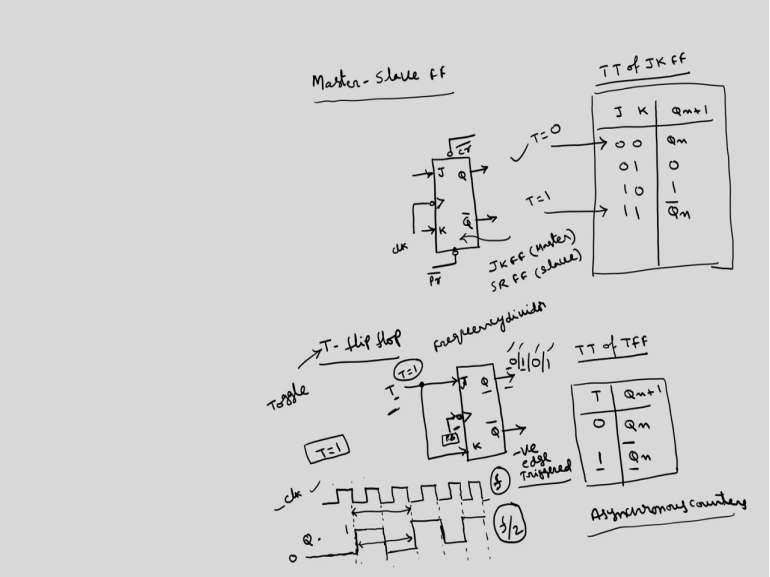
System Design through VERILOG
Prof. Shaik Rafi Ahmed
Department of Electrical and Electronics Engineering
Indian Institute of Technology, Guwahati

Gate level modeling - II
Lecture - 10
Verilog modeling of flip-flops

In the last lecture, we discussed about the SR flip flop. So, the problem associated with the SR flip flop, i.e. S is equal to R is equal to 1 is not allowed. So, to avoid this condition, we discussed about the JK flip flop. How this 1 1 condition can be allowed in JK flip flop. After that again in JK flip flop also there is a problem called race round condition. So, to avoid this race round condition we discussed about the master slave flip flop.

(Refer Slide Time: 01:17)



So, if I take this master slave flip flop as a block, the overall inputs of this master slave flip flops are J, K, Q, Q complement, then we have clock. So, if we use this bubble it is a negative edge triggered. Then in addition to this, we have two more signals called preset bar and clear bar. So, you have discussed the operation of this preset bar and clear bar. In order to establish the initial state of the flip flop, we can use the preset and the clear inputs.

So, they are called as asynchronous inputs because before application of the clock itself we can establish some initial state. So, inside this master slave JK flip flop, we know that we

have one JK flip flop which will act as a master and another SR flip flop which will act as a slave.

Now, using this master slave JK flip flop, we can derive two more flip flops one is called T flip flop; T stands for toggle. So, T flip flop can be obtained by just shorting this J and K terminal together, this is clock. If I connect this J and K terminals together, this is J, this is K. So, that common point you can call as T, Q, Q complement. This is the block diagram of T flip flop.

So, we know that the truth table of JK flip flop is J, K, Q n plus 1. This is actually simplified one 0 0 Q n itself. 01 0, 10 1, 11 Q n complement, but what will be the truth table of this T flip flop? This is truth table of JK flip flop. Among these four conditions only two conditions are allowed because J and K should be equal. So, only this condition is allowed, this condition is allowed.

So, this condition is corresponding to T is equal to 0, this is corresponding to T is equal to 1. So, the simplified truth table of this one is T, Q n plus 1; T is equal to 0 this is corresponding to this. So, simply Q n plus 1 is Q n, T is equal to 1, Q n complement. This is the simplified truth table of T flip flop. T stands for toggle. Why the name toggle? If I fix T is equal to 1, say initial state is 0.

So, if I apply one clock signal here then after the first clock signal, this is becomes 1 because when T is equal to 1 always the output will be complement of the input, complement of the previous output. So, previous output of 0 now, it will be 1. So, after another clock signal 0, after another clock signal 1 and so on. If you connect this clock to a push button, if you go on pressing this push button, the output will change 0, 1, 0, 1, 0, 1 and so on.

