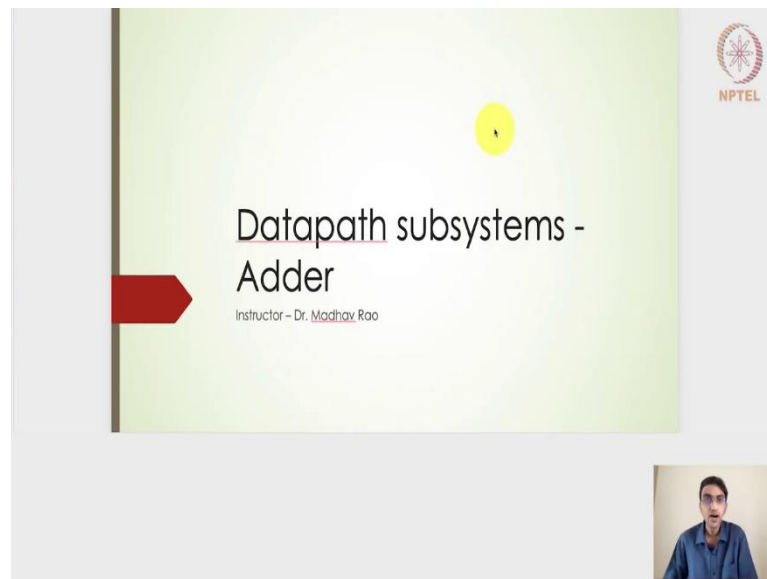


**Design and Analysis of VLSI Subsystems**  
**Dr. Madhav Rao**  
**Department of Electronics and Communication Engineering**  
**International Institute of Information Technology, Bangalore**

**Datapath subsystems - Adder**  
**Lecture - 85**  
**1-bit Adder design**

(Refer Slide Time: 00:16)



Hello students welcome to this lecture on Data Path Subsystems and we will be looking mostly to the adder. In this first particular module of the adder circuit we will look at the circuit level designing of the adder. What we will be doing is we will look into the 1 bit half adder and then the full adder and then go about understanding the layout of the 1 bit full adder circuit.

Subsequently we will also take a look into the adder subsystem. If I want to make it into an 8 bit or a 16 bit or a 32 bit adder subsystems how do we go about designing those in the future lectures. Let us begin with this particular adder circuit, understanding the adder circuit.

(Refer Slide Time: 01:01)

Half Adder

A	B	Sum	Carry-out C <sub>out</sub>
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = A \oplus B$$

$$C_{out} = A \cdot B$$

As we know the half adder which does not involve the carry input and then has only 1 bit of A and 1 bit of B and then we want to get the summation of the 1 bit of A and 1 bit of B and then the carry out due to the addition of the 1 bit of A and B. In this the two input bits, where the carry input is not a lot considered we will have four such combinations starting from 00 to 11 and then we will have 4 such output values.

For 00 the sum is 0 the carryout is 0 for 0 1 the sum is 1 carryout is 0 for 1 0 the sum is 1 the carryout is 0 and then for 1 1 the sum is 0 and then the carryout is 1. In that sense if I look into the sum bit wherever it is 1 it is either A or B is either 1 and then the other one is 0. We will have the XOR operation of A and B, the XOR operation of the input A with that of B will give us this sum bit.

$$S = A \oplus B$$

$$C = A \cdot B$$



The carryout if I consider, if I look into the logical values of 1, it is only 1 here in the last combination when A and B is 1. The carry out for a half adder will be nothing but A and B, hope this is pretty clear to everyone.

(Refer Slide Time: 02:28)

Full Adder				
A	B	C <sub>in</sub>	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$Sum = A \oplus B \oplus C_{in}$   
 $Sum = AB\bar{C}_{in} + \bar{A}B\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in}$

$Carry = AB + BC_{in} + AC_{in}$   
 $= \text{Majority}(A, B, C_{in})$

For a full adder we will consider the not only A and B, but also the carry input, it may be the carry input coming from the previous input additions. If I consider this 3 inputs I will have 8 such combinations, this will be 8 such combinations and then, we will have 8 such output combinations of the summation output and then the carry output.

