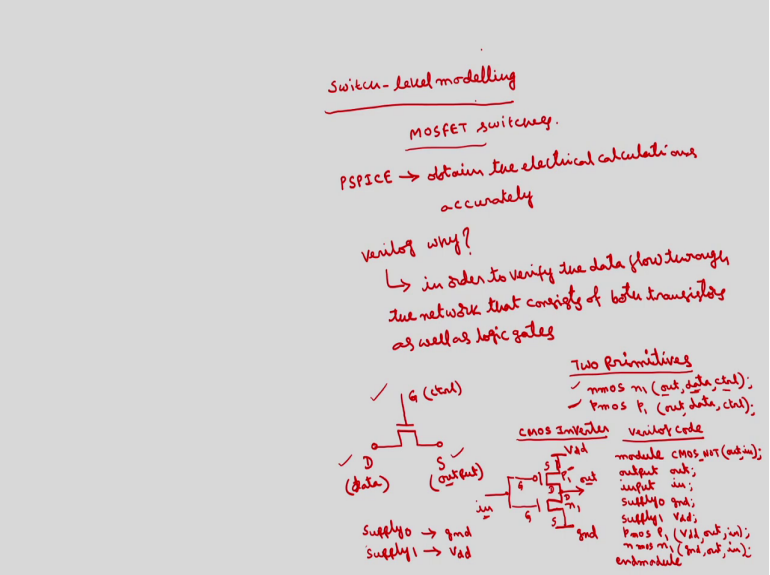
System Design Through VERILOG
Prof. Shaik Rafi Ahmed
Department of Electrical and Electronics Engineering
Indian Institute of Technology, Guwahati

Switch level modeling
Lecture - 11
Modeling of CMOS gates and Boolean functions

(Refer Slide Time: 00:37)



Out of the four levels of Verilog modelling, switch level modelling is one which deals with the switches. So, normally, we can use the transistor as a switch. We will deal here with instead using BJT, I will deal with the MOSFET as a switch. So, we know that at the circuit level, normally we will use the software like PSPICE in order to obtain the critical electrical calculations more accurately.

If you use the PSPICE and all, they will obtain the electrical calculations such as voltage current and all accurately. Then, why we have to go for the Verilog modelling? So, this Verilog modelling will be useful, in order to verify the dataflow through the network that consists of both transistors as well as logic gates. If the output of some transistor circuit is connected to some logic circuit so, a network which consist of both the transistors as well as the logic gates.

So, in order to verify the dataflow through such networks, normally we will use the Verilog coding. So, basically this switch level modelling here we are going to deal with the MOSFETs. So, this I will explain the behavior of the MOSFETs. So, we know that if I take the n-channel MOSFET, this will have three terminals in fact, four terminals and the subtract we will not discuss here. One is called gate, another is called source, another is called drain.

So, normally, in order to code in the Verilog so, we will use two primitives, there are two primitives to represent this NMOS and PMOS. So, one is NMOS is one primitive, another is PMOS. The general structure of this two primitives is NMOS some name, gate name it is optional n1. So, we have to represent out, data, control. There are three signals. Similarly, PMOS also some p1 out, data, control. Here, normally this gate will act as a control signal ctrl, drain will act as a data and source will act as output.

Suppose, if I want to write a Verilog code for a simple inverter. So, we know CMOS inverter consisting of one NMOS device and another PMOS MOSFET. So, this is basically Vdd, two MOSFETs, this is ground, this is Vdd, this is PMOS, this is NMOS, this two will be connected to the input, let input is in and here, we have take the output out.

This is PMOS, I will call as p1, this is NMOS I will call as n1. So, the Verilog code correspond to this CMOS inverter will be. We have to use this primitives NMOS and PMOS. In addition to this, we are going to use two more primitives called as supply 0, supply 1, this is equivalent to ground. Supply 1 is another primitive normally used to represent Vdd.

So, in the Verilog code, I am giving the module I will give some name CMOS underscore NOT, only one input, one output out, in. So, output is out, input is in. In addition to this, you have to give supply 0 is nothing, but gnd, supply 1 is nothing, but Vdd. Then, p1, if I take this p1 so, this primitive is PMOS p1, this is optional name of the gate.