So, when does it change? If I assume that say for example, initially Q is equal to 0 and this is clock signal, say this is clock signal and I am assuming that here I have given a bubble. So, this is negative edge triggered. So, initially before the application of this clock signal, negative edge is this; this is the first negative edge. This is the second negative edge; this is third negative edge and so on.

I am fixing T is equal to 1 here. If initial state of this Q is 0, it will come with 0 from the infinity, this is logic 0. How to establish this initial state before the application of the clock signal? We can use preset and clear signals that we have discussed in the last lecture.

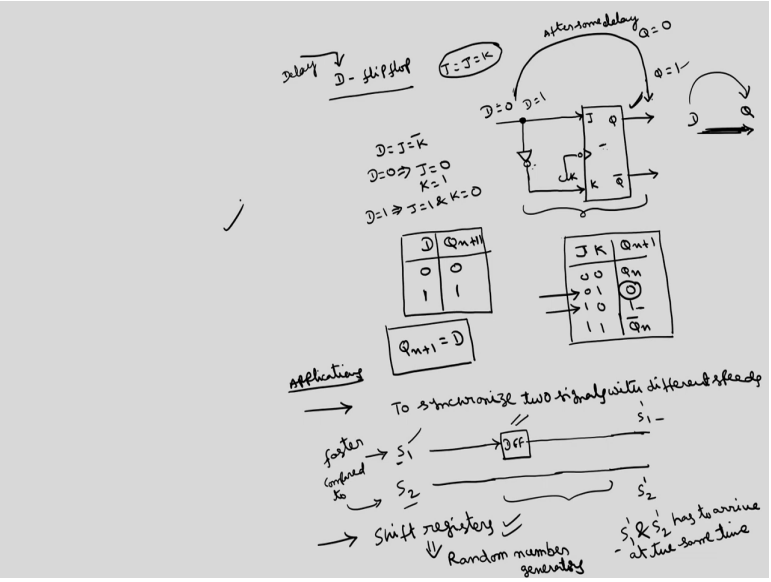
Now, at negative edge this 0 will toggle to 1 and this 1 will continue until the next negative edge. At this negative edge, this again becomes 0; again at the next clock negative clock edge, this will be 1 and the next negative edge 0, next negative edge 1 and so on. So, this is the waveform at Q with respect to this clock signal. So, that is why the name toggle.

At every clock signal, it will toggle. Because of this negative edge triggered, at negative edge, it will toggle. If I design a positive edge triggered, at the positive edge, it will trigger and so on. One important property you can see here is. So, if I fix this T is equal to 1 and if I apply some clock signal then I will get some waveform at the Q. So, if I assume that the clock is having a frequency of f Hz, so, what is the frequency of Q?

We can see that this is one period of Q and here we have two periods of clock within one period of Q, you have two periods of the clock. So, the clock period of Q is twice that of clock then the frequency will be half of the clock. This frequency is f by 2. So, we can use this circuit as a frequency divider also. If we want to divide by a factor of 4, we need two such T flip flop. This is not J, this is T flip flop.

You have to design two such T flip flops. This is about the third flip flop which is T flip flop. So, mostly this T flip flop will be used in asynchronous counters. So, while discussing about the behavioural modelling, I will discuss the details of this asynchronous counters.

(Refer Slide Time: 10:12)



Next, another type of a flip flop is called D flip flop. In order to obtain the D flip flop from the JK flip flop, you have to connect a NOT gate between J and K inputs. This is J input, this is K input, this is clock and this means this negative edge clock. So, if I short this J and K, you will get T flip flop between J and K if I connect a NOT gate inverter you will get D flip flop, this will be D input, this is Q and Q complement.

So, D is equal to J is equal to K bar whereas, in previous case T is equal to J is equal to K in case of T flip flop. So, in order to obtain the truth table of D flip flop, if I take the same JK flip flop simplify truth table 0 0, Q n; 01, 0; 10, 1; 11, Q n complement. From this, you can easily derive the truth table of D flip flop also. So, this is D, Q n plus 1.

