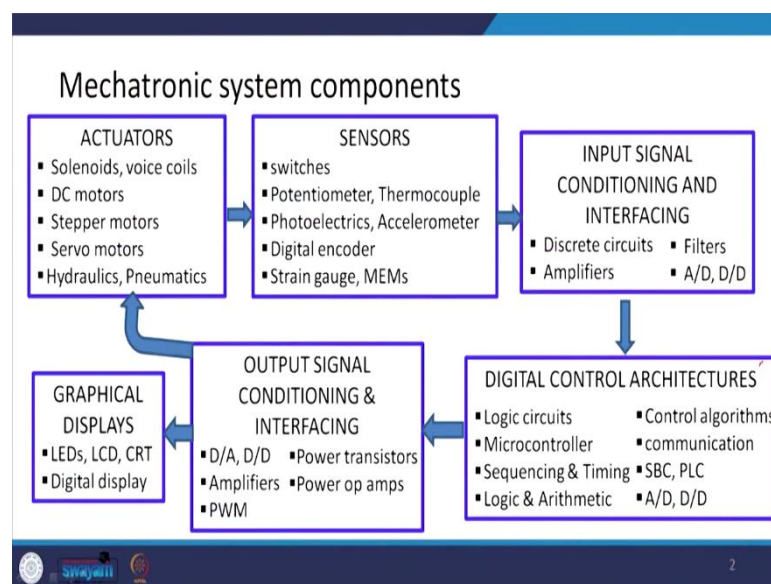


**Mechatronics**  
**Prof. Pushparaj Mani Pathak**  
**Department of Mechanical and Industrial Engineering**  
**Indian Institute of Technology, Roorkee**

**Lecture - 22**  
**Digital Circuits - II**

I welcome you all to today's NPTEL online certification course lecture on Mechatronics. Today we are going to talk about Digital Circuits. It is the second lecture. So, here, I will be talking about how to convert a truth table into a Boolean expression and the various types of memory devices such as flip-flops . So, various versions of it that I am going to talk about in this lecture.

(Refer Slide Time: 01:15)



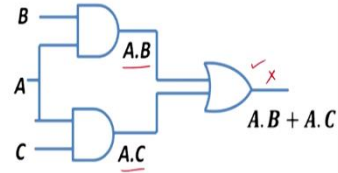
So, here in this particular portion is the digital control architecture of the mechatronic system components.

(Refer Slide Time: 01:23)

## Finding a Boolean Expression Given a Truth Table

### 1. Sum of Products ✓

- Represent an output as sum of products containing combinations of the inputs.



- For example, if we have inputs (A, B, and C) and an output (X), the sum of products would be  $X = (A.B) + (A.C)$

Furthermore, last in the last lecture, we have seen the various type of gates, and here, as I said, we are going to find out a Boolean expression given a truth table. So, two approaches are used. One is the sum of the product approach, and the other is the product of some approach.

So, in the sum of the project sum of the product approach, the output is represented as the sum of the product containing the combinations of the input. So, suppose we have the two gates the two AND gates here, and these two AND gates output are connected to an OR gate. So, input for the first one is A and B.

So, the output of that is A dot B. Similarly, for the second AND gate, the inputs are A and C. So, the output is A dot C, and when these two are passed as input to the OR gate, we get output,

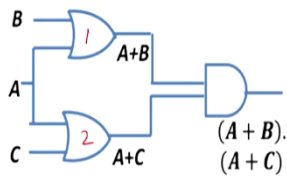
$$X = A.B + A.C$$

So, here as I said if we have input A B and C and output X, the sum of the product would be A into B plus A into C.

(Refer Slide Time: 03:05)

## 2. Product-of-Sums ✓

- Represent output as product of sums containing combinations of the inputs.
- For example, if we have three inputs (A, B, and C) and an output (X), the product of sums would be  $X = (A + B).(A + C)$  ✓



The 2nd method is what we call the product-of-sums method. So, here the inputs are given to the two OR gates here, and these outputs of the two OR gates are given as input to a AND gate. So, as you can see here A and B are the input to the first OR gate and so, the output is going to be A + B and for the second OR gate the inputs are A and C.

So, the output is going to be A+ C and when these two are sent as an input to a AND gate what we get is as (A + B) .( A+ C). So, it represents output as a product of sum containing combinations of the input.

(Refer Slide Time: 04:02)

### Example

- Steps
- Write expression for row having non zero output.
- Sum the expression of the non zero row.
- The sum of the products can be expressed as
- $Q = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C}$
- $= \bar{A}\bar{C}(\bar{B} + B) = \bar{A}\bar{C}$

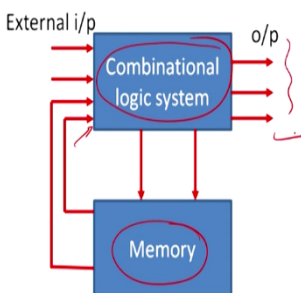
A	B	C	Output	Product
0	0	0	1	$\bar{A}\bar{B}\bar{C}$
0	0	1	0	
0	1	0	1	$\bar{A}B\bar{C}$
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	0	

Now, I can give an example. Suppose I have been given a truth table, and I am asked to write a Boolean expression for that. So, let us take an example. You have the three inputs A, B, C, and you have an output like this. So, the process is that what we do is that we write expressions for the row having a nonzero output. For example, nonzero outputs are there in these two rows. So, the expression for this in the product form through AND gate can be written as A invert, B invert, and C invert. When you take a product of that, you will be getting output as 1. Similarly, we see that we have input 0 and 1 and 0. So, for 0 for A and C, which has input 0, we take the invert like this. We have the B, which is already 1. So, B is written as it is. So, the first step is that sum the expression of the nonzero rows, and the sum of the products can be expressed here as you can see that A invert, B invert, C invert plus A invert B, and C invert. I can take A invert and C invert out. So, I get B invert plus B and which is going to be equal to A invert C invert. So, this way, I can convert this truth table into this Boolean expression.

