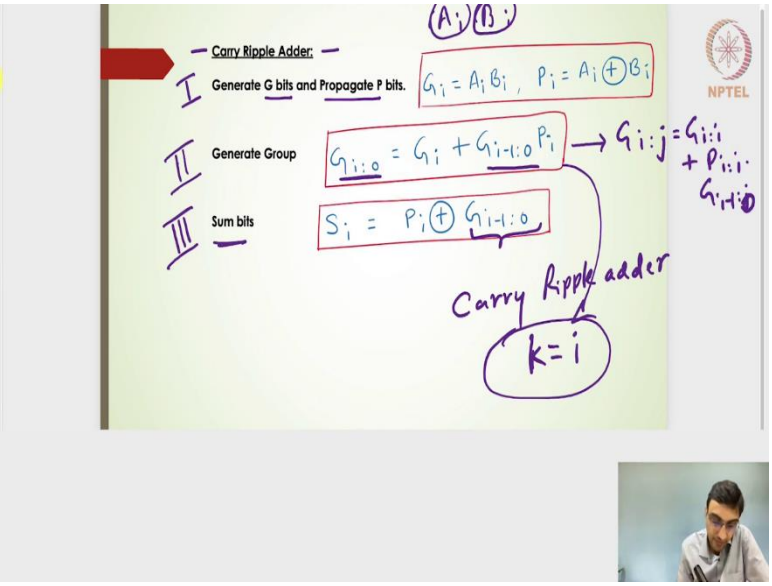
Design and Analysis of VLSI Subsystems
Dr. Madhav Rao
Department of Electronics and Communication Engineering
International Institute of Information Technology, Bangalore

Lecture - 88
PG architecture - Part2

(Refer Slide Time: 00:16)



For a carry ripple adder which we had seen what it really means is the carry output of the lower LSB side is going on propagating to the next subsequent bit and that is carryout will be then the propagated to the next subsequent bit addition and then so on till it reaches the MSB bit.

The MSB the last sum bit in terms of a 16-bit addition S_{16} bit it is actually depends on the carryout of the previous addition and then the whatever the carryout depends on the previous to previous addition carryout and then so on. That is why it is called as the carry is kind of rippling through the adder and that is why it is called as a carry ripple adder.

The carry ripple adder if you want to construct it using the group generate and propagate its bits, the group as well as the bit level generate and propagate signals then how do we do it. There are three stages, the 1st stage is generating the bit level generate G and propagate P bits, the 2nd level is generating the group generate signals and the 3rd level is the generating the sum extracting the sum bits.

For the individual or the individual bit level generate and propagate signals we can use this particular expression,

$$G_i = A_i B_i, \quad P_i = A_i \oplus B_i$$

$$G_{i:0} = G_i + G_{i-1:0} P_i$$

$$S_i = P_i \oplus G_{i-1:0}$$

This satisfies our expression of Gij where k is nothing but

$$G_{i:j} = G_{i:i} + P_{i:i} G_{i-1:0}$$

This satisfies our original expressions of the group generate signals starting from the i-th bit to the j-th bit where here for the carry ripple addition we are going to use k value as nothing but the i itself. In this particular for carry ripple addition or adder block we are going to use i is equal to nothing but or rather k is nothing but i.