So, D can be 0 or 1, D is 0. So, this D is equal to 0 implies J is 0, K is equal to 1 this combination. So, what is the output here? 0. And D is equal to 1 corresponding to J is equal to 1 and K is equal to 0, this 1. So, what is the corresponding output? 1.

So, we can observe that here this Q n plus 1 is equal to simply D regardless of the previous state whatever the previous state of this D flip flop. So, if I apply D is equal to 0 output becomes 0, D is equal to 1, output equal to 1. So, simply whatever the information on this D signal will be transferred to the output. Then the question arises is what is the use of this circuitry then?

I can simply connect this D line and Q line; this is D input and this is Q output. If I short these two so, this will transfer the D signal to the Q signal that is what happening here also. Then what is the use of this NOT gate and inside this we have 8 NAND gates. We have discussed the circuit diagram of this master slave flip flop.

So, what is the use of this NOT gate and this NAND gate which are present inside this JK flip flop? Because the function is simply you are transferring the D data onto the output.

So, where this type of flip flop will be useful? Because this creates the delay so, if I give D is equal to 0, you definitely will get Q is equal to 0, but after some delay. So, what is that delay? Delay caused by this NOT gate and inside this we have nearly 8 NAND gates. So, after a delay of 8 NAND gates and one NOT gate, we will get Q is equal to 0 here. D is equal to 0, you will get Q is equal to 0; D is equal to 1, Q is equal to 1.

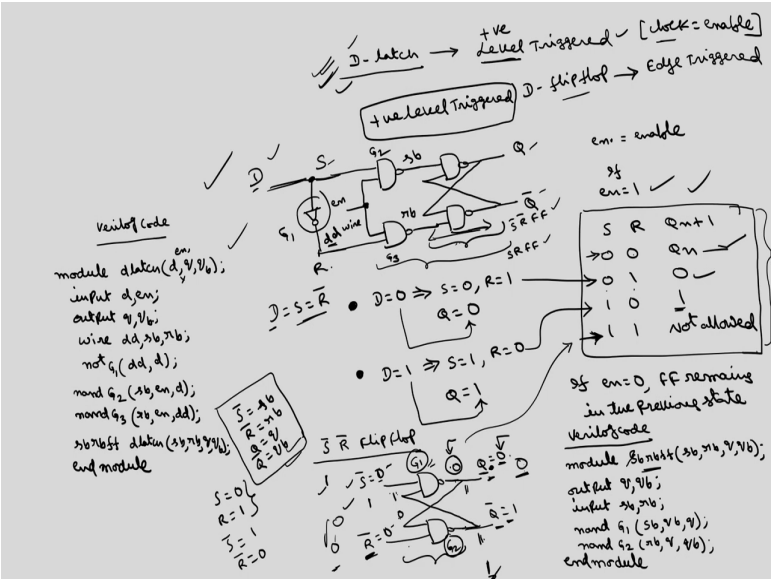
So, that is why the name D stands for Delay flip flop. Then where do you require this type of delays? So, this delay flip flop will be using to synchronize two signals with different speeds. So, there is a signal coming here S1, S2; S1 is a faster signal when compared to S2. But finally, here I want to synchronize this. Let us assume that here S1 dash, S2 dash. So, I want the speeds of this S1 dash and S2 dash has to arrive simultaneously.

So, these two S1 dash and S2 dash has to arrive at the same time because S1 is faster S1 dash will arrive fast if I do not have any circuitry in between this. So, to synchronize this S1 dash and S2 dash what is required is S1 is faster, this I will pass through the D flip flop. We can have single D flip flop or we can have series of D flip flops then we will connect this directly. So, that S1 dash and S2 dash will be synchronized.

So, this number of flip flops D flip flop that you have to choose depends upon how fast is this S1 compared with the S2. So, we can use the delay equalization circuits. In addition to this, we have some other applications of this D flip flops are: we can use this D flip flop in shift registers. In order to construct the shift registers, we need D flip flops; even using this shift register also, you can generate the random number generators which are very much useful in many of the communication circuits.