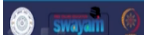
(Refer Slide Time: 05:51)

### Sequential Logic

- In combinational logic system the o/p is determined by combination of the i/p variables at a particular instant of time.
- When a system requires an o/p which depends on earlier values of the i/p a sequential logic system is required.
- Thus a sequential logic system requires a memory.



The diagram illustrates a sequential logic system. It consists of two main components: a 'Combinational logic system' and a 'Memory' block. On the left, three red arrows labeled 'External i/p' point into the 'Combinational logic system' box. From the right side of the 'Combinational logic system' box, three red arrows labeled 'o/p' point outwards. Below the 'Combinational logic system' box is a 'Memory' box. A red arrow points from the bottom of the 'Combinational logic system' box down to the top of the 'Memory' box. Another red arrow points from the top of the 'Memory' box up to the left side of the 'Combinational logic system' box, indicating a feedback loop where the system's previous state (stored in memory) influences its current output.


6

Next, let us look at the sequential logic. So, in the combinational logic system, which we have seen in the previous lecture, the output is determined by the combination of the input variables at a particular instant of time. So, that we have seen. I have taken an example also so that we have already seen it. Now, when a system requires an output, which depends on the earlier values of the input, a sequential logic system is required.

So, in this case, since the output depends on the earlier values of the input, we require a certain type of memory device, which means that the sequential logic system requires a memory. So, we have a combinational logic system here, and if this has a memory, a device is also there. So, the input from the memory device will be leading to the sequential logic system, and you will have the output from here.

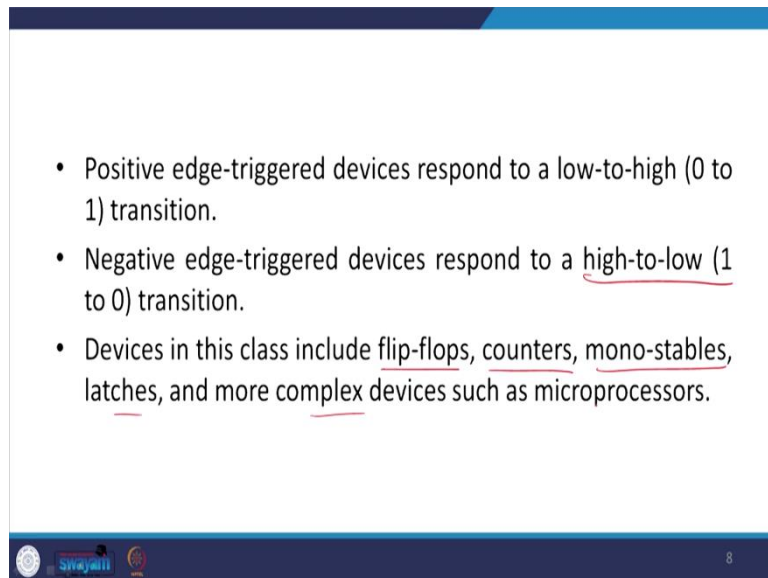
(Refer Slide Time: 07:14)

- Sequential logic devices usually respond to inputs when a separate trigger signal transitions from one level to another.
- The trigger signal is usually referred as the clock (CK) signal.
- The clock signal can be a periodic square wave or an aperiodic collection of pulses.

logic 1  
Level: high  
Positive edge ✓  
0  
Negative edge ✓  
Level: low  
Negative edge ✓  
Positive edge ✓  
Clock Pulse Edge

Now, sequential logic devices are usually a response to input when a separate trigger signal transitions from one level to another level. And this trigger signal, which we usually call a clock signal, is represented in a short CK that is a clock signal. The clock signal can be a periodic square wave or a periodic collection of pulses. These clock pulses could be a positive edge when it goes from 0 to 1 or low to high or a negative edge from 1 to 0. So, it could be this way, or it could be this way.

(Refer Slide Time: 08:09)



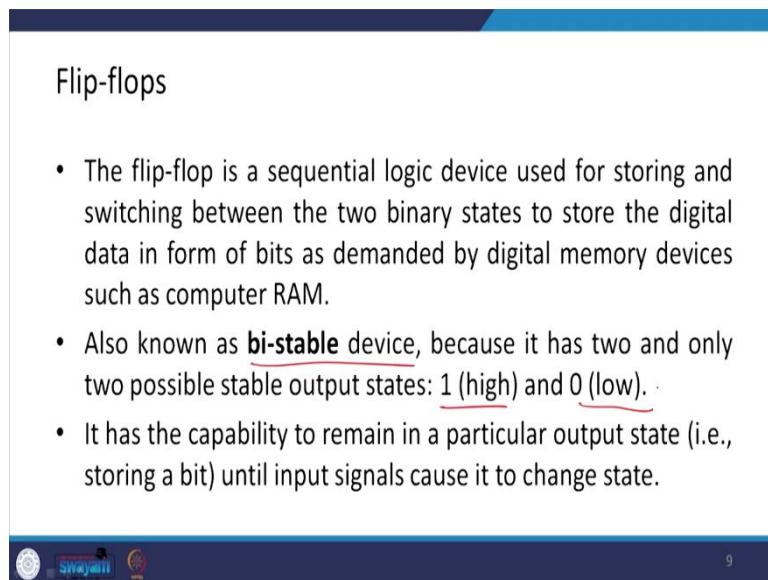
Slide 8 contains a bulleted list of characteristics for edge-triggered devices. The text is as follows:

- Positive edge-triggered devices respond to a low-to-high (0 to 1) transition.
- Negative edge-triggered devices respond to a high-to-low (1 to 0) transition.
- Devices in this class include flip-flops, counters, mono-stables, latches, and more complex devices such as microprocessors.