If I start to looking into the input combinations from all 0 0 0s to all 1 1 1. We will get the sum and then they carry out and it is very interesting to find out for which particular input combination the sum is actually 1. The sum is 1 here when the C<sub>in</sub> is a one and then the other two are 0s and then the sum is 1 here, when B is 1 and the other two are 0.

Sum is 1 here when it is A is 1 the other two are 0 and then 1 here when all the three inputs are 1. For the inputs where it is two of them are 1, the sum is actually 0 and then where it is two of them are 1 it is 0, two of them are 0 it is 0 all of them are being 0 it is 0.

In that sense the sum is actually can be considered as an XOR operation of all the three inputs. It will provide either it will give us the odd number of one's between these 3 input combinations. That is why the sum is considered as

$$Sum = A \oplus B \oplus C$$

Carry out it is again very interesting to see if I consider the carryout let me take a different color. If you know in this particular combination the carryout is 1 when we have B and C to be 1 the carryout is one here when A and C are both ones carryout is 1 when we have A B to be 1 and it is 1 finally for all the three combinations to be 1.

For all these three combinations we can easily say with the majority of the three input with the majority of the three input that means, if the majority of the three if it all it is two of them are showing one out of this three combinations A B Cin, if two of them are showing 1 then carryout is 1.

If all the three are showing them 1 then definitely the carry out will be 1. This is called as a majority gate with the inputs coming from A B and Cin. Its logical expression is nothing but,

$$\text{Carry} = AB + BC_{in} + AC_{in}$$

(Refer Slide Time: 05:18)

Propagate	A	B	C <sub>in</sub>	S	C <sub>out</sub>	Generate
0	0	0	0	0	0	0
0	0	1	1	0	0	0
1	0	1	0	1	0	0
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	0	1	0
1	1	1	0	0	1	1
1	1	1	1	1	1	1

$G = A \cdot B$

Hope this is clear for the 1 bit full adder. This is again 1 bit full adder, if I consider this particular portion this is nothing but the previous eight input combinations with the output values associated with it. Here the addition is nothing but the propagate and the generate bit I think this is going to be very useful in designing the higher order at a subsystem level the higher order the adder blocks; and this propagate and generate bits which are kind of evaluated from the input combinations and that will be used for generating the higher order adder bits.

Just to complete the truth table I have added the propagate and generate bit and let us take a look at the propagate first and then take a look at the generate bit. The propagate bit it is

1 and 1 here, when you consider the Cin the carry input is 1 here which is propagated to the carry out or if the 0 is there it is propagated to the carry out.

It is 1 here, it is propagated to the carry out here. Similarly, Cin is 1 here it is propagated to the carry out and if it is 0 it is propagated to the carry out. For the other combinations when the Cin is 1 it is propagated as 0. That will not be considered. What I meant is this 1 and 1 bit, the propagate is 1 only when we consider for an input combinations when the Cin is 1 and it is propagated to the carry output, without generating the carry.

Now, what do we how do we define the generate. The generate signal is without Cin, if the inputs A and B it is able to generate the carry, that is possible only when A and B are 1 and if it is able to generate the carry here then the generate signal is 1.

Out of all these 8 combinations we can see that the carryout is actually 1 in three of the combinations or rather in this four combination the three of the combinations this one this one and this one and this one in the two combinations. The carry input is 1 here and that is how it is generating the 1 here in the carryout and when the carry input is 1 we are getting the Cout to be 1 here.

Only for these two conditions the carry output is 1 and it is inherently generating a carry because of A and B both of them are 1. The generate bit or whatever the G is nothing but,