I will discuss this shift register also while discussing about the behavioural modeling. Now, I will discuss how to write the Verilog code for the D flip flop. So, I will start with a simple SR flip flop because D flip flop no need of this edge triggered also. We can have level triggered D flip flops also, we can use in the shift registers. The reason for this one is so, in case of D flip flop 1 1 condition never occurs. To get the D flip flops, you can use simple SR flip flop.

(Refer Slide Time: 17:56)



So, I will just discuss how a D latch can be obtained. Generally, this latch is nothing but level triggered. Whereas, D flip flop if I write, I will discuss after this D flip flop using SR flip flop this is called edge triggered.

So, I will discuss first D latch. So, in D latch because this is level triggered, I will assume this as positive level triggered. So, whenever this you can call as a clock or simply you can call as enable signal. So, in case of level triggered normally, we will use this clock signal as enable signal. Normally, this clock concept will be useful for this edge triggered flip flops.

So, if I draw the simple circuit diagram of this one. So, we know that this SR flip flop is this. This is Q, this is Q complement, then we have S, R, then here we have the clock, but I will call this as clock or enable, I will call simply as enable en signal; en stands for enable. This is a simple clocked SR flip flop or enable SR flip flop because we are assuming positive level

triggered; that means, whenever enable is equal to 1, then the SR flip flop truth table will follow.

S, R, Q n plus 1 is 0 0, Qn; 0,1, 0; 1,0, 1; 11 is not allowed, but here if I want to take this D latch, this 11 condition will never occur. So, this will be valid only for enable is equal to one and if enable is equal to 0, the flip flop remains in the previous state. Now, to obtain the D latch, so, we know that between S and R we have it can be JK or it can be SR if I connect between these two a NOT gate and you can call this one as D input.

So, D is equal to S is equal to R bar ok. So, in the earlier slide we have discussed about the derivation of D flip flop or D latch from JK flip flop. Now, this is actually from D latch. So, the construction of D flip flop from SR flip flop. So, when D is equal to 0 that implies S is equal to 0, R is equal to 1. So, this is corresponding to this. What is Q? Is equal to 0. So, whatever the D is followed to Q.

When D is equal to 1 implies S is equal to 1 and R is equal to 0 implies this is corresponding to this. So, output is 1. So, the same input is transferred to the output. In case of D latch, no need of master slave flip flop, even you can use a simple SR flip flop to derive the D flip flop, but this is not the case of T flip flop. T flip flop we cannot derive from this SR in a simple manner by starting this S and R because 1 1 condition is not allowed here.

Now, how to write the Verilog code for this D latch? So, for that I will just simply start with this part, I will call this as S bar R bar flip flop. After that I will discuss about the SR flip flop, this is S bar R bar flip flop, this I will call as SR flip flop. Now, I will derive the D latch by instantiating D latch with SR flip flop, SR flip flop with S bar R bar flip flop. So, what is S bar R bar flip flop? This portion S bar R bar flip flop.

So, simply I am taking two NAND gates connected back to back. This is Q, this is Q bar, this is S bar, this is R bar. You can verify this also will give you the same circuit, this same truth table. So, when S is equal to 0, R is equal to 0, if I consider this combination S is 0, R is 0 means S bar is equal to 1, R bar is equal to 1. So, to get this the output of this NAND gate I should know the other input.

So, if I assume that Q initially is 0, Q bar is equal to 1, this 0 is tied here, this 01 becomes 1, 11 becomes 0. So, 01 becomes 1, 11 becomes 0. So, it will get previous state is if assume that as 0 present state is also 0. Similarly, if I assume that previous state is 1. We will get present state as also 1. So, we will get this Q n.

Similarly, if I give this S is equal to 1, R is equal to 1, 1 1. So, S bar is 0, R bar is 0. So, this will give ambiguous state. This is ambiguous state because both will become 1. So, same as the ordinary SR flip flop and similarly if I give S is equal to 0, R is equal to 1 this S is equal to 0, S bar becomes 1, S is equal to 0, R is equal to 1 implies. So, what will be applied here? S bar is 1, R bar is 0.