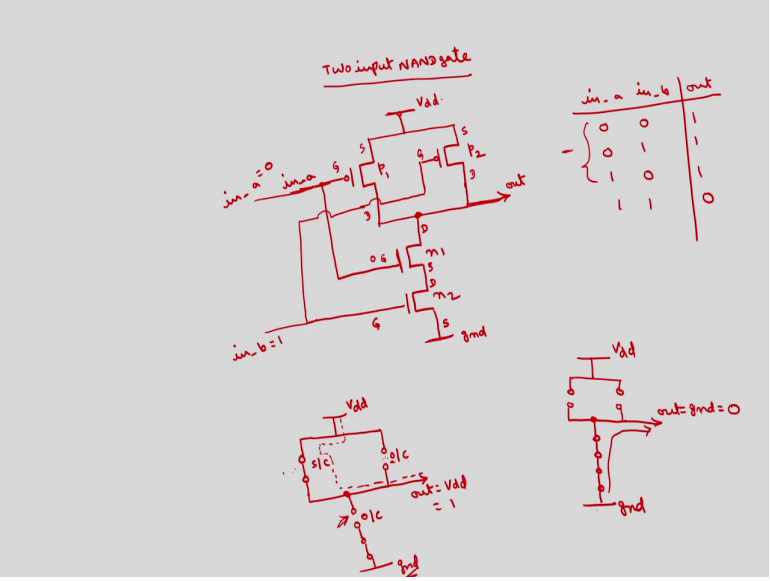
So, the general structure is out, data, control. So, out is normally source, drain, gate. So, for this inverter, this is source of PMOS, this is gate of PMOS, this is drain of PMOS whereas, for NMOS, this is source, this is drain, this is gate.

So, for PMOS p1, first we have to mention the output which is nothing, but source, source is at dd so, we have to write Vdd comma, then data, data is nothing, but drain, drain is nothing,

but output out, then control, control is gate which is in. At gate, we are applying the input signal. So, this will describe the PMOS transistor.

Similarly, NMOS transistor n1. So, what is source? Source is ground so, gnd, output is drain, drain is output. In both the cases, output is taken from the drain, out and both the gates will be operated at in only so, in, this is end module. So, this is how we can represent any of this CMOS based circuits using Verilog code. So, I will take some other gates the universal gates such as NAND and NOR and how to represent the Verilog code corresponding to these gates.

(Refer Slide Time: 09:46)



If I take two input NAND gate. So, the circuit diagram of this two input NAND gate is there will be two NMOS and two PMOS MOSFETs, this is ground point, and this is Vdd point and here the output is taken, these are two inputs. So, this will be connection is there from here to here, this is PMOS 1, this is PMOS 2 p2, this is NMOS 1, NMOS 2.

So, this I will call as in underscore a, this is also in underscore a, these two will be connected. This is in comma b, this input is, this is in underscore a, this is in underscore b, we have two inputs and one output. We can easily explain the operation of this one. So, for the NAND gate, the operation is if we have in a, in b and output, 0, 0, 0 x 0 bar should be 1; 0, 1 also 1; 1, 0 also 1; 1, 1 is 0.

If any one of the inputs is at 0, output will be 1. If say for example, this in a is at 0 so, what happens is the corresponding PMOS will be on so, this will be in ideal case, this will act as short circuit, the operation is something like this is Vdd and here, I am taking the ideal case, this will act as short circuit because in a is 0, in b is say for example, is 1, it can be 0 also.

So, this will be open circuit, this will be off, I am trying open circuit, this is short circuit, this is ideal case. I will discuss the practical case later, this is Vdd.