The slide footer includes a logo on the left, the text 'Swayam' in the center, and the number '8' on the right.

So, the positive edge-triggered devices response to low-to-high that is 0 to 1 transition, whereas negative edge-triggered devices respond to high to low that is 1 to 0 transition. Devices in this class include a flip-flop, counter, mono-stables, latches, and more complex devices such as microprocessors.

(Refer Slide Time: 08:39)



Slide 9 is titled 'Flip-flops' and contains a bulleted list of descriptions. The text is as follows:

### Flip-flops

- The flip-flop is a sequential logic device used for storing and switching between the two binary states to store the digital data in form of bits as demanded by digital memory devices such as computer RAM.
- Also known as bi-stable device, because it has two and only two possible stable output states: 1 (high) and 0 (low).
- It has the capability to remain in a particular output state (i.e., storing a bit) until input signals cause it to change state.

The slide footer includes a logo on the left, the text 'Swayam' in the center, and the number '9' on the right.

Now, let us look at the flip-flop. The flip-flop is the sequential logic device used for storing and switching between the two binary states to store the digital data in the form of bits as demanded by digital memory devices such as computers or RAM. It is also known as a bi-stable device because it has two and only two possible, stable output states that are high

and low. It has the capability to remain in a particular output state until the input signal causes it to change the state.

(Refer Slide Time: 09:31)

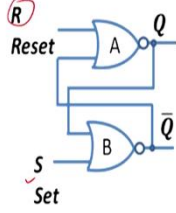
- It is a basic memory element which is made up of logic gates.
- There are number of forms say SR(set-Reset) flip flop involving NOR gates.
- Initially say  $Q = 0, \bar{Q} = 0, S = 0, R = 0$
- Next say  $S = 1, \text{so } \bar{Q} = 0, \checkmark$ 
  - so both i/p to NOR gate A are 0, so  $Q = 1 \checkmark$
  - So both i/p to NOR gate B are 1, so  $\bar{Q} = 0$ , and no more change.

So, as I said, it is a basic memory element that is made up of logic gates, and there is a number of forms S, R that is set, reset, flip-flop involving the NOR gates. So, here we can see that we have the NOR gate A and NOR gate B. From here is the set input, and from here is the reset input. And the other input to these gates is the output of the other gate. If these outputs are Q and Q bar, so, the input for A is going to be the Q bar, and similarly, input for B is going to be the Q. Now, initially Q is equal to 0, Q bar is 0, S is equal to 0 and R is equal to 0. Next, if we have S is equal to 1, so, if one this is 1, so, and this is 0. So, you are going to OR gate output to be one, and invert is going to be 0. So, the Q bar will be going to be 0. So, both inputs to the NOR gate A are 0.

So, this is already 0, and this Q bar is 0. So, here the NOR gate A output is going to be 1. So, you have Q is equal to 1. Now, both inputs to the NOR gate B are 1. So, this was already one, and now this is 1. So, you have both as 1. So, the NOR gate output is going to be 0. So, you have the Q bar is equal to 0 and no more changes. So, there is no more change by putting S equal to 1.

(Refer Slide Time: 11:35)

- Next say  $S = 0$ , so  $\bar{Q} = 0$ ,
  - There is no change in o/p state of  $\bar{Q}$  even when  $S$  changes from 1 to 0.
  - So it remembers the state it was set to.
- If  $R$  changes from 0 to 1, when  $S=0$ 
  - The o/p from NOR gate A changes to 0. ✓
  - And o/p for NOR gate B changes to 1. ✓
- So the flip-flop has been reset. ✓



11

Next, what I am doing is that I am putting  $S$  as equal to 0. Now, if this is 0 and so, this was 1, so, again, this invert is going to be 0. So, I have the  $Q$  bar is equal to 0. So, there is no change in the output state of the  $Q$  bar even when  $S$  changes from 1 to 0. So, what does this mean? This means that it remembers the state it was set to. Now, suppose, what we do? We change  $R$  from 0 to 1, and when  $S$  is equal to 0.

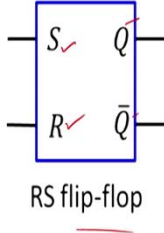
In the previous case, we have taken  $S$  is equal to 0. So, when  $S$  is equal to 0, suppose I am changing  $R$  from 0 to 1. So, what we have is that the output from NOR A, NOR gate A changes to 0 so, 1 and 0 so, its output is going to be a 0.

So, the output from the NOR gate changes to 0, and the output of NOR gate B changes to 1 because the 0 0 and its invert is going to be 1. So, the flip-flop has been reset. So, this is what it means that when you change the  $R$  to 1, the flip-flop has been reset as your  $Q$  bar is going to change to 1.