$$G = A \cdot B$$

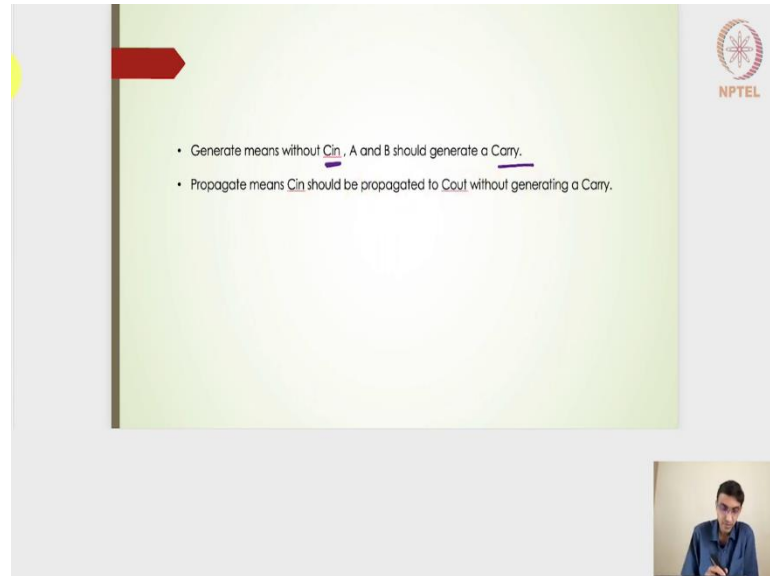
Inherently it generates a carry that is when the generate bit is 1, the propagate bit is 1 when we say that the Cin is propagated to Cout without inherently the A and B is generating a carry.

That is possible only in this particular two combinations, when Cin is 1 and Cout is 1, it is propagated to 1 and then when Cin is here this 1 is propagated to the carry output as 1. The only because the A and B here either of them is 1, that is when and then Cin is 1 then that is how the Cout is propagated.

For the propagate signal it is these two combinations is 0, this is 0 because anyways Cout is 0 and this is 0 because A and B is inherently generating a carry that is why we say that it is not propagating rather it is generating. This is 1 and 1 because, here the Cin which is

generated in the previous lower bit the generated carry is kind of propagated in this present bit.

(Refer Slide Time: 09:34)



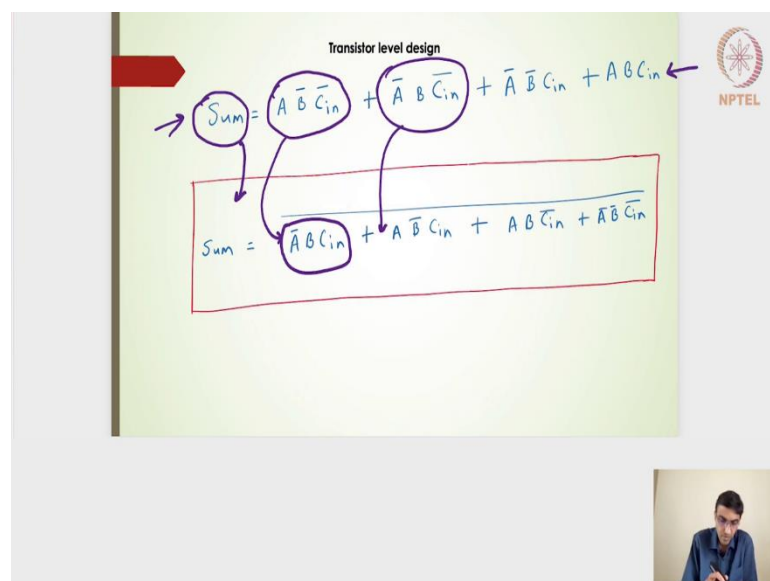
NPTEL

- Generate means without  $C_{in}$ , A and B should generate a Carry.
- Propagate means  $C_{in}$  should be propagated to  $C_{out}$  without generating a Carry.

NPTEL

Because, my A or B one of them is 1, that defines the generate and then the propagate bit which will be useful which will use it to design the higher order adder systems. Generate means without  $C_{in}$  A and B should be able to generate a carry and then propagate means  $C_{in}$  should be propagated to  $C_{out}$  without generating carry.

(Refer Slide Time: 09:53)



Transistor level design

NPTEL

NPTEL

$$\text{Sum} = \overline{A} \overline{B} C_{in} + \overline{A} B C_{in} + A \overline{B} C_{in} + A B C_{in}$$
$$\text{Sum} = \overline{A} B C_{in} + A \overline{B} C_{in} + A B C_{in} + \overline{A} \overline{B} C_{in}$$

Hope that is clear. At a transistor level how do we design this 1 bit a full adder circuit, the sum bit is given by this particular expression of,

$$\text{Sum} = \overline{A}B\overline{C}i_n + \overline{A}B\overline{C}i_n + \overline{A}B\overline{C}i_n + ABCi_n$$

The adder circuit as such and if I consider the sum and then a carry out it is nothing but it is a mirror topology. Mirror topology in the sense, if I actually get whatever is there in the pull up side the same thing can be done in the pull down side or vice versa. This particular expression can also be written as within the CMOS technology if I want to have a pull down circuit.

