So, if this enable line is active that is enable line is 0 then only this whatever be the state in the register so they will be available on this 8-bit line. So, there is 8-bit input and 8-bit outputs. So, whatever be the 8-bit value that you are giving here that is stored into this flip flop, but that is not immediately visible at the Q output for getting the output here so, you have to make enable equal to 0, then only they will be visible.

So, this type of gates, they are known as tri-state gates. So, this tri-state gates are enabled by means of this enable pin. So, if this enable line is 0, then this tri-state gates are

enabled. As a result this Q outputs are visible at this at the output of the tri-state gate. So, this helps us in connecting a number of register outputs together ok.

(Refer Slide Time: 17:14)

A "standard" 8-bit register with 3-state outputs

The slide illustrates the 74x374 8-bit register with 3-state outputs. It features a pinout diagram on the left, a circuit diagram in the center showing three registers (R1, R2, R3) connected to the OE pin, and a detailed internal logic diagram on the right showing the register structure with data inputs (D), clock inputs (CLK), and data outputs (Q) for each bit.

So, like this 74 374, so this is a chip where we have got two controls. One is clock that is the standard clock and there is another pin which is known as output enable or OE bar line.

(Refer Slide Time: 18:20)

Registers with clock enable

The slide shows the logic symbol for a register with a clock enable pin (Load) and a detailed circuit diagram of an 8-bit register with clock enable. The circuit diagram shows eight D-type flip-flops connected in parallel, each with its own data input (In0 to In7), data output (Out0 to Out7), and a common clock input (Clk). A clock enable pin (CE) is connected to the clock input of each flip-flop through an AND gate. A clear pin (ClrN) is also shown.

Clock enable is often called Load or "gated" clock

So, if you look into this logic diagram, it shows that the OE bar line. So, this is connected to this tri-state buffers. So, if this OE bar line is 0, then this will become equal to 1, as a result all these tri-state buffers will be enabled and the value that is available at Q will be coming to a pin number 2 which is corresponding to the 1 Q line. Similarly, this Q output will be available at pin number 5 which corresponds to the 2Q line.

So, this way we can have these 8-bit register with 3-state outputs, so once apart from this logic high and logic low output so, it has got high impedance state which is neither logic 0 or logic 1 and it is useful while you are connecting number of such devices together, like what I mean is that suppose I have got two registers like this and these two registers outputs, they are fed to another register. So, this output is one 8-bit output. So, this output is also another 8-bit output and both of them are feeding to this register. So, this is R1 and R2 and R3.

Now, if I do not have this type of tri stating facility, then one way to do it is that I have got this R1 R2 and then, I put some multiplexer type of structure, the 8-bit multiplexer where this two comes and then, they finally go to R3. All these lines are 8-bit that is these multiplexer is consisting of 8, such 2 2 1 multiplexers, fine.

Now, I have to have a select line by which I have to select which multiplexer output I want to go to R3, but if I have this type of tri-state facility, then what I can do, I have this R1 and R2 and their outputs are tri-stated and then I can if I enable this R1 register, then this if I enable, if I give enable signal to this OE signal output enable signal to this R1 register and it do not give V signal to R2 register, then R1 content will be coming here.

Similarly, if we do not give OE signal to this and give OE signal to R2 ok, then these value will coming through R3 like this. So, I do not need to put separate multiplexers if my registers are tri-state buffers or having tri state buffers in them. So, that is the outputs tri-stated. So, that is why in computer design, so while we have got a number of registers and their outputs are made to be common, so we take help of this tri state registers and then, there is a controller that we have in the computer system. So, that will be controlling this output enable lines and it will dedicate, it will determine like which registers output will be going to which register.