(Refer Slide Time: 13:20)



- Schematic representation of RS flip-flop is as shown in figure.
- The  $S$  is the set input,  $R$  is the reset input, and  $Q$  and  $\bar{Q}$  are the complementary outputs.
- Most flip-flops include both outputs, where one output is the inverse (NOT) of the other.



RS flip-flop

So, a schematic representation of RS flip-flop is as here. You can see that you have set, reset and output  $Q$ , and here you have the  $Q$  bar. So, here  $S$  is the set input,  $R$  is the reset input,  $Q$  and  $Q$  bar are what they are called complementary outputs. Most flip-flops include both outputs, where one output is the inverse of the other.

(Refer Slide Time: 13:52)

- Thus the RS flip-flop operates based on the following rules:

1. As long as the inputs  $S$  and  $R$  are both 0, the outputs of the flip-flop remain unchanged.
2. When  $S$  is 1 and  $R$  is 0, the flip-flop is set to  $Q = 1$  and  $\bar{Q} = 0$ .
3. When  $S$  is 0 and  $R$  is 1, the flip-flop is reset to  $Q = 0$  and  $\bar{Q} = 1$ .
4. It is "not allowed" (NA) to place a 1 on  $S$  and  $R$  simultaneously since the output will be unpredictable.

So, I can summarize. The RS flip-flop operates on the following rules. As long as the input  $S$  and  $R$  are both 0, the outputs of the flip-flop remain unchanged. And when a set is one and  $R$  is 0, the flip-flop is set to  $Q$  is equal to 1, and  $Q$  bar is equal to 0. And when  $S$  is 0 and  $R$  is 1, the flip-flop is reset to  $Q$  is equal to 0 and  $Q$  bar is equal to 1. And it is not

allowed to place as 1 on both S and R simultaneously since the output will not be predictable in that case.

(Refer Slide Time: 14:38)

Truth Table for the RS flip-flop

Inputs		Outputs	
S	R	Q	$\bar{Q}$
0	0	$Q_0$	$\bar{Q}_0$
1	0	1	0
0	1	0	1
1	1	NA	

So, here we have at S 0 0 we have some initial values, and with 1 0, this is 1 0, and with 0 1, this is 0 1 and 1 1; this is not defined as not allowed.

(Refer Slide Time: 14:56)

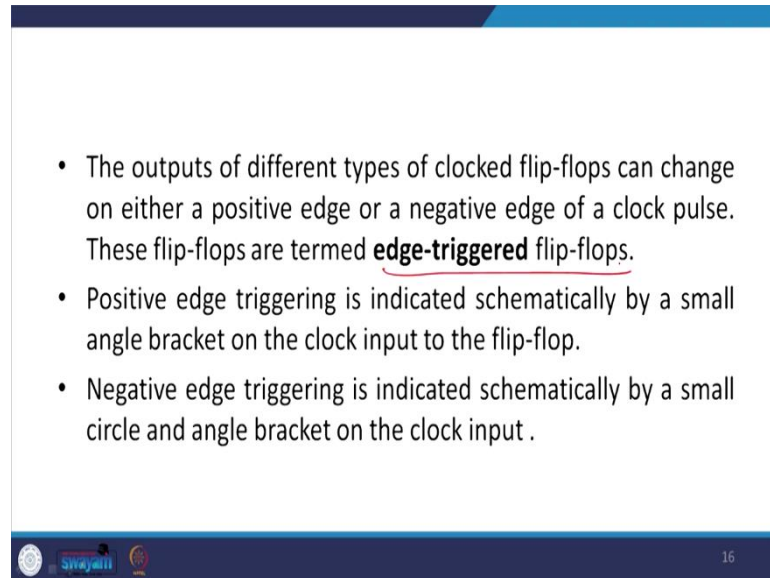
### Triggering of Flip-Flops

- Flip-flops are usually **clocked**; that is, a signal designated "clock" coordinates or synchronizes the changes of the output states of the device.
- This allows the design of complex circuits such as a microprocessor where all system changes are triggered by a common clock signal.
- This is called **synchronous** operation because changes in state are coordinated by the clock pulses.

Then triggering of the flip-flop: so, when should this change from S to R is to be done. This is what we call the triggering. So, flip-flops are usually clogged, that is, a signal designated clock coordinates or synchronizes the change of the output states of the device.

This allows the design of complex circuits, such as a microprocessor, when all system changes are triggered by a common clock signal. And this is what is called the synchronous operation because the change in states is coordinated by the clock pulses.

(Refer Slide Time: 15:46)



- The outputs of different types of clocked flip-flops can change on either a positive edge or a negative edge of a clock pulse. These flip-flops are termed edge-triggered flip-flops.
- Positive edge triggering is indicated schematically by a small angle bracket on the clock input to the flip-flop.
- Negative edge triggering is indicated schematically by a small circle and angle bracket on the clock input .

Now, the output of different types of clock flip-flops can change on either a positive edge or a negative edge of a clock pulse. These flip-flops are termed the edge-triggered flip-flops, and the positive edge-triggering is indicated schematically by a small. And angle bracket on the clock input to the flip-flop and negative edge-triggering is indicated simultaneously by a small circle and angle bracket on the clock input.