So, this is 1, this is 0. So, if this is 0, this is 1 because for NAND gate if one input is 0, output is 1. So, this second input is 1. So, this 1 1 will give 0. So, this is 0 is tied here again 0 0 becomes 1. So, output will remain Q. You can easily verify that the same truth table is valid for this S bar R bar flip flop also. Now, I will take this S bar, R bar flip flop as a basic circuit because this portion is common in D latch also.

So, later I am going to discuss about the positive edge triggered flip flop, there also we have this type of the portion. So, in order to simplify this programming so, we can instantiate these logics. So, first I will write the Verilog code corresponding to this S bar R bar flip flop. So, I will give the module name as s, I will write all lowercase letters sbrbff flip flop, this is the notation I am using.

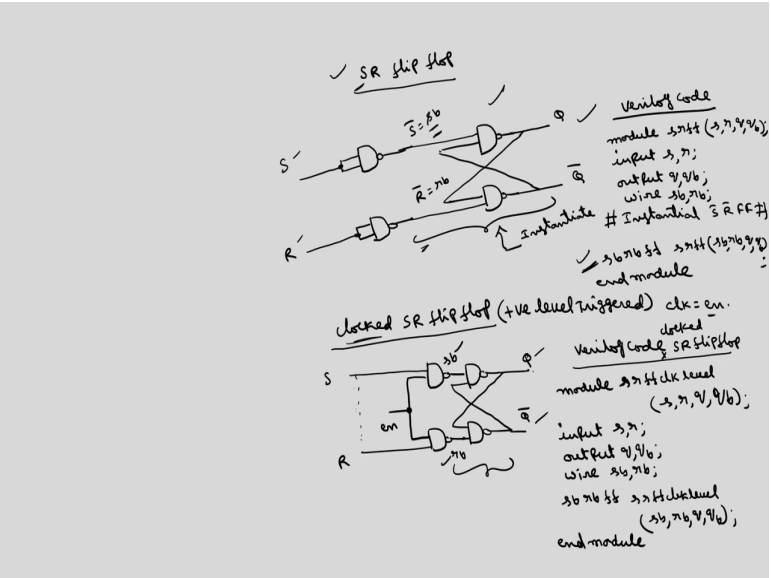
sb stands for this S bar input, rb stands for R bar flip flop and what are the inputs and outputs of this flip flop? I can give in any order. So, sb I will call as the input. This input S bar instead of I cannot give this S bar I am calling this one as sb; R bar I am calling as rb; Q I am calling as lowercase letter; Q bar I am calling as qb ok this was notation I am using.

So, sb is the input, rb is the input; corresponding to sb, this S at this output we have Q at R, we have Q bar. So, you can give in the same order q, qb semicolon. So, this module name I am calling as small sbrbff. Later if want to instantiate you have to call this name. So, I will write here the outputs are q and q bar, inputs are s bar, r bar.

Then NAND if I call this NAND gate as G1 gate if I call this one as G2 gate NAND is the primitive used for this NAND gate G1 is the name optional. So, what are the outputs and inputs of this G1 gate? I will give first inputs and then outputs. One input is sb, other input is this qb, output is q.

Second NAND gate G2, what are the two inputs? One is rb, another is q, output is qb end module. So, this is the code corresponding to this S bar R bar flip flop. So, finally, I want D latch. So, I will derive this D latch after two stages.

(Refer Slide Time: 29:19)



Then I will discuss about SR flip flop. So, this is Q, Q complement, this was S bar, R bar. Now, without clock signal SR flip flop if I call this one as this. Simply you have to use a NOT gate to get SR as the name implies SR S bar R bar SR, only difference is NOT gate.

So, NOT gate I am going to implement using NAND gate, two minimum inputs are required. So, I am going to short this. This is your SR we can verify the truth table of this one also same as the previous truth table. So, this is S bar R bar. This circuitry we have discussed. Now, what is the Verilog code corresponding to this SR flip flop? This code you can instantiate. So I will call this module as module sr flip flop.