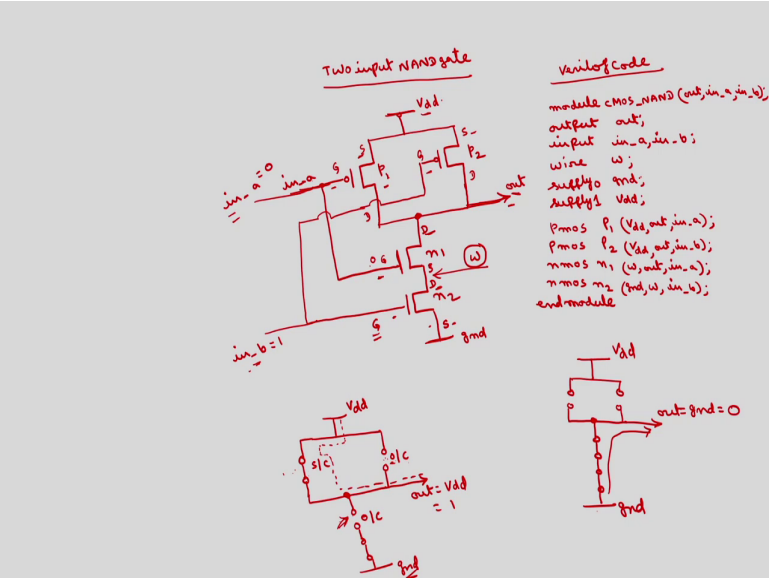
So, for the lower NMOS MOSFETs, in a is 0 means this will be off, this will acts as a open circuit and this will acts as short circuit, this is gnd. So, we can see that here we are taking the output. We can see that here so, here there is no path to connect to ground, here open circuit is there, but through this Vdd, it will come from this point through this short circuit so, this output will become Vdd that is logic 1.

So, regardless of whether both these switches are closed, if any one of the switch is closed, this is the situation, these three conditions. So, in these three condition, either of this one is short circuited or both will be short circuited, in any case this will connect through this. So, here in any one of these three cases, one of this, at least one of this switch will be open circuited.

So, there will be no path for the ground and if I take the last case, if both are 1's, the equivalent circuit will be so, we have something like both of this transistors will be off so, here there will be open circuit whereas, the NMOS transistors both will be on, they will act as short circuit, so that this path will be connected here, output becomes ground which is 0. So, this is how we can explain the operation of two input NAND gate.

So, now, how to write the code corresponding to this two input NAND gate? So, first we have to write down this source and drains of this NAND gates. So, this is the source, gate, drain. This is source of n2, we can call this as gate of n2, this is drain of n2. Similarly, here, these are sources for PMOS, these are drains and these are gates.

(Refer Slide Time: 16:07)



Then, how to write the code? I will write here itself; I will remove this. Module, I am giving the name as CMOS underscore NAND, two inputs, one output. So, output we are calling as out, inputs we are calling as in underscore a, in underscore b. Then, we have output is out, input is in a, in b, then we need a wire.

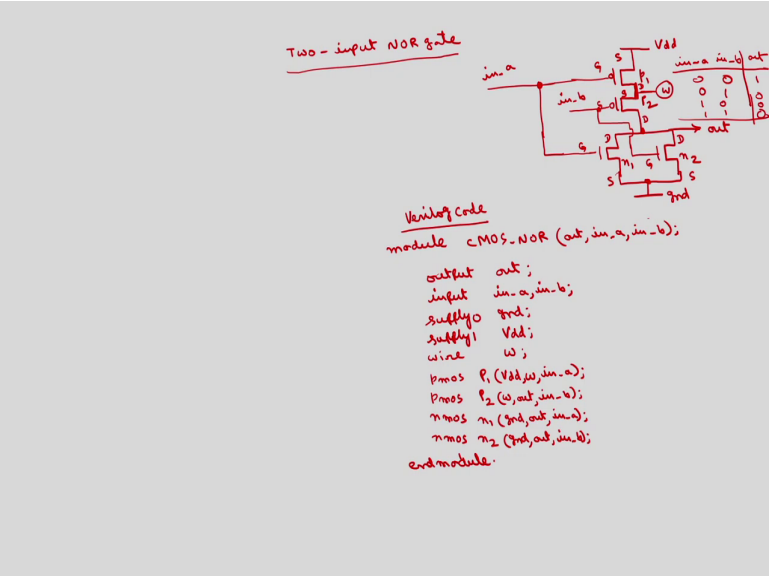
So, why because you can also think so, this terminal is already available as in a, this terminal is available as in b, this is Vdd, sources are Vdd, Vdd also we are going to define, this is ground we are defining, this also in b, this also in a.

So, the only thing that is not accessible is this point, the point where the source and drain of this n1, n2 are connected. So, this is internal connections so, I have to define a wire, I will define this as a wire w. Then, supply 0 ground, supply 1 Vdd. So, this is a basic initialization.