(Refer Slide Time: 16:25)

Positive edge-triggered RS flip-flops

Negative edge-triggered RS flip-flops

Positive edged-triggered RS flip-flop truth table ✓

S	R	CK	Q	$\bar{Q}$
0	0	↑	$Q_0$	$\bar{Q}_0$
1	0	↑	1	0
0	1	↑	0	1
1	1	↑	NA	
X	X	0,1↓	$Q_0$	$\bar{Q}_0$

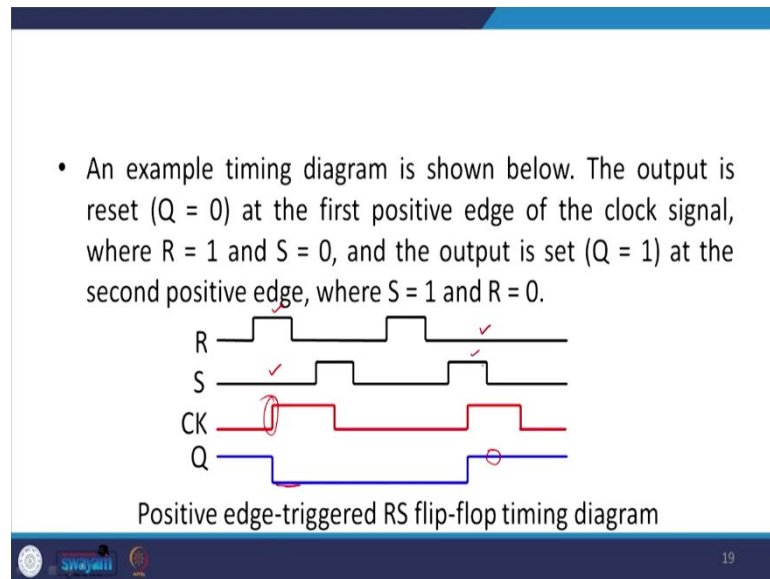
So, this is here how we represent the negative edge-triggered one, and this is how we represent the positive edge-triggered RS flip-flop. And the positive edge-triggered RS flip-flop truth table is going to be this one, as I discussed with you. Now, the up arrow here in the clock column represents the positive edge transition from 0 to 1, and the NA row indicates, as I said, the input condition that row is not allowed.

(Refer Slide Time: 17:05)

- The up-arrow ↑ in the clock (CK) column represents the positive edge transition from 0 to 1.
- The NA row indicates that the input condition for that row is not allowed.
- As long as there is no positive edge transition, the values of S and R have no effect on the output as shown by the X symbols in the last row of the table.

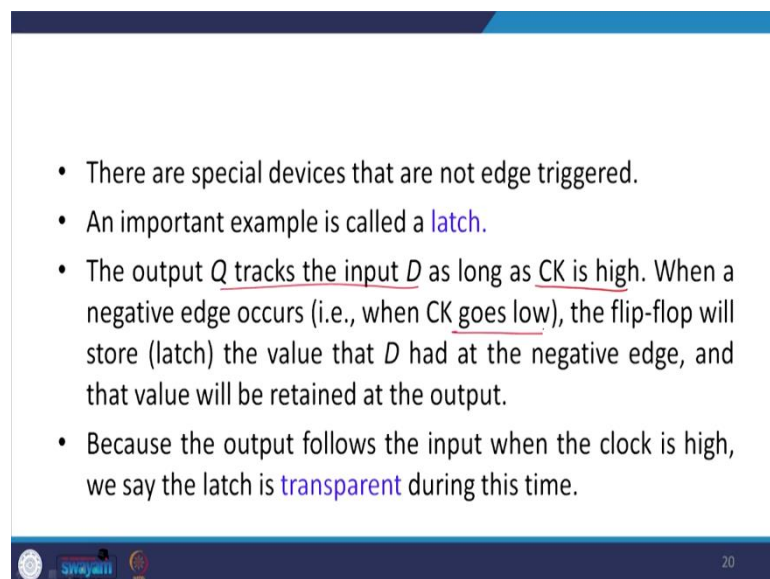
So, as long as there is no positive edge transition, the values of S and R have no effect on the output, as shown by symbol X in the last row of the table.

(Refer Slide Time: 17:14)



An example timing diagram is shown here below. So, the output is reset that is  $Q$  is equal to 0 at the first positive edge of the clock signal. So, where  $R$  is equal to 1 and  $S$  is equal to 0. And at the output set,  $Q$  is equal to 1 at the second positive edge where  $S$  is equal to 1 and  $R$  is equal to 0. So, here you can see that at this location, the  $Q$  value is 0, and here  $R$  is equal to 1 and  $S$  is equal to 0, and you have the clock signal over here. Similarly, we can set see that the  $Q$  value is one see over here at the second positive edge that is at this place, where  $S$  is equal to 1 and  $R$  is equal to 0 in this zone.

(Refer Slide Time: 18:37)



There are special devices that are not edge-triggered, and an important example in this is what we call a latch. Now, the output  $Q$  tracks the input  $D$  as long as the clock is high.

And when a negative edge occurs, that is when the clock goes low, the flip-flop will store the value that the D had at the negative edge, and that value will be retained by the output. And because the output follows the input when the clock is high, we that the latch is transparent during this time.

(Refer Slide Time: 19:28)

- The output (Q) tracks the input (D) while the clock level is high (CK = 1). The X in the last row of the table indicates that the value of D has no effect on the output as long as CK is low.