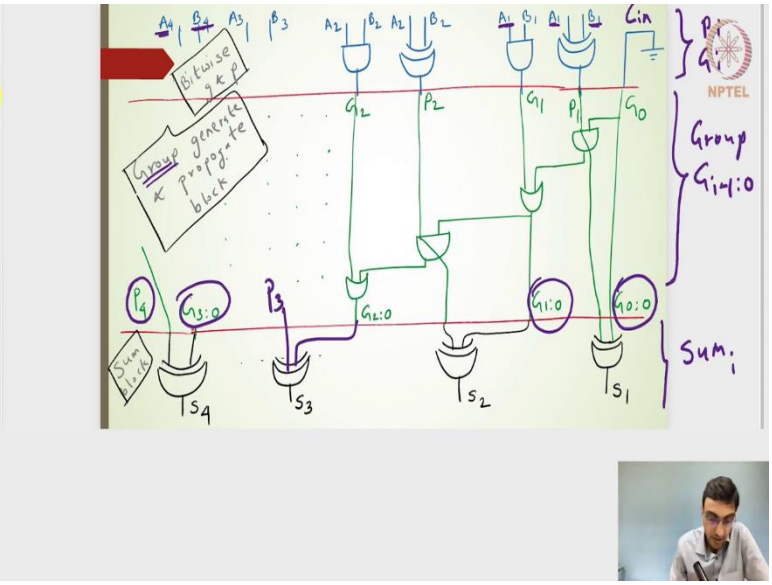
Then we will also see different cases where k is a different value than that of i, then what kind of an adder system we can develop whether it is a fast adder or whether it is a slow adder. But, for carry ripple addition we will use k is equal to i and for sum bit for extracting the sum bit it will be nothing but the propagate of that i-th bit XOR with that of the group generate of i-1:0.

With this particular three stages, that we will be able to extract the sum bits. Now, we need to design this individual stage and then find out the sum bits. Individual stages mean I will have A of i B of i values. If it is say 8-bit system or a 16-bit system I will have i starting from 1 to 16, B 1 to 16.

I will have all the 16 lines of the A input of the B input and then I will have to generate the individual bits G_i means G 1 to 16, G or rather P 1 to 16. Both of them will be generated in the 1st stage. In the 2nd stage we will be able to design a circuit such that it is going to generate 1:0, 2:0, up till 16:0 or rather 15:0 and then the carry out of 16:0, that will be the 2nd stage.

The 3rd stage is where we are going to design the S_1, S_2, S_3 up till S_{16} from the initial stage individual the propagate signals with that of the group generate signals which is created in the second stage, there are three stages of the design.

(Refer Slide Time: 05:03)



This is what it represents, this level we say that this is the bit level Pi and Gi signals design. This one is the group level $G_{i-1:0}$ extracting $G_{i-1:0}$ that is the design for that and last one is our sum bits.

If I look closer into it, I have taken A_4, four bits of addition A_1 here A_1 and B_4 to B_1 here and to generate the individual bit level generate and propagate signals that is nothing but G_1 and P_1, I need an XOR operation and I need an AND operation to get the G_1 and then to get P_1.

Similarly, for P_2 and G_2, I require a 2 input XOR and then 2 input AND gate and then similarly, for up till G_4 P_4. After this is done once we have the G_1P_1 P_2G_2, P_3G_3 P_4G_4, then I should be able to create $G_{1:0}$, $G_{2:0}$, $G_{3:0}$ and the way we will create $G_{1:0}$ is nothing but G_1 or with that of the P_1 and $G_{0:0}$.

The $G_{0:0}$ is nothing but our carry input which we have assumed it to be 0 here in this particular case. I will use this is $G_{0:0}$ and with that of the P_1 which is being generated in the first stage of the design. This particular AND output is going into an OR gate where the other input is G1 so as to get $G_{1:0}$.

Once we have $G_{1:0}$, this $G_{1:0}$ will be able to create $G_{2:0}$ which will go to the AND gate here where the other input is P_2, the output of this AND gate is going to the OR gate here where