Now, coming for this p1, PMOS primitive p1 source, drain, gate. The terminals of the source we have to write first so, for p1 source is at Vdd so, Vdd, drain of PMOS is at out. So, comma out and gate is at for PMOS p1 in a, similarly PMOS 2, p2, source is at Vdd same thing, drain is at out, the only difference is instead of in a, this will be at in b.

Similarly, NMOS n1 source, drain, gate, n1 source is wire, drain is out, and gate is in a whereas for NMOS 2, source is at ground, drain is at wire, gate is at in b, this is end module. So, this is how you can describe a two input NAND gate using Verilog code.

(Refer Slide Time: 19:57)



So, I will take the next universal gate which is NOR gate. So, the connections are reverse here. So, NOR gate is here basically this is Vdd. So, in case of NAND gates, NMOS transistors are in series, p are in parallel whereas, here this is reverse, p are in series, n are in parallel. This is ground point, this is out point, this is in a, this is in b.

So, we know the operation is for in a, in b, output 0, 0. NOR is 0 plus 0 bar, 0 bar is 1. 0, 1, 1, 0, 1, 1. So, for any other case, 1 plus anything is 1, 1 bar is 0. So, if all the inputs are 0, output should be 1. If any 1 of the input is 1, output should be 0. So, you can easily verify the operations similar to that of the NAND gate. So, in ideal case, if a device is on, we will take this as a short circuit otherwise open circuit, we can verify in the similar manner.

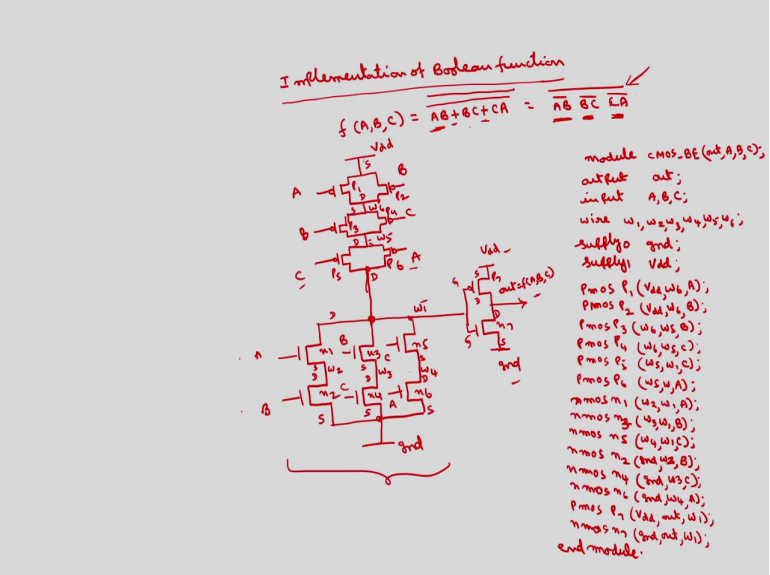
Now, if I take the Verilog code corresponding to this one, we will call this one as p1, p2 similar to NAND operation, this is n1, n2. So, module I will call as CMOS NOR, we have only one output and two inputs, output out, input in a comma in b, then we require supply 0, gnd, supply 1 Vdd, then here also we require a wire, which is not accessible to anyone of this

inputs, outputs see its this point. So, this is the point where this p1 and p2 are connected, this is not accessible so, we have to define as a wire.

And if you write down the source and drains of these one, these are the sources of both NMOS transistor, these are the gates, these are the drains, this is the source, gate, drain, this is again drain sorry this is source, gate, drain. So, if I take PMOS p1 so, what is source, drain, gate? Vdd comma drain is at wire and the input is gate is at in a, PMOS p2 source is at for p2 source is at w, drain is at out and gate is at in b.

If I take NMOS n1, source is at ground in both the cases, these are ground and drain in both the cases will be output, gate will be in case of n1, it is in a, in case of n2, it is in b, this is end module. So, like this you can verify the Verilog corresponding to the NOT gate, NAND gate and NOR gate. But any Boolean function can be represented in terms of this NOT, NAND and NOR gates.