D	CK	Q	$\bar{Q}$
0	1	0	1
1	1	1	0
X	0	$Q_o$	$\bar{Q}_o$

Latch truth table

So, you can see that you have 0 1 here, and you have 0 1 here. And so, here, when this is 0, this is 0, and when this is 1, this is 1, and these are the clock. So, this is the latch, this is the clock, your output QQ bar, and this is the input signal. Here we can see that the output Q tracks the input D while the clock level is high; that is, CK is equal to 1. So, when the CK is equal to 1, your output here tracks the input, so that is there. The X in the last row of the table indicates that the values of D have no effect on the input as long as CK is low.

(Refer Slide Time: 20:27)

## Asynchronous Inputs

- Flip-flops may have preset and clear functions that instantaneously override any other inputs. They are called asynchronous inputs, because their effect may be asserted at any time.
- They are not triggered by a clock signal.
- The preset input is used to set or initialize the output  $Q$  of the flip-flop to high (1).
- The clear input is used to clear or reset the output  $Q$  of the flip-flop to low (0).



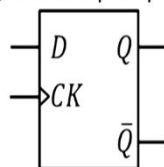
22

Next, let us look at isochronous inputs that are one, which is not clocked. So, flip-flops are not triggered by a clock signal. So, flip-flop may have pre-set and clear functions that instantaneously overrides any other input. They are called asynchronous input because their effectiveness may be asserted at any time, and as I said, they are not triggered by a clock signal. So, the preset input is used to set or initialize the output  $Q$  of the flip-flop to high, and a clear input is used to clear or reset the output of the flip-flop to a low.

(Refer Slide Time: 21:18)

## D Flip-Flop

- The **D flip-flop**, also called a data flip-flop, has a single input  $D$  whose value is stored and presented at the output  $Q$  at the edge of a clock pulse.
- A positive edge-triggered D flip-flop shown below.



Positive edge triggered D flip-flop



23

So, here we have a D flip-flop. The D flip-flop, also called a data flip-flop, has a single input  $D$  whose value is stored and presented at the output  $Q$  at the edge of a clock pulse. A positive edge-triggered D flip-flop is shown over here.

(Refer Slide Time: 21:50)

- Unlike a latch, a D flip-flop does not exhibit transparency. The output changes only when triggered by the appropriate clock edge (in this case, a positive edge).

D	CK	Q	$\bar{Q}$
0	↑	0	1
1	↑	1	0
X	0	$Q_o$	$\bar{Q}_o$
X	1	$Q_o$	$\bar{Q}_o$

Positive edge triggered D flip-flop truth table

Now, unlike a latch, the D flip-flop does not exhibit transparency. So, the output changes only when triggered by the appropriate clock edges and here is the truth table for the D flip-flop.

(Refer Slide Time: 22:16)

### JK Flip-Flop

- The JK flip-flop is similar to the RS flip-flop where the *J* is analogous to the S (set) input and the *K* is analogous to the R (reset) input.
- The major difference is that the *J* and *K* inputs may both be high simultaneously.
- This state causes the output to toggle, which means the output changes value (i.e., a 1 would become 0, and a 0 would become 1).

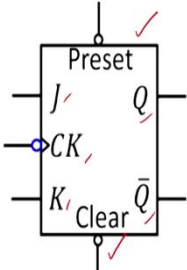
Next, let us look at the JK flip-flop. The JK flip-flop is very similar to the RS flip-flop, which we have seen where *J* is analogous to the set input and *K* is analogous to the R or the reset input. Now, the major difference is that the *J* and *K* inputs may both be high simultaneously.



We have seen that in the case of RS flip-flop, the R and S value both one way are not allowed. But, in the case of the JK flip-flop, these values may be high simultaneously; that is, both the J and K values can be one together. And this state causes the output to toggle, which means that the output changes values that are 1 would become 0 and 0 will become 1.

(Refer Slide Time: 23:21)

Truth table of negative edge-triggered JK flip-flop



Preset	$\overline{\text{Clear}}$	CK	J	K	Q	$\overline{Q}$
0	1	X	X	X	1	0
1	0	X	X	X	0	1
0	0			NA		
1	1	↓	0	0	$Q_o$	$\overline{Q}_o$
1	1	↓	1	0	1	0
1	1	↓	0	1	0	1
1	1	↓	1	1	$\overline{Q}_o$	$Q_o$
1	1	0,1	X	X	$Q_o$	$\overline{Q}_o$

Negative edge triggered JK flip-flop

So, this is how JK flip-flop looks like. You have a preset terminal, clear terminal, J, K, and here is the clock. So, you have output Q, and Q bar, and this table explains to you the truth table of a negative edge-triggered JK flip-flop.

(Refer Slide Time: 23:44)

- The first two rows of the table describe the preset or clear functions that can be used to initialize the output of the flip-flop.
- The third row precludes presetting and clearing simultaneously.
- The symbol  $\downarrow$  represents the negative edge of the clock signal, which causes the change in the output.
- The last row describes the memory feature of the flip-flop in the absence of a negative edge.

Now, here the first two rows of the table describe the preset or clear function that can be used to initialize the output of the flip-flop. And the third row precludes presetting and clearing simultaneously, and the down arrow represents the negative edge of the clock signal, which causes the change in the output. The last row describes the memory feature of the flip-flop in the absence of a negative edge.

(Refer Slide Time: 24:20)

### Applications of Flip-flops

- Switch De-bouncing. ✓
- Data Register. ✓
- Binary Counter and Frequency Divider. ✓
- Serial and Parallel Interfaces. ✓
- Bounce elimination switch. ✓
- Data storage. ✓
- Data transfer. ✓
- Latch. ✓

Now, coming back to the application of flip-flops. So, these flip-flops can be used for situations, such as switch de-bouncing, which is very useful. Then data register, binary counter and frequency divider, serial and parallel interfaces, and bounce elimination switch, data storage, data transfer, and latch.