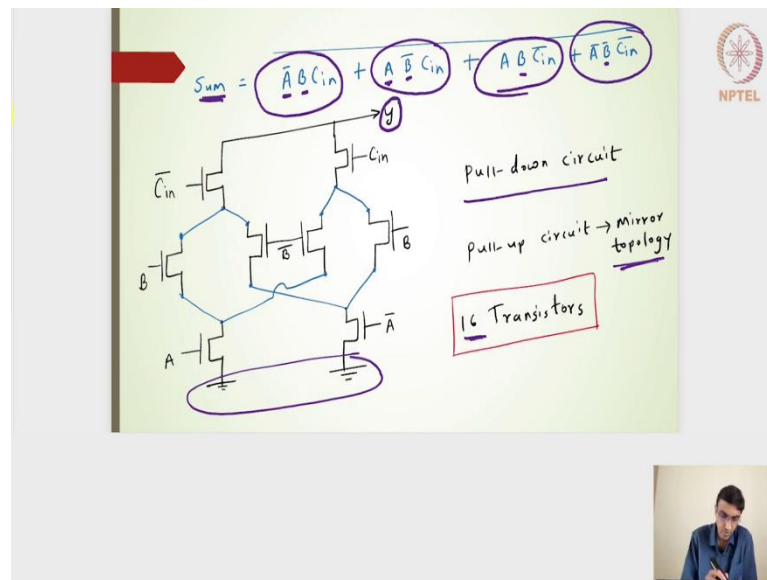
Then I will get a complement of that and this  $\overline{A}B\overline{C}i_n$  when it goes under a complement bar under an inverted bar. I will actually get a inverted topology here, that will be instead of  $\overline{A}B\overline{C}i_n$  it will become  $\overline{A}B\overline{C}i_n$ , it becomes kind of an inverted expression of this.

The  $\overline{A}B\overline{C}i_n$  will become this particular expression will become  $\overline{A}B\overline{C}i_n$  and then,  $\overline{A}B\overline{C}i_n$  will become  $\overline{A}B\overline{C}i_n$  and  $ABCi_n$  will become  $\overline{A}B\overline{C}i_n$ .

$$\text{Sum} = \overline{\overline{A}B\overline{C}i_n + \overline{A}B\overline{C}i_n + \overline{A}B\overline{C}i_n + \overline{A}B\overline{C}i_n}$$

This particular sum is equal to the inverted expression of the individual expressions of the sum expressions that is been written here and then it becomes a complement of that. The advantage of complement is we can easily have this circuit design the circuit in the pull down side and then take the output as directly the complement of whatever we have given here.

(Refer Slide Time: 11:47)



The sum is nothing but this particular expression which we have seen and now how do I actually design this at a transistor level. The transistor level we will take A and  $\bar{A}$ . I will design this transistor as A and  $\bar{A}$  and then, we will have  $\bar{B}$  here B here  $\bar{B}$  here.

I need four such transistors. I will have B and  $\bar{B}$  here on the extreme sides and then  $\bar{B}$  with input given to the two transistors. If I connect either A to here or  $\bar{A}$  to here it will be actually be A in series with that of  $\bar{B}$  or A in  $\bar{A}$  in series that of  $\bar{B}$ .

Finally, if I want  $Cin$  and  $\bar{C}in$ , so I will take you know the two transistors here with one of them as  $\bar{C}in$  the other one is  $Cin$ . I can easily connect in series between this 1 2 3 4 5 6 7 8 transistors and then get this 4 different expressions. Which will give us this pull down output.