So, we do not need separate multiplexers for them that reduces the overhead. We can have registers with clock enable like so, we have got here this clock and clock enable is

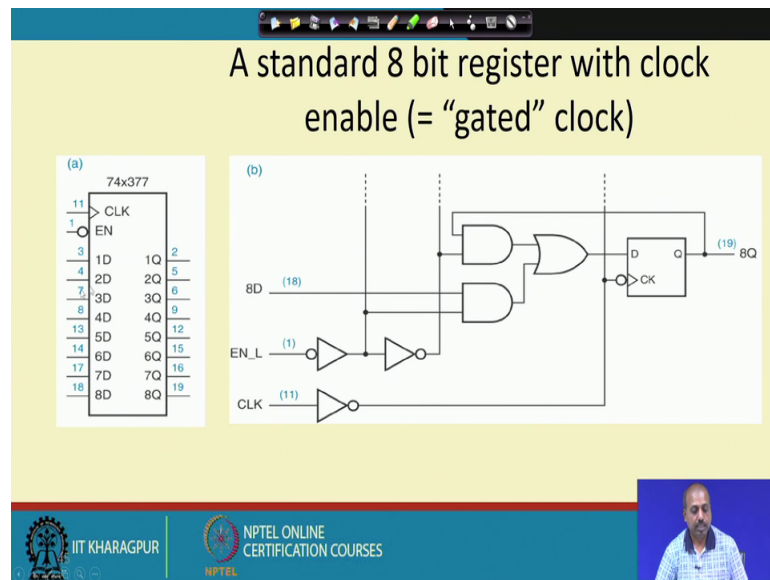
often called load or the gated clock. So, this load signal is given. So, here the clock signal is given, but until and unless this load signal is also given, the value that is coming on these D will not be stored into this register, ok. So, this design is shown here. So, you see if this clock enable is equal to 0, if the clock enable line is equal to 0, then whatever is output Out 0 so, that is circulated back into D flip flop.

So, for loading something on to this D flip flop, I must make this C line equal to 1 or the load line equal to 1, then what will happen if this line is equal to 1 then, this In0 will be coming here. So, this In0 will be, input 0 will be stored into this D flip flop. Similarly, this In1 will be coming on to this line and this will be stored into this D flip flop. I am sorry there is a shift in this 0's and 1's. So, this 0 is with this input and this 1 is with this input. They are actually identifying which input will be selected if C is equal to 0. So, C equal to 0, Out 0 is selected if C equal to 1 In0 is selected.

Similarly, if C equal to 0, Out 1 is selected in this case and In1 is selected if C equal to 1. So, this way apart from this clock and everything we can have another control which is the load control. So, this load control will tell when the value should be loaded into the register when the next clock comes, like if the load is low even if the clock signal is there, the value is not loaded and there is a clear of course that if clear line is 0, then the register will be cleared so, that is there and in this particular case we do not have that tri-state buffering shown here. There can be OE bar control also, output enable control also in the general case.

So, normally in a register we will have the clock input, the clear input and we will have this load input and OE bar, the output enable line. So, all those lines can be there.

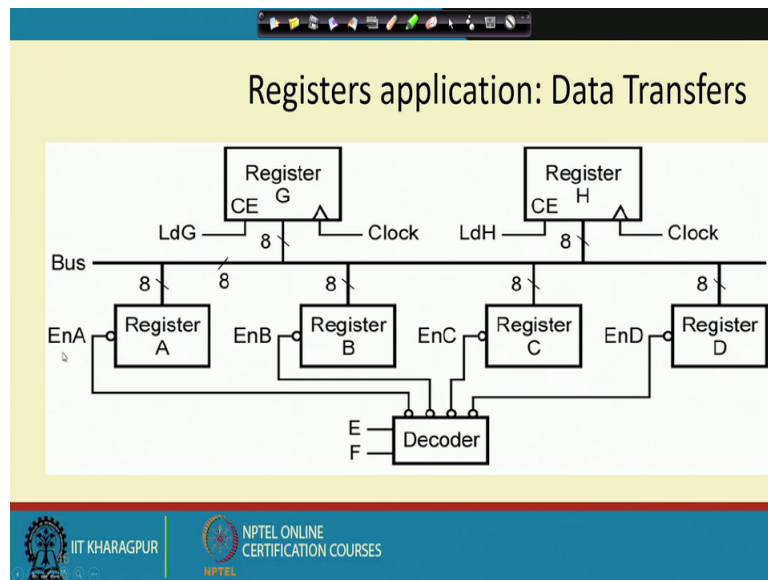
(Refer Slide Time: 23:04)



So, this is one chip 74 377. So, that has got that has an 8-bit register with clock enable. So, clock enable it is some sort of you can say it is a gated clock type of structure as if this clock signal is given, but if this enable line is low, if the enable line is high in that case, so this line, this point is 0. As a result this point is 1 and then this Q line will be circulated here and it will be put into this D flip flop into D input. Otherwise, if this enable line is low, so this is 1 so, this is 0 and as a result this output is 0, but this was equal to 1. So, whatever be this D input will be coming through this AND gate and through this OR gate till finally reach this D input and that will be stored into this D flip flop.