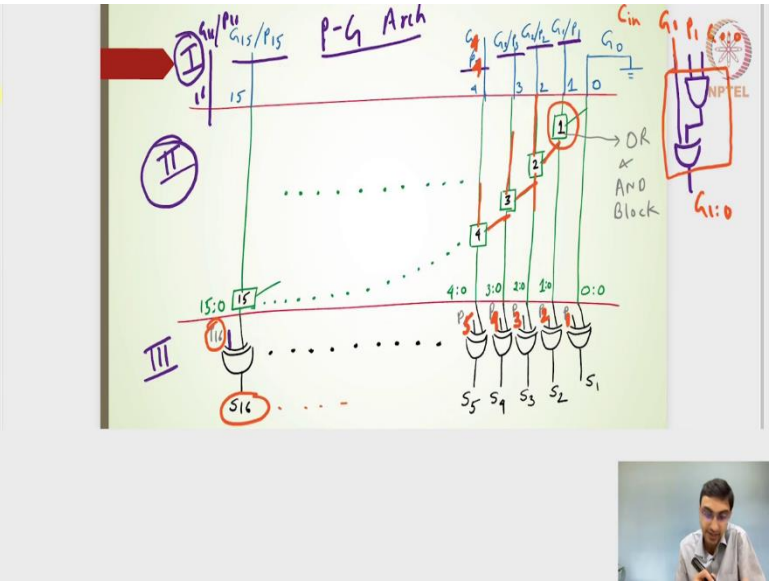
the other input is G_2. I will get $G_{2:0}$ and then similarly we should be able to create $G_{3:0}$ and of course $G_{4:0}$.

That completes the design for the group and generate propagate block here. This is basically the group block here. This is the bitwise block here, bitwise stage here, this is a group stage here. The last one is where we are extracting the sum bits. Remember that S_1 is nothing but P_1 XOR $G_{0:0}$ that will give me S_1. The S_2 is nothing but P_2 XOR $G_{1:0}$, S_3 is nothing but $G_{2:0}$ XOR with that of P_3 signal and S_4 is nothing but $G_{3:0}$ XOR with that of the P_4 signal.

That will give me S_4 bit, the last stage is nothing, but the XOR representation 2 bit XOR gate, that I will get S_4, S_3, S_2, S_1. Just to summarize this particular whole three stage consisting or drawing the P-G architecture which is nothing but the propagate and generate signal architecture so as to design then adder subsystem block.

Just to represent the first stage consist of AND, and XOR AND and XOR gate, the second stage consist of AND and OR gates, the 3rd stage consist of the XOR gates so as to get or extract the sum bits. The first stage is nothing, but AND and XOR, the second stage is nothing, but AND and OR, the 3rd stage is nothing, but the XOR blocks.

(Refer Slide Time: 08:55)



In this particular this is basically the P-G architecture in a very very simple form. Drawing the XOR gates, drawing the OR gates, drawing the AND gates and then drawing the XOR

gates in the three stages will become very very you know will requires a lot of space to draw those individual gate designs.

The P-G architecture representation says in the 1st stage I mean it draws the 1st stage, 2nd stage and then the 3rd stage in a very minimal diagram level or in minimal gate level. It represents or the focus is more on the 2nd stage the 1st stage and then a 3rd stage. The 1st stage is generating the bit level the group the generate and then the propagate signals.

Here we do not actually use an XOR or an AND gate or rather we directly say that the bit level G_1 P_1, bit level G_2 P_2, G_3 P_3, G_5 P_5 up till P_{15} G_{15} is being created. In fact, it will go to the G_{16} and P_{16} also that I will get the P_{16} signal here. I can also write G_{16} P_{16} and this will be my 16 bit which will give me the P_{16} bit here.

The 1st level 1st stage, we will instead of representing an XOR and then the AND gate for generating the bit level generate and the propagate signals we directly consider that it is a group or rather it is a bit level G_1 and P_1 signals up till G_{15} P_{15} is directly there.

These things are directly available that means, that it has to pass through the XOR as well as the AND gate and then all these things are generated. The first level we have the bit level generate and then the propagate signals. The 2nd level we are annotating we are labeling the AND gate and then the OR gate, we are annotating or representing the AND and then the OR gate in terms of a square block.

This square block represents one input here rather three inputs here where the one input is this particular input is nothing, but G_1, this one is nothing but P_1 and then this is $G_{0:0}$. If I consider this particular block, the output here will be $G_{1:0}$ that is what we have seen earlier. This representation of a 2 bit OR gate and a 2 bit AND gate is indicated by a block just for an easier representation.