If I connect this A to that of B and then B to that of  $\bar{C}in$ , I will get  $AB\bar{C}in$  expression. The  $AB\bar{C}in$  is done and if I connect A with that of this one  $\bar{B}$  and then this gets connected to  $Cin$ . I will get  $A\bar{B}Cin$ , this one gets now we have arrived at the transistor level connection to get this particular expression.

This two we will see with  $\bar{A}$ . The  $\bar{A}BCin$ , this particular connecting these two transistors and then this two transistors by this blue wire. I will get this particular expression  $\bar{A}BCin$  and then finally  $\bar{A}\bar{B}Cin$ . I have connected this particular blue wire and then this particular



blue wire completing this expression, having this completes our pull down design, now the pull upside is nothing but a mirror topology.

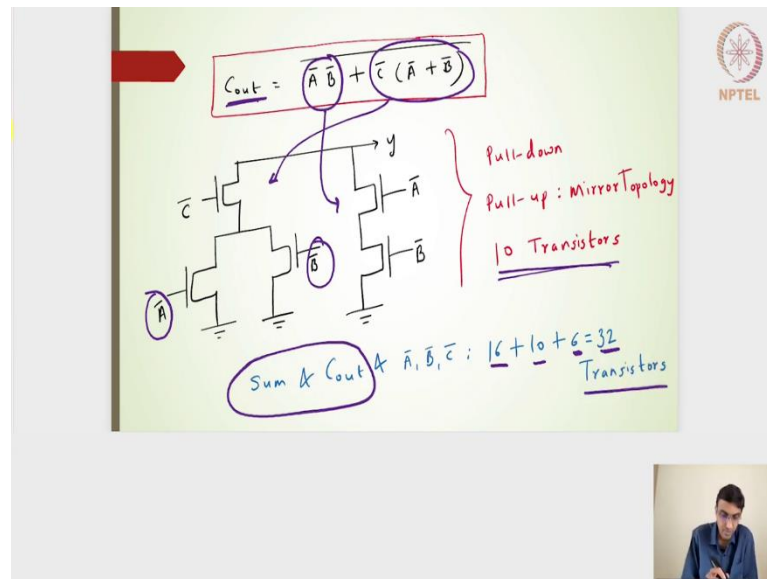
I will have the same level of same transistors  $\overline{C_{in}}$   $C_{in}$  and  $B \overline{B}$   $B$  and then  $A$  and  $A$  bar on the pull up side which will be connected to the ground rail instead of which will be connected to the  $V_{dd}$  rail instead of the ground rail. Then, I will do the similar connections and then the inputs will be similar  $A \overline{A}$   $B \overline{B}$   $B \overline{C_{in}}$   $C_{in}$ .

I will do the similar connections on the pull upside and then I will achieve the same expressions. In fact, I will get the same expression here. The sum will be nothing but the complement bar or rather the bar here and then this particular four expressions which will be the output.

The output here if I complete the pull upside I will get the  $y$  as nothing but the sum output. Remember that generally in a conduction complement topology what we will do is, if I have two of the transistors in series in a conduction complement topology in the pull up side I will have the two of the transistors in parallel. Whereas, in a mirror topology here it will be the same topology which goes onto the pull up side.

Then the reason is nothing but if I actually change this instead of  $A \overline{A}$  I will change it to  $A \overline{B}$  to  $B$  to  $\overline{B}$  and  $C_{in}$  to  $\overline{C_{in}}$  I will actually get the same expressions, then that is the reason why the mirror topology is kind of very useful for the adder as well as the adder circuit especially to extract the sum and then the carry output bits.

(Refer Slide Time: 15:32)



$$C_{out} = \overline{A\bar{B}} + \bar{C}(\bar{A} + \bar{B}) = 16 + 10 + 6 = 32 \text{ Transistor}$$