(Refer Slide Time: 24:53)

Special Purpose Digital Integrated Circuits

- Decade Counter. ✓
- Schmitt trigger. ✓
- 555 timer ICs. ✓

29

There is a special-purpose digital integrated circuit for such as decade counter, Schmitt trigger, and 555 timer ICs.

(Refer Slide Time: 25:09)

Decade Counter

- Like flip –flop circuits, decade counter is another common counter which can be used to perform binary counting.
- It is a negative edge-triggered counter.
- The output is in binary coded decimal (BCD) consisting of 4 bits, making it useful for decimal counting applications.
- BCD counters can be cascaded to count in powers of 10.
- Cascading the two BCD raise the range for counting from 0 to 99. Further cascading allows counting of higher powers of 10.

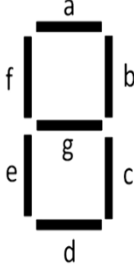
30

So, we will just discuss this very briefly about this. So, the decade counter is like a flip-flop circuit. A decade counter is another common counter, which can be used to perform binary counting. It is a negative edge-triggered counter. The output is a Binary Coded Decimal that is BCD consisting of 4 bits, making it useful for decimal counting applications. And the binary coded decimal counters can be cascaded to count in the power

of 10s. And cascading that two BCDs raise the range for counting down from 0 to 99, and further cascading allows the counting of higher powers of 10.

(Refer Slide Time: 25:57)

- A convenient device for viewing a BCD output is a seven-segment LED display driven by a 7447 BCD-to-seven-segment decoder.



Seven-segment LED display

The diagram shows a square-shaped seven-segment LED display. The segments are labeled as follows: 'a' is the top horizontal segment, 'b' is the top-right vertical segment, 'c' is the bottom-right vertical segment, 'd' is the bottom horizontal segment, 'e' is the bottom-left vertical segment, 'f' is the top-left vertical segment, and 'g' is the middle horizontal segment.

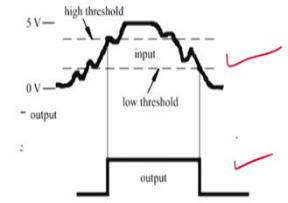
31

So, a convenient device for viewing a BCD output is a seven-segment display driven by a 7447 BCD-to-seven-segment decoder. The details about this I have discussed in another lecture on the display.

(Refer Slide Time: 26:17)

### Schmitt Trigger

- The Schmitt trigger is a device that can convert such a “noisy” (jumpy) fashion signal into a sharp pulse using the threshold hysteresis effect.



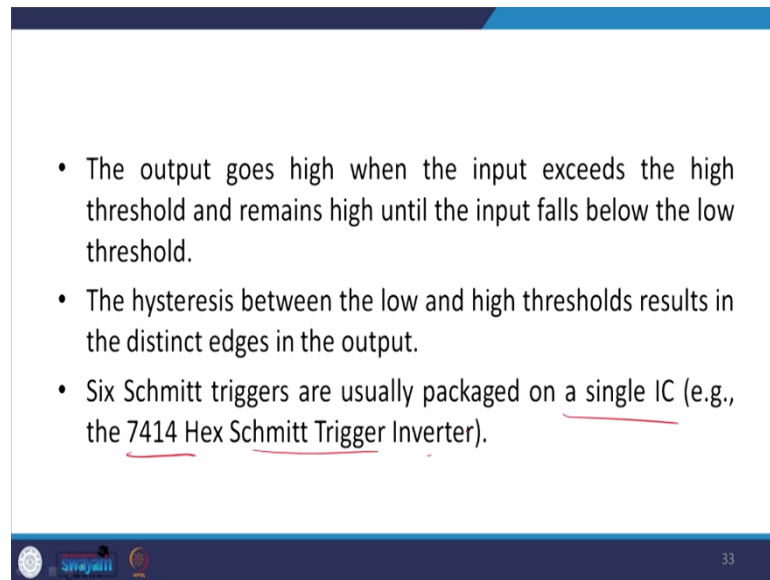
Input and output of a Schmitt trigger

The graph plots voltage against time. The input signal is a noisy square wave fluctuating around 0V and 5V. Two horizontal dashed lines represent the high and low thresholds. The output signal is a clean square wave that transitions from high to low when the input crosses the high threshold and from low to high when the input crosses the low threshold. Red checkmarks are present on the right side of the graph.

32

Schmitt trigger is the device that can convert a noisy or jumpy fashion signal into a sharp pulse using the threshold hysteresis effect. So, you can see that these noisy signals can be converted into a sharp square pulse.

(Refer Slide Time: 26:46)



- The output goes high when the input exceeds the high threshold and remains high until the input falls below the low threshold.
- The hysteresis between the low and high thresholds results in the distinct edges in the output.
- Six Schmitt triggers are usually packaged on a single IC (e.g., the 7414 Hex Schmitt Trigger Inverter).

The output goes high when the input exceeds the high threshold and remains high until the input falls below the low threshold that we can see from the figure—the hysteresis between the low and high threshold results in the distinct edges in the output. And the six Schmitt triggers are usually packaged on a single IC and the, for example, 7414 Hex Schmitt trigger inverter.

(Refer Slide Time: 27:18)

## 555 Timer

- The 555 integrated circuit is known as the “time machine” because it performs a wide variety of timing tasks.
- It is a combination of digital and analog circuits.