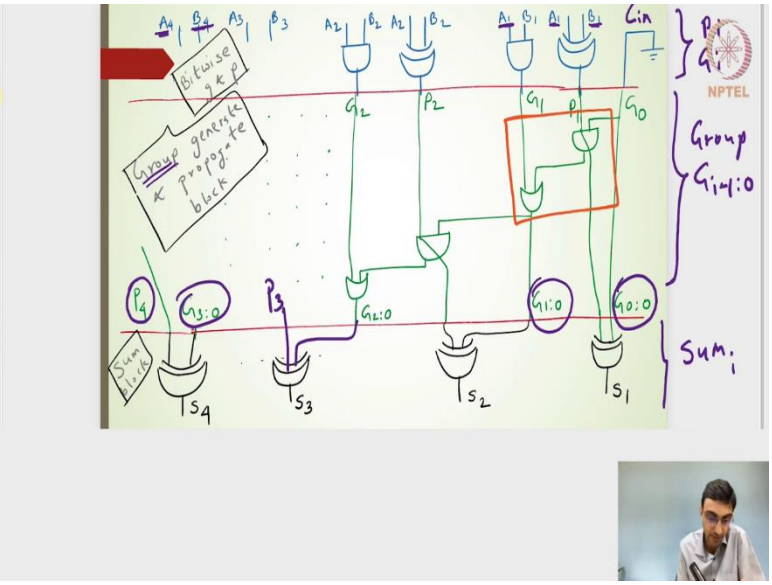
The first block represents the first block that has been utilized that the three inputs are G_1 P_1 and $G_{0:0}$. The $G_{0:0}$ is nothing but our C input to this particular 16 bit addition, we have this one block. The output of this is $G_{1:0}$, the second block where there are three inputs this particular line represents two inputs and then this particular line represents one input. This particular one input is nothing, but 1:0 line because this generates 1:0.

This particular line indicates two inputs where one is G_2, other one is P_2 generating 2:0. This particular the third block will be nothing but AND and OR gate where one input is $G_{2:0}$, the other two inputs are coming from here which is G_3 and P_3 and this particular block again represents an AND an OR gate where there are three inputs one input is 3 is to 0, the other two inputs are G_4 and P_4.

This is a very sequential design. What it really means is $G_{1:0}$ has to go to the block of 2nd to generate $G_{2:0}$, $G_{2:0}$ has to go into the block of 3rd which will generate 3:0. It will go into the block of 4 to generate 4 is to 0 and then subsequently it will go till this block of 15 block where it will generate $G_{15:0}$. The 15:0 will go into an XOR gate with P_{16} so as to generate S_{16} output.

Similarly, if I want to find out the S_1, S_2, S_3, S_4 up till S_{16}. The S_1 will have $G_{0:0}$ and then P_1 as an input, S_2 will have $G_{1:0}$ as one input and then P_2 as the second input, S_3 will have P_3 here and then 2:0, S_4 will have P_4 and then $G_{3:0}$ and then S_5 will have P_5 and then $G_{4:0}$.

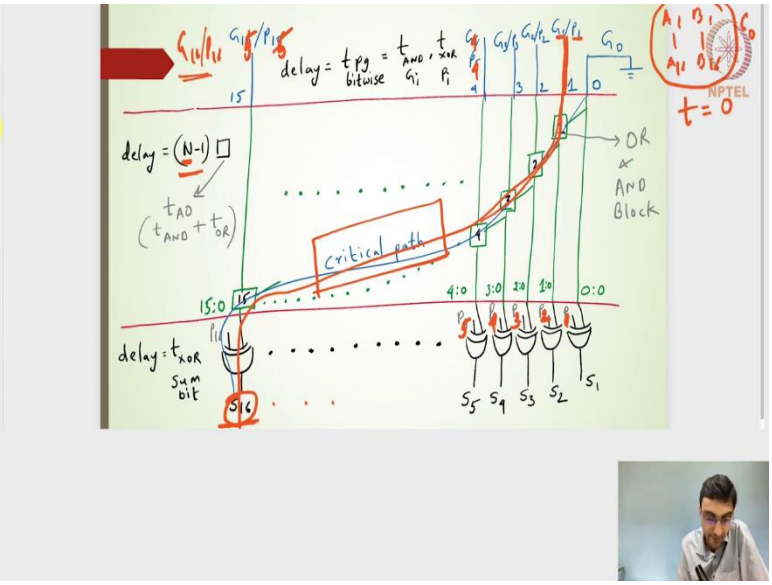
(Refer Slide Time: 14:14)



What this diagram the P-G architecture is nothing but it is a very simpler form of representation of the previous AND – OR block space. Here we have the bit wise, group wise and then the sum block. The three stages are now represented by either a straight line or this particular the AND and the AND and OR blocks are represented by a square block in the 2nd stage.