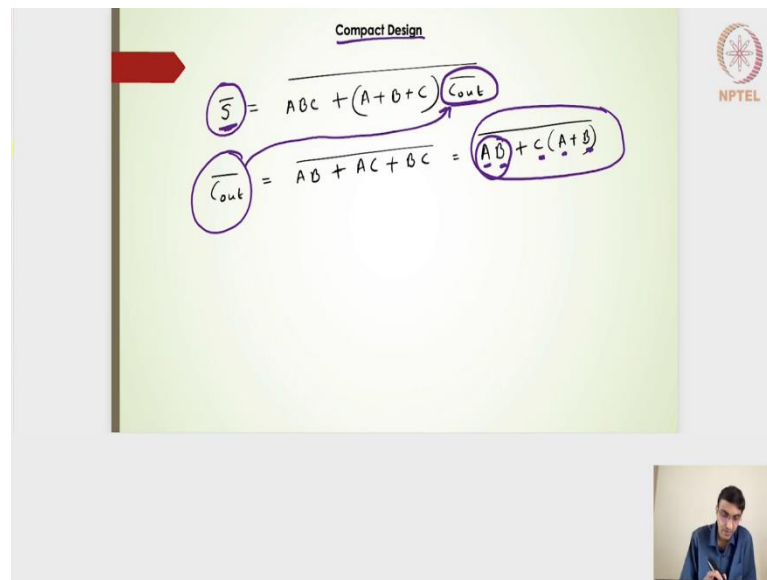
Going forward the carry output here, with this particular expression we will get  $A$  and  $\bar{B}$  in series. This completes this particular expression  $\bar{A}$  and  $\bar{B}$  and then,  $\bar{C}$  in series with that of  $A$  and  $B$  which are in parallel. The  $\bar{A}\bar{B}$  in parallel with that of in series with that of  $\bar{C}$ . That completes this particular expression.

I will have 5 transistors on the pull up pull downside and then I will have the similar mirror topology. I will have to complete this  $C_{out}$ , I will have the 5 transistors on the pull upside with a similar arrangement and then we will get the  $C_{out}$  expression, which will take around a 10 transistors.

Remember that this  $\bar{A}$  and then this  $\bar{B}$  and then this and then in the previous sum expression we had  $A\bar{A}B\bar{B}C\bar{C}$ . I need to get the complement of the inputs. The complement of the inputs means inputs are  $A B C$  and the complement of that will be  $\bar{A}\bar{B}$  and  $\bar{C}$ . I will require an inverter circuit.

For each of them I will require 2 transistors, for 3 inputs I will require 6 transistors. The total number of transistors that will be required to extract that the sum and then the carry output for 1 bit full adder design will be nothing 10 transistors for the  $C_{out}$  and then, 16 transistors for the sum. Then the 6 transistors for getting the complement of this the inputs, I will get actually get need to design 32 transistors.

(Refer Slide Time: 17:34)



The 32 transistors it turns out to be very large for a single 1 bit full adder. If I want to actually do for a 32 bit adder system it will be 32 transistors multiplied by 32 pretty large. Now, can we actually save some of the transistors can we make it little bit more compact.

One such design is this particular compact design, where we can actually write generate,

$$\bar{S} = \overline{ABC + (A + B + C)\bar{C}_{out}}$$

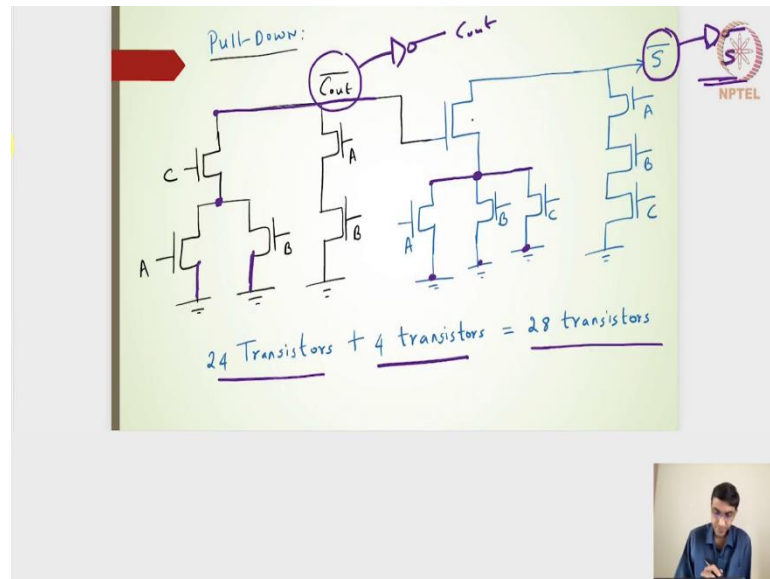
$$\bar{C}_{out} = \overline{AB + AC + BC} = \overline{A\bar{B} + C(A + B)}$$

To extract Cout I will have an inverter here from the  $\bar{C}_{out}$ , but nevertheless if I can do the  $\bar{C}_{out}$  and then take an inverter and then get the Cout and if I can get some bar and then take the inverter and then get the sum and having a in the number of transistor designs if it drastically reduces or whatever the number it reduces, if it is still less than 32 it is worth pursuing them.