555 pin-out

Now, let us look at the 555 timer. The triple 5 IC is known as the time machine because it performs a wide variety of timing tasks, and it is a combination of digital and analog circuits. So, you are here seeing the two types of packaging. So, that is there, the FK package is there, and this is the 555 pinout.

(Refer Slide Time: 27:50)

## Applications for the 555 include


- Bounce-free Switches ✓
- Cascaded Timers ✓
- Frequency Dividers ✓
- Voltage-controlled Oscillators ✓
- Pulse Generators ✓
- LED Flashers ✓

Application of this 555 includes bounce-free switches, cascade timers, frequency dividers, voltage control oscillators, pulse generators, and LED flashers.

(Refer Slide Time: 28:05)

## Application of Logic Gates

- Parity generator. ✓
- Digital comparator. ✓
- Code converter. ✓




Coming back to the logic gates, the logic gates find very wide application in these three purposes. One is the parity generator, what are parity I am going to talk about, and the digital comparators. We have seen mostly using the feedback circuits and the code converter that is for the conversion of code from one system to another one.

(Refer Slide Time: 28:40)

## Parity generator ✓

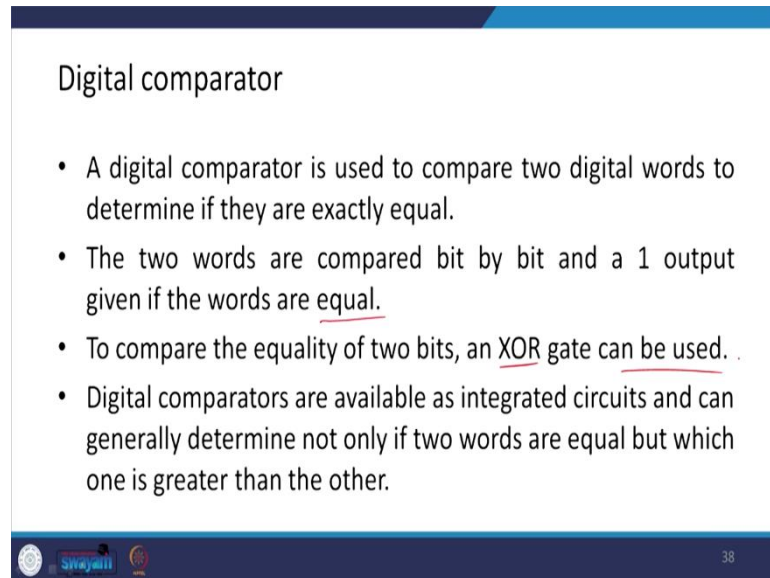
- What is parity? ✓
- When data is transmitted, because of the noise in the transmission, it may happen that 1 may be received as 0 or vice versa.
- To overcome this an extra bit is added. This may be 0 or 1, called parity bit.



So, as I said, parity generator. So, first of all, let us see what parity is. So, when the data is transmitted, because of noise in the transmission, it may happen that 1 may be received at 0 or vice versa 0 may be received as 1. So, to overcome this, an extra bit is added, and this may be 0 or 1 and what we call the parity bit. And then what do we do? By adding then, we count that in the input side how many 1s were there and on the output side whether the

same number of 1s are there or not. So, that is how we ensure that the same data has been transmitted.

(Refer Slide Time: 29:07)



**Digital comparator**

- A digital comparator is used to compare two digital words to determine if they are exactly equal.
- The two words are compared bit by bit and a 1 output given if the words are equal.
- To compare the equality of two bits, an XOR gate can be used.
- Digital comparators are available as integrated circuits and can generally determine not only if two words are equal but which one is greater than the other.

38

Next is the digital comparator. A digital comparator is used to compare two digital words to determine if they are exactly equal. And the two words are compared bit by bit, and a 1 output is given if the words are equal. To compare the equality of two bits, an XOR or exclusive OR gate can be used. Digital comparators are available as ICs and can generally determine not only if the words are equal, but which one is greater than the other that can also be found out by the digital comparator. Next, let us look at the code converter.

(Refer Slide Time: 30:29)



## Code converter

- Used to change data from one type of code to another.
- For example, the output from a microprocessor system might be BCD (binary-coded decimal) and need to be transformed into a suitable code to drive a seven-segment display.
- Data decoding is used for the process of converting some code group, e.g. BCD, binary, hex, into an individual active output representing that group.
- A 3-line-to-8-line decoder is used where a decoder has three input lines and eight output lines.

And as I said, this is used to change data from one type of code to another type. For example, the output from a microprocessor system might be a BCD that is binary-coded decimal, and this needs to be transformed into a suitable code to drive a seven-segment display. The data decoding is used for the process of converting some code group that is BCD, binary, hex, into an individual active output representing that group. A 3-line-to-8-line decoder is used, where a decoder has three input lines and eight output lines.

(Refer Slide Time: 31:14)

## References

- W. Bolton, Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering (6th Edition), Pearson, 2015
- R. Merzouki, A. K. Samantaray, P. M. Pathak, B. Ould Bouamama, Intelligent Mechatronic Systems: Modeling, Control and Diagnosis, ISBN 978-1-4471-4627-8, 2013, Springer, London
- D.G. Alciatore and Michael B. Hiestand, Introduction to Mechatronics, Tata Mc Graw Hill, 2012.

So, these are the references. If you wish to read further about digital logic, digital devices, the combinational as well as the sequential one, you can go through these books.

Thank you.