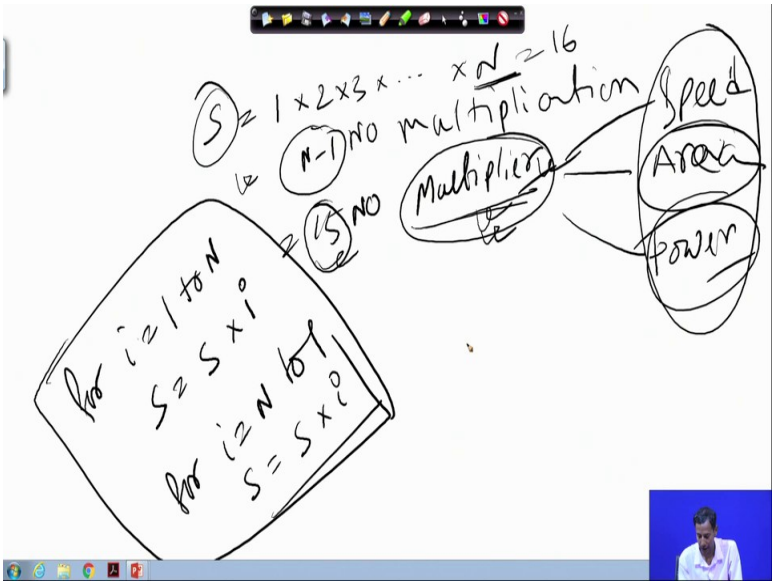
Architectural Design of Digital Integrated Circuits
Prof. Indranil Hatai
School of VLSI Technology
Indian Institute of Engineering Science and Technology, Shibpur, Howrah

Lecture - 09
Algorithm to Efficient Architecture Mapping (Contd.)

Welcome back to the course on Architectural Design of IC's. So, in the last class what we have seen? We have seen that control twos complementation circuit and then sum of N natural numbers. Initially, we have done that sequential circuit of that; that means, sequential implementation circuit for N natural numbers. We have seen then we have seen that how we can optimize the circuit more to in terms of or in order to get the benefit of area and power consumption ok. So, that we have seen in the class.

So, again another type of circuit implementation if I just want to implement; that is, suppose if I just want to do there what I need? I need sum of N natural numbers.

(Refer Slide Time: 01:13)



Now if I need product of N natural numbers; that means, if my S equals to 1 star 2 star 3 star up to N. So, if I want to implement this kind of function. So, at the time how can I do? If you see that it is something like same; that means, here also N minus 1 number of multiplications I need for parallel implementation or if I just want to compute it within 1 clock cycles. So, N if the N increases if N is let us consider 16. So, at that time I need 15 number of multiplier ok. So, if you I increase the N to a higher values.

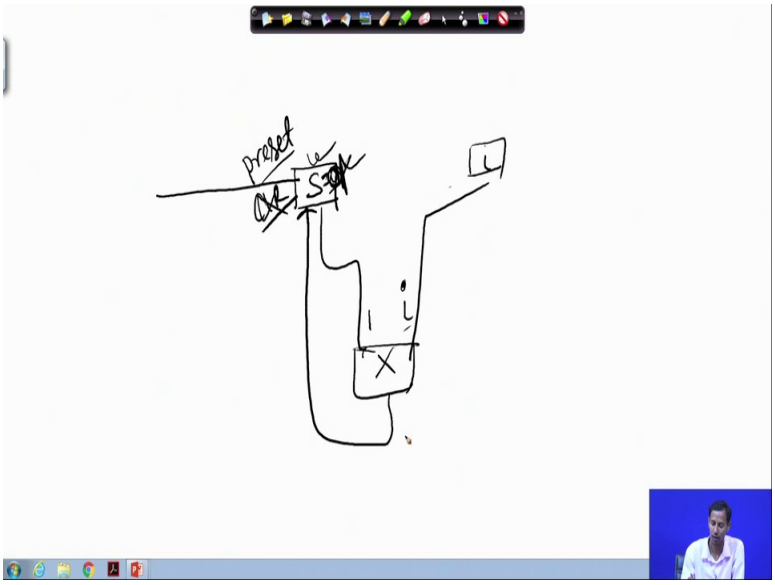
So, this value will also increase and as we know that this value of that means, the corresponding hardware cost for multiplier design that is on the higher side and for that reason there are several works or the several architecture for getting the reduced hardware or in terms of high speed or in terms of low power multiplier architecture or people have or there is a; that means, very in depth research on this particular multiplier architecture where I can get this that that means, this 3 things where speed as major concern or area as major concern or this power as a major concern.