This is the P-G architecture where we consider the 1st stage we will have the bit level G generate and propagate signals and then in the 2nd stage everything the AND and OR gates are represented by a square blocks and the 3rd stage we can have the XOR gates generating the sum bits.

(Refer Slide Time: 14:58)



For a carry ripple addition what really happens in the 2nd stage is once this is generated once G_1 P_1 is generated it goes to the block 1. The output of the block 1 will go to the 2nd block, the output of the 2nd block will go into the 3rd block, 4th blocks subsequently to the 15th block so as to generate the S16 bit. Remember P this has to be P_1, this has to be P_2, P_3, P_4, P_5.

If I want to find out the delay starting from to extract the S_{16} bit starting from the A_1 B_1 to whatever A_{16} B_{16} that is being made available. What is the propagation delay? the propagation delay is always measured with respect to the critical path. Critical path is nothing but where we will have the maximum number of gates that is been there so as to get the outputs.

If I look into the maximum number of gates or the signals that passes to the maximum number of the gates, that it generates the output S_{16} output will be the last one to be generated. In fact, S_{16}, S_{15}, S_{14}, S_{13}, S_{12}, S_5, 4, 3, 2, 1 will be extracted or will be generated much much earlier than that of the S_{16} bit.

My critical path in this case is nothing but whatever is the signal that is been propagated so as to generate the S_{16} bit that will be the critical path. To generate the S_{16} bit it has to propagate through the 2 input XOR gate here and it has to propagate to the 15th block. It has to propagate 15 block input is nothing but the 14 block.

It has to propagate to the 14 block, 14 block input is nothing but the 13th block. It has to propagate so on, 4th block, 3rd block, 2nd block, 1st block and then whatever it takes to generate this G_1 P_1. We are here assuming that all A_1 to A_{16} and B_1 to B_{16} as well as C_0 or the C input are all available at time t = 0 and then trying to calculate what is the delay for this particular 16 bit addition. In that sense G_1 P_1, G_2 P_2, G_3 P_3 up till G_4 P_4 and then up till G_{16} P_{16}. No, this could be G_{15} P_{15} and G_{16} P_{16}. All of them will be simultaneously be generated because I will have 16 such XOR gates and 16 such AND gates generating the generate and then the propagate signals at a bit level.

Once this the A_1 B_1 to A_{16} B_{16} is available, then the critical path starting from the 1st stage to the 2nd stage and then passing to the 3rd stage will be nothing but this particular portion and then the XOR. If I actually look into the 2nd stage the delay is nothing but the number of the square blocks here. It has to go through the 15th block starting from the 1st block to the 15th block, that output of the 15th block will go to the XOR gate so as to generate the S_{16}.