The previous module name we have given as sbrb flip flop because the inputs are S bar R bar, now we have S and R inputs. So, I will give the inputs as s and r, outputs as q and qb,

semicolon, then we require here two wires. So, we have to define two wires and inputs and outputs; input SR output q, qb. Then wire I will define this as S bar this is sb, this is equal to rb, sb comma rb. Then we have to instantiate S bar R bar flip flop.

So, how do instantiate? The name that I have given is previous one sb rb flip flop. You can give some name for this as sr flip flop comma what are the outputs and inputs here? Inputs are sb rb, outputs are q, qb end module. So, simply we have discussed about the S bar R bar flip flop which is the basic block then simply we have moved on to SR flip flop.

By instantiating this we can easily write the Verilog code. Now, another step is enabled SR flip flop or clocked SR flip flop. So, this clock, we are calling as this as a positive level triggered. So, clock you can also called as enable signal here. So, if enable is equal to 1, the circuitry will be enabled. So, this of course, you have drawn already the circuit diagram. I will write directly the Verilog code.

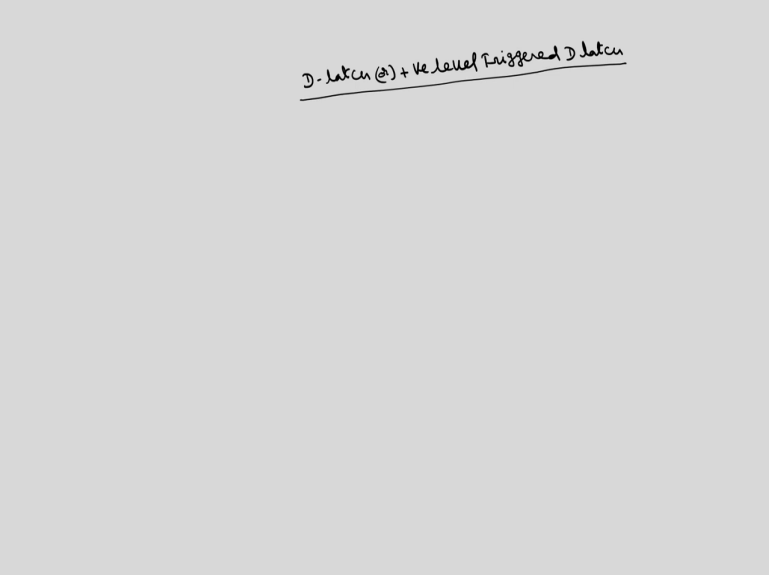
So, it is the same circuit diagram except for this NOT gate. Here will be having clock signal or enable signal, then we have S, we have R, the remaining circuitry same as S bar, R bar flip flop. This you called as sb wire, this you called as rb wire, this is q, q bar. Now, this part is again common. So, I can instantiate this ok.

Now, what will be Verilog code corresponding to SR flip flop with enable signal? You can also call clocked say module you can call this one as sr flip flop clock level. I have given this as name. So, what are the inputs and outputs? s, r the inputs, corresponding outputs are q, qb.

Then of course, the same as this input s, r; output q, qb; wire sb, rb then you have to instantiate. I am then relating sbrbff this was same name that we have given in the last slide. Then this name I am calling as srffclklevel you can give any name. So, what are the outputs and inputs of this flip flop? This sb rb are the inputs, q and q bar are the outputs. sb you can give sb, rb, q, qb then end module.

So, slowly we started with S bar R bar flip flop, then we moved to SR flip flop, then we have applied the clock also. Now, what is D latch? Simply you have to connect a NOT gate between this point to this point.

(Refer Slide Time: 36:45)