So, we will come to that particular architecture what are the; I mean architectures available for multiplier designing in corresponds to this speed power and area that we will see later of this course ok. So, one particular module is on only for this multiplier design. So, that is not a part of at this point of this course ok, but the thing is that from this particular point, it is just the message that multiplayer is basically a that means, a circuit which cost a more area. So, that means the more number of a multiplier we use more the area and power we consume for that particular circuit ok.

So, that is why we are we have to that means, avoid or we have to reduce the number of multiplier architecture in a particular system or in a particular function implementation so that I can get the benefit of area as well as this power. So, how can I do that? Here also the same thing same logic whatever we have done in case of sum of N natural numbers. So, here that only the change is that so here all the logic will be the same.

So, the only logic will be that i equals to 1 to N S equals to S star i. So, initially we you can start with this logic or you can start with the same logic; i equals to N to 1, S equals to S star i ok. So, the same thing here you can do and here in this particular case, another change you require that was in the in the sum of N natural numbers, initially what we have done?

(Refer Slide Time: 05:16)

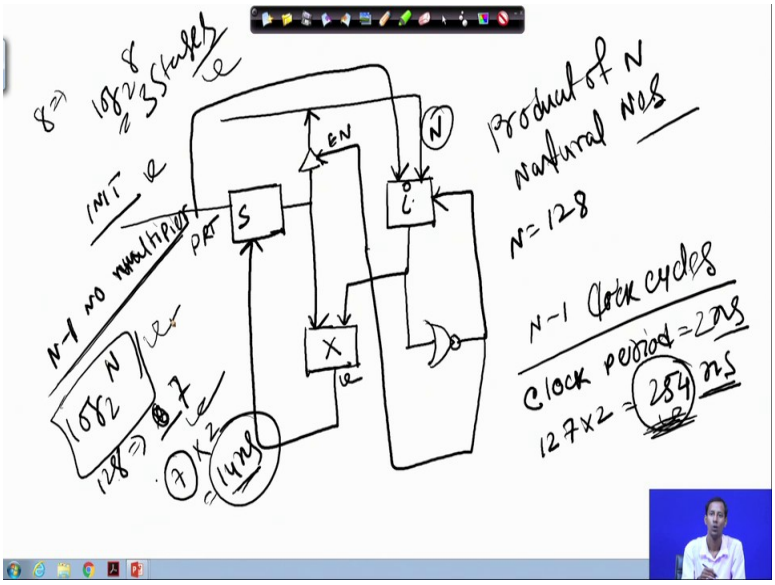


This S register is set to 0. So, initially S equals to set to zero for whenever we are finding the sum of N natural numbers, but here if you just consider this is i and this is basically multiplier here. So, initially if you set to 0, whatever cycles or how many cycles you do once you set to this two 0. That means, in again this will come to this.

So, that means 0 will be computed in each of this particular cycles. So, you do not have to set to 0 initially for this particular S registers; here initially it will be set to 1. So, there the initial signal that was clear for sum; but here this will not be clear this will be preset to that is the sum this registers value initially it will be 1. So, 1 and then whatever value is initially coming from this that will be i ok.

So, that is the only change otherwise the rest of the circuit will remain unchanged; it will be same as the sum of N natural numbers that means, if I just draw this circuit.

(Refer Slide Time: 06:42)



So, here this is the multiplier; again, this is the S this is the i ok. So then again, it is coming to here and i is coming here; then i is basically connected to this and here what I require and I need what? I need another control signal for this; this is the initial signal here it will be preset and this is the circuit for product of N natural numbers ok.

So, you see here what I require? I require only 1 multiplier, but the problem with this sequential implementation in the last class also we have discussed; that means, whatever the value of the; that means, this is the N. So, whatever the value of N is I have to wait for N minus 1 clock cycles.

So that means, the total competition time for this particular circuit suppose if the minimum clock period this, let us consider 2 nanosecond and N value is let us consider 128 