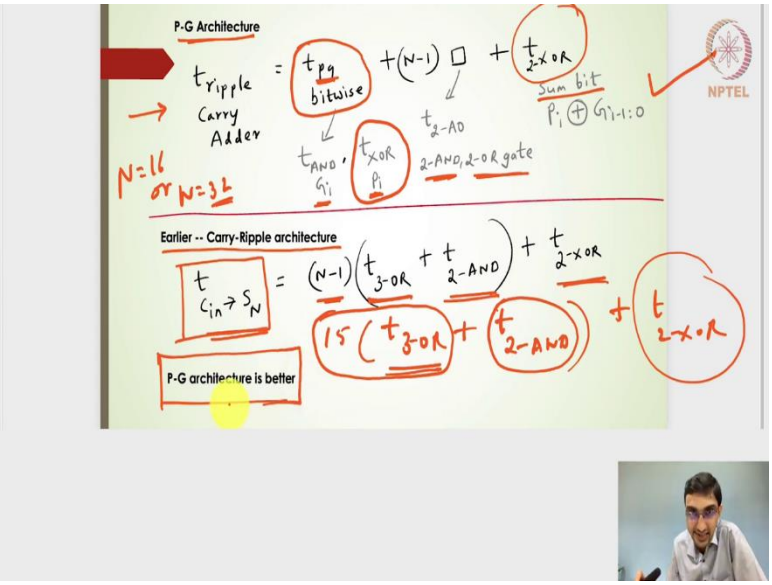
The 2nd stage if I want to find out the delay of the 2nd stage it will be nothing but 15 such blocks it has to propagate. If the number of addition is 16, if the number of addition is N, the delay for the 2nd stage will be nothing but N - 1 into the square block which is nothing but the AND and OR gates propagation delay.

That is what I have written. I have written N - 1 into the square block which is nothing but the square block represents the propagation delay of AND and OR. The AND plus the propagation delay of the OR gate. For the 3rd stage which is nothing but an XOR gate so that for the 3rd stage it will be t_{XOR} to generate the sum bit.

$$\text{delay} = t_{\substack{pg \\ bitwise}} = t_{\substack{AND, t_{XOR} \\ Gi \quad Pi}}$$

For the 1st stage it is nothing but the t_{AND} and t_{XOR} gate, whichever one is taking higher whether it will take G_1 or whether the P_1 will be delayed whichever one is having a higher delay that we have to consider.

(Refer Slide Time: 19:34)



For the P-G architecture the delay for this carry ripple adder is nothing but the P-G bit wise whatever is the delay,

$$t_{\substack{\text{ripple}\\\text{carry}\\\text{adder}}} = t_{\substack{\text{pg}\\\text{bitwise}}} + (N-1)\square + t_{2-xor}$$

The P-G bitwise will be nothing but the AND to generate a bit wise Gi signal or the XOR to generate the P_i signal.

Whichever one is longer that should be considered in the propagation delay of the bit wise P-G signals. The N - 1 we know that it is 2 input AND and then 2 input OR gates that is the block and 2 XOR gate that will create the sum bits, this is the overall expression. I can use this expression for designing N = 16 bit adder or I can use it to design the 32 bit carry ripple adder or N = 64 bit for the carry ripple adder.

Earlier what we had seen is a carry ripple architecture

$$t_{Cin \rightarrow Sn} = (N-1)(t_{3-OR} + t_{2-AND}) + t_{2-XOR}$$

It actually resembles very very similar. The only change here is if I look closely into the OR gate and an AND gate, if you look into the OR gate and AND gate it was nothing but 2 input AND gate and this was the 3 input OR gate.

If I consider the 16 bit addition it was nothing but 15 times 3 input OR gate + 2 input AND gate and of course + the 2 of XOR. If I compare with that of the P-G architecture here this will be 2 input XOR gate, this will be the same and N - 1, 15 turns out to be the same, but here the square block represents 2 bit AND gates and 2 bit OR gates.

Our 2 bit AND gate is there that is satisfying, but here it is 3 bit OR gate, here it is only 2 bit OR gate and if I consider the P-G bit wise one of them will be nothing but if I consider the XOR gate to be higher, I will have t_{XOR}. The P G architecture is going to give me t of XOR plus 15 of 2 bit OR gate. The other things are same.

The t of XOR plus 15 of t_{OR} gate here it is 15 of 3 OR gates 3 input or gates. This naturally will be higher. This representation where we had utilized the carry ripple architecture this is naturally going to be higher whereas the P-G architecture which utilizes the bit wise propagate generate signals and the group wise generate signals, and then use it to generate the sum bits is likely to have a better delay or a lesser delay.

The P-G architecture which utilizes the multiples of the XOR and then AND gates in parallel, this is likely to have a better delay characteristics.