(Refer Slide Time: 26:25)



So, if I take example of some other Boolean function. Let us take f of A, B, C is equal to AB plus BC plus CA this is nothing, but actually the carry of a full adder. Now, how to implement this? AB plus BC plus CA so, in order to implement using the NAND gates, I can represent this as double bar.

If I take the double bar, we will get the same value. So, if I evaluate the lower one according to the De Morgan's law, P plus Q plus R bar is equal to P bar into Q bar into R bar. So, this is AB bar BC bar CA bar. This is one NAND gate, this another NAND gate, this is another NAND gate overall, this is another NAND gate.

So, how to implement this? So, AB is a NAND gate, BC another NAND gate, CA another NAND gate overall, you have to implement another NAND gate. So, this structure will be something like so, in the NAND we will be having this in series, for the NOR we will be having in parallel.

So, I will first write down the pull-down network. A, B, this is B, C of course, this B and this B are same, B, C, this is C, A and above this one will be, this will be in the series components in parallel, parallel components will be in series. This is A, B, B, C, C, A see we are going to take the output here, this is Vdd. So, what will be this output is nothing, but AB plus BC plus CA whole bar.

So, we have to take this output, we have to pass through the NOT gate again. Of course, these two grounds are same, this Vdd is same, this is final out which will give AB plus BC plus CA. So, to get this AB, we know that this is will be series, BC which should be series, CA series to get this NOR operation, they will be in parallel ok. So, this is the circuit diagram of f of A, B, C is equal to AB plus BC plus CA.

We can do in other way also, we can take the NAND operation of this one, NAND of this one, NAND of this one and then, you can take the overall NAND. So, for this, what will be the Verilog code? So, module CMOS Boolean expression BE, we have three inputs, one output, out comma A comma B comma C, output out, input A comma B comma C.

So, now, you tell me how many wires are required? So, all the points which are not accessible require a wire ok.

So, what are the wires required? So, this point is not accessible, here I require a wire, I will define this wire as wire w1, here we require w2, w3, w4, this is Vdd, this is ground, this is out, here do not require, here this C, this is A so, this point is also wire w1 whereas, here we

require some w5, this is w6 and here of course, this is Vdd. So, total six wires are required and then of course, supply 0, gnd, supply 1 Vdd.

We name this transistor, total we have 12 transistors, we call this 1 as p1, p2, p3, p4, p5, p6, n1, n2, n3, n4, n5, n6. So, PMOS p1 so, what is source, drain and gate? This is source, this is drain of p1 and p2, this is source, this is drain of p3 and p4, this is source, this is drain of p5 and p6. Similarly, these are the sources of n2, n4, n6 so, these are the drains, these are the sources of n1, n3, n5, these are the drains.

So, for p1 source; p1 source is at Vdd, drain is at the w6 and gate is at A, this is for p1. Similarly, PMOS p2 source; p2 source is at Vdd same, p2 drain is w6 only, but input will be B. Similarly, PMOS p3 source is at w6, drain is at w5 and gate is at B, PMOS p4 also same except for the control signal, which is gate w6, w5 instead of B, this will be C and PMOS p5 source will be at w5, drain will be at w1, gate will be at C, PMOS p6, p6 also same w5, w, the only difference is input is A.

Then, similarly six NMOS, NMOS n1 source of n1 is w2, drain is w1, gate is A; NMOS n2 source is same w3, w1, the only difference is instead of A sorry n1, n2, n2 is different, n2 is source is at ground, this will be for n3 otherwise. So, n3 will be same as n1 source, drains are same.

So, source is at w3 and drain is at w1, input is B; NMOS n5 so, the source is at w4, drain is at w1, input is at C; NMOS n2 source is at ground, drain is at w2, gate is at B; NMOS n4 source is at ground, drain is at w3, gate is at C; NMOS n6 ground, w4, A, this is up to here.

Then, we have another NOT gate, here also I will call this one as p7, n7. So, PMOS p7 source of p7 is this is source, this is gate, this is drain; this is source, drain, gate. Source is at Vdd, drain is at out, gate is at w1, NMOS n7 source is at ground, drain is at out, gate is at w1, this is end module.

This is how you can implement any Boolean function; you can implement any Boolean function and you can write the corresponding Verilog code in this manner. So, I will discuss about this pull up, pull down networks if we do not consider the transistors in ideal case in the next lecture.

Thank you.