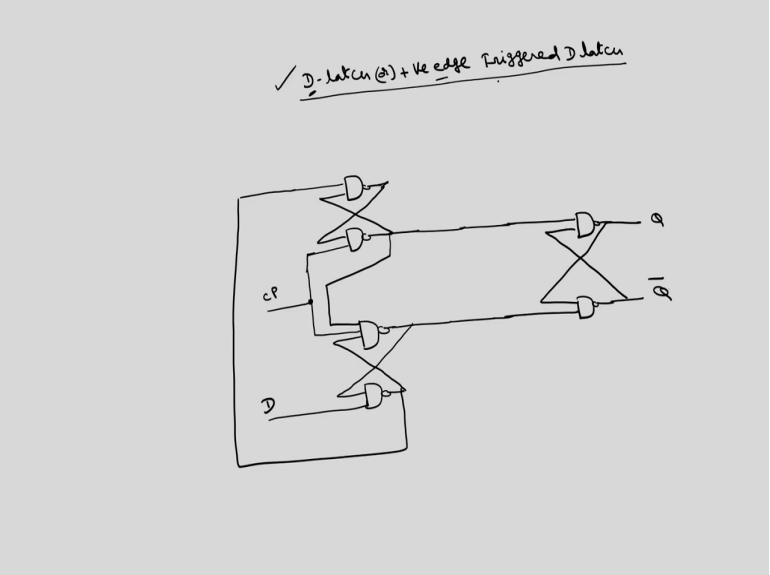
So, what is the code corresponding to D latch or positive level triggered D flip flop? So, I given the circuit already. This is circuit diagram of positive level triggered D latch.

So, input is 1, outputs are q, q bar and here also we have defined one wire. I will call this D wire as dd wire. I will write here itself the Verilog code. So, module dlatch input is d, outputs are q, qb. Then we can have input d, output q, qb then we require here sb rb as the wires another wire is dd, sb, rb.

So, after that to derive this dd we require a NOT gate, primitive is NOT; the output of this NOT gate is dd, input is D, then you can call this one as gate G1, this is G1, this is G2, G3 after that we can instantiate that nand G2. So, what are the output is sb enable also you have to give input here you have to write enable and d and nand G3 output is rb enable dd then you have to instantiate this.

So, sbrb flip flop and you can give the name of this as some other circuit which is dlatch. So, what are the inputs and outputs? Same sb, rb, q, qb end module. So, this is about the Verilog code corresponding to a positive level triggered D latch. So, we can derive this negative or positive edge triggered D latch also using the same basic circuit.

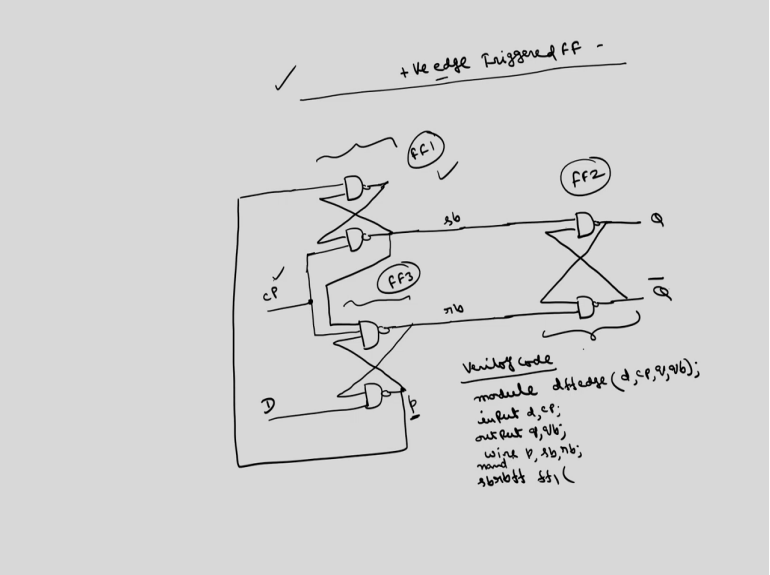
(Refer Slide Time: 40:34)



So, instead of positive level now I will discuss about the positive edge. So, you can easily verify that I have explained the edge triggered flip flop using the circuit in the last lecture. So, we can construct now using three such latches. This is one S bar, R bar, SR flip flop, this is final output Q, Q bar. This is two inputs and then we have one more this type of S bar, R bar flip flop.

Here we are going to apply the clock signal CP because this is negative edge triggered. So, we cannot call as enable ok we can call as clock signal. So, this is having of course, third input also this is having two inputs, the second input of this one is D. So, the D input of D flip flop. Of course, this is connected to this. So, you can easily verify that this circuit will act as a D positive edge triggered flip flop, you can call this one as not latch flip flop.