Remember that A B here will be nothing but two transistors in series and C transistor will be in parallel with that of A and B transistors and then the output of that will be the  $\bar{C}_{out}$ , this sum the complement sum expression or the sum bar expression is written as the whole bar and not only that it will be  $AB + AC + BC$  the handling with that of the  $\bar{C}_{out}$  which is generated here.

This  $\overline{C_{out}}$  goes here, this  $\overline{C_{out}}$  will go to a transistor and then this transistor on the pull down side which will be in series with the three of the transistors in parallel and then this will go in parallel with that of three of the transistors in series and then we will get the sum bar output. Once we get the  $\overline{C_{out}}$  and then a sum bar output we will be able to connect to the inverter and then get the sum output and then the  $C_{out}$  output.

(Refer Slide Time: 19:55)



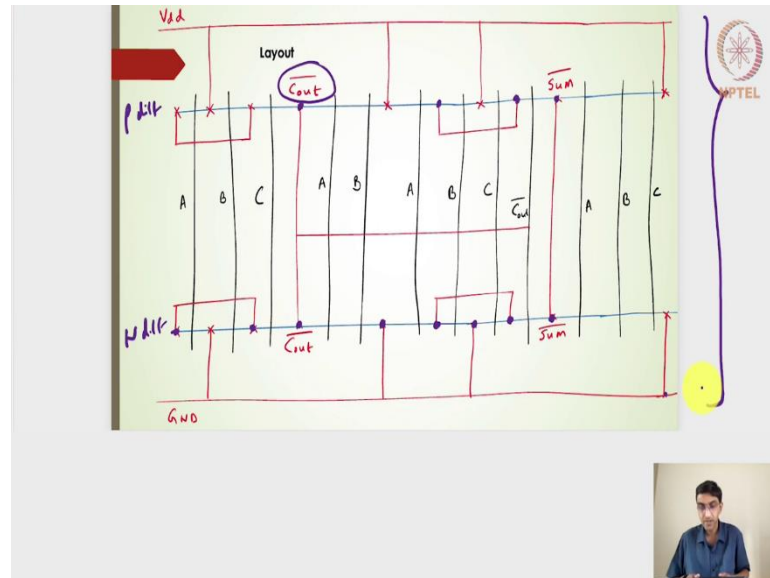
What I meant was we will first generate the  $\overline{C_{out}}$  and I have written only the pull down circuit the pull up side will be exactly the mirror topology. It will be the similar transistors and then the mirror of that which will come on to the pull up side. On the pull down side as we said C will be C transistors in parallel with A and B and then here the A and B will be in series we get connected to the  $C_{out}$  and then this particular line will be  $\overline{C_{out}}$ .

The  $\overline{C_{out}}$  will go to a transistor which will be in series with that of three of the A B C transistors in parallel and this will go in parallel with that of A B and C which will be in series. I will be able to generate sum bar I will be able to generate the  $\overline{C_{out}}$ . If I now try to calculate the number of transistors it will be 1 plus 2 plus 1 2 3 4 5 and then 6 7 8 9 10 11 12.

Then, 12 transistors on the pull downside and then 12 on the pull upside we will get a 24 transistors to get  $\overline{C_{out}}$  and  $\overline{S}$  and then if I connect an inverter here to generate a  $C_{out}$  and then I connect an inverter here to generate this sum I will have to spend 2 transistors here 2 transistors here.

I will have two add 4 more transistors and finally, within 28 transistors I get the Cout and then the sum expression. The Cout logical output and then sum logical output we will be able to extract.