So, this shows the structure of these clock enable circuit tree. So, you can have this enable and this clock. So, this extra part that we have put here that is like a multiplexer type of structure that is useful for selecting whether I am going to circulate the previous value because enable line was not active or I am loading a new value because the enable line is active.

(Refer Slide Time: 24:27)



So, this shows a typical diagram like normally in application that include involve data transfer which is very common in case of computer systems or processors. So, we have got a bus and from this bus a number of registers are connected, ok, so this A B C D E F G and H. So, these are a few registers and each register, they are connected to the same bus, and what we want is that they should be able to communicate between themselves like one register content I should be able to transfer to some other register. Now, suppose I want to transfer the control of register A to register G, now how can I do this? So, for that purpose I have to enable these register output, and I have to enable the load signal for this G register. So, this enable A and load G, so these two inputs are to be given and accordingly it will be doing the transfer from A to G.

Similarly, if I want to make a transfer from A to H, then this load enable A should be given and this load H signal should be given. So, in a very simplistic situation, it may be the case that I have got these four registers A B C D from where we want to make transfer to this registers G and H. So, I can have some set of decoder where there where this E and F, they come as control and they will say from which register the transfer will take place and only one of this is a decoder only one of this outputs will be made equal to 0. So, as a result only one of these registers enable line will be 1 enable line will be active and these 8-bit will be coming from that register. Other registers those out all the outputs are tri-stated. So, they will not affect the bus.

Once the content is available in the bus, then it is assumed that there is some other controller it will provide this either this load G signal or load H signal to load the value on to the proper register. So, this way we can have these registers implemented register data transfer implemented by means of individual registers that has got enable and load lines in it.

(Refer Slide Time: 26:50)

The slide is titled "Shift Registers". It contains a bullet point: "It is a register that stores input values in sequence. At each clock tick the values stored are shifted from one flip flop to the adjacent". Below this, there are two diagrams. The left diagram shows a chain of three D flip-flops. The first flip-flop has a D input labeled "SERIN" and a Q output. Its Q output is connected to the D input of the second flip-flop. The second flip-flop's Q output is connected to the D input of the third flip-flop. The third flip-flop's Q output is labeled "SEROUT". All flip-flops share a common "CLOCK" input. The right diagram is a block diagram of a "Serial-In, Serial-Out Shift Register". It has a "SI (Serial In)" input on the left, a "SO (Serial out)" output on the right, and a "Clock" input at the bottom. The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a presenter in the bottom right corner.

Another class of registers, so they are known as shift registers. So, in shift register, it is a register that stores input values in sequence. So at each clock tick, the value stored at shifted from one flip flop to the adjacent flip flop. So, there is a serial input, so problem with this previous structure is that if you are trying to store some value on to this registers, so I have to give 8-bit parallel lines, but often getting so many parallel lines from somewhere is difficult. So, you want to transfer them, you want to transfer the data serially and that way we have got only.

So, these may be some higher number of, higher size register may be 8-bit register or 16-bit register or whatever, but I am not feeding all those register simultaneously with external data inputs, they are coming through this serial input line and this the serial input line so, it effects only the first D flip flop and this first D flip flop Q output is connected to D input of the next flip flop and it continues like that. This Q output is connected to the next flip flops D inputs. So, it continues like this. So, this way with the

when the clock pulse comes, then this serial input data gets loaded into D flip flop and the previous value of this D flip flop are transferred to this next flip flop.

So, this way this acts as a serial change. So, this type of registers, they are known as serial input, serial output register, shift register or SISO register serial input, serial output register and with the individual clock pulse, one data 1-bit data is shifted from this SI input into the first register and 1-bit of data from last register is shifted out. So, this is the serial input serial output type of shift register structure.