(Refer Slide Time: 42:37)



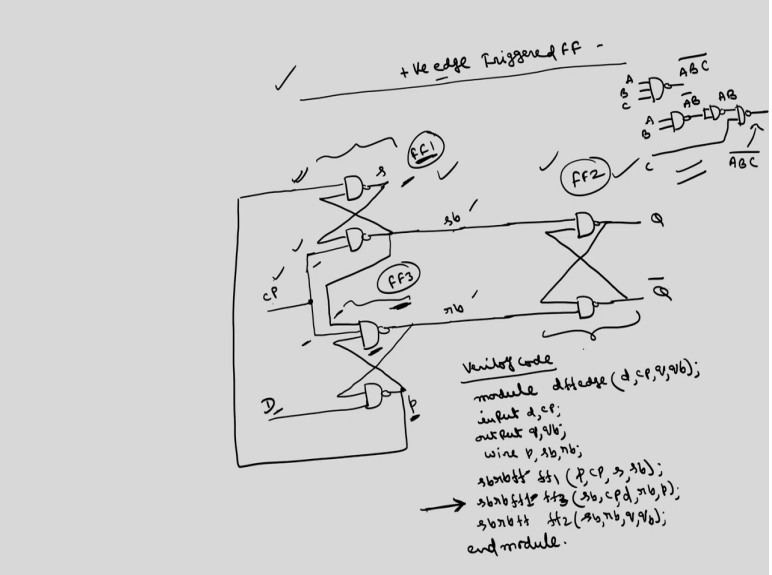
So, the basic difference between this latch and flip flop is normally edge triggered means we will call as flip flop, level triggered means we will call as latch. Now, here we have this circuitry, we have code, this circuitry we have code, this we do not have the direct code. But we can derive this code of this one by slightly modifying these two codes.

So, if I write this Verilog code corresponding to this one, module I will give this name as d flip flop edge triggered, what are the inputs and outputs? We have input is d, CP, this clock signal then q, q complement qb, and then input d comma CP, output q comma qb then we require some wires. So, what are the wires here? You call this one as sb rb wires.

So, what else is required here we require some wire we call this one as some P wire, this is also another wire is required. Of course, this is sb. So, only these three wires are required wire p, sb, rb. Then you have to instantiate 3 times. So, I will call this one as flip flop 1, flip flop 2; of course, this is not exactly same as flip flop 1, flip flop 2. I will call as flip flop 3. I will discuss how this can be derived from the other two flip flops.

Then sbrb flip flop this was the name used you can call as this as flip flop 1, this one. What are the inputs and outputs of this? One is p, how to derive the p? First of all here you have write the p. So, nand rb. p is nothing, but output of this one. So, we will first derive this p.

(Refer Slide Time: 45:30)



So, flip flop 1 this one. If you consider this flip flop 1, the inputs are this is one input, this is input. This input is we are calling as p, this input is CP. p, CP, outputs are Q and Q bar. So, of course, one is sb you can call this one as some s, outputs are s, sb corresponding to this flip flop sbrb ff1. I will call this as 1, the code corresponding to this one is sbrbff1.

So, this is ff2 or ff3. What are the inputs and outputs? We have three inputs. This sb is the one input, other input is cp, the other input is d. The outputs of this one are rb comma p, then sbrbff2 ff2 what are the inputs? sb, rb, outputs are q, qb this is end module.

So, the code corresponding to this flip flop I am calling this is ff 1, this is ff ff this is ff only this is also ff only, but this is ff1. So, to derive this ff1 from this two input NAND gate basically here you should have only two input NAND gate, but we have three input NAND gate. Suppose, if I want to derive a three input NAND gate from two input NAND gate this is A B C.

I require A B C bar using two input NAND gate how to derive? You take two input NAND gate we will get AB bar you pass through a NOT gate we will get AB, then we give a third input as C NAND gate. So, this output will be now AB is one input, C is the other input, ABC whole bar. So, using this logic we can write down the code corresponding to this, I am not writing this ok.

So, this is all about this gate level modeling. So, in the next lecture we will discuss about the switch level modeling.

Thank you.