(Refer Slide Time: 21:24)



With that 28 transistors. Now, just to complete the 1 bit full adder using a 28 transistors I have just drawn the layout diagram here. The stick diagram basically, and then the stick diagram and then this I have completed both the PMOS side as well as the NMOS side, that means that the N diffusion line here.

Then the P diffusion line here and then this is the  $V_{dd}$  rail and then this is the ground rail. Just to get back into the transistor level diagram, first we will be generating the  $\overline{Cout}$  and then going and ahead and then generating the sum expression. It is similarly in the layout also we will start from the left side here and then go to the right side of the transistor design.

Looking at the left side this will be nothing but the C transistor the polysilicon of the C will go here and then the diffusions of here one will go to the Cout the other one will be going and connecting to the A and B transistors which are in parallel. The other diffusions of A and B are connecting to the ground.

If I want to do that in the stick diagram. I will have this A and B A and B transistors and especially if I have drawn for the pull down side and I said that it will be a mirror topology

mirror replica image of the on the pull up side. Let me have a look at the N diffusion and then the P diffusion will be nothing but the similar thing similar connections the only thing is the connections will be to the  $V_{dd}$  rail and here it will be towards the ground rail.

Looking into the N diffusion lines here if I have A and A and B polysilicon on one side it gets connected to the ground both A and B transistors on the other side it gets connected it is get connected to the C transistor. I will have the A transistor which gets connected to the C transistor here, the C diffusions and then, anyways B is anyways having the merge diffusion with that of the C.

I have this connection and then this connection, that completes this particular portion. Then the next portion is A and B which will be in series and then connected to the ground. I will have the A and B polysilicon on one side it is connected to the  $C_{out}$  and then the other side it is the ground. This completes the  $C_{out}$  on the pull down side and the pull up side it will be the mirror replica image of the same thing what we have done here because the transistors will have the mirror replica image.

The only thing is instead of ground it will be connected to the  $V_{dd}$  rail and then, we will be left with this point and then this point. We will need to have a kind of a metal connection here metal line here which will be connecting and then saying that this will be the  $C_{out}$  line or rather the  $\overline{C_{out}}$  line.

This completes my  $\overline{C_{out}}$  line. The  $\overline{C_{out}}$  line actually goes to a transistor here. That is why I have drawn this  $C_{out}$  metal line which gets connected to the polysilicon of  $\overline{C_{out}}$ . Let me go quickly and have a look at the transistor level. This will be the transistor the  $\overline{C_{out}}$  line which is connected to the transistor or the gate of the transistor here.

Then we will have A B and C in parallel on one side of the diffusion for all the three transistors it is getting connected here and on the other side it is actually the ground. If I look into the stick diagram here I have A B and C on one side it is getting to the diffusions of the  $\overline{C_{out}}$  transistor, this is that transistor where C is anyways having a merge diffusion.

The A and B is merge diffusion is getting connected to this the same the diffusion of this  $C_{out}$  transistor. All the 3 transistors are connected on one side it is connected to the  $\overline{C_{out}}$  diffusion and all the three transistors will go and connect to the ground. We will have the

A transistor here which gets connected to the ground and the B and C which is having a merged diffusion which will get connected to the ground.

Similar mirror topology will come onto the pull up side, we will have these two connected or rather these two connected here and then this will get connected to the  $V_{dd}$  rail. This completes the A B Cin parallel with that of the Cout transistor which will be in series. The final one is the A B Cin series.

The C transistor will get one side it is connecting to the B diffusion and on the other side it is the ground, the A transistor on the other side it is the sum bar output. If I look into this A B C I will have the  $\bar{C}$  the one of the diffusion  $\bar{C}$  is connected to the ground and then similarly on the pull upside it is connecting to the  $V_{dd}$  and A B C on B side on the A side the diffusion is getting connected to the  $\overline{\text{Sum}}$ .

Now we have this sum bar on both the pull up side and then the pull up side that needs to be connected by a better line. This completes the layout or the stick diagram of the 1 bit full adder and we can easily calculate or evaluate what is the area that it occupies. This particular stick diagram on the layout form is also available in the very first page of the of your textbook which you are referring which is the West and Harris textbook. This gives what kind of digital design digital subsystem design blocks we will be referring to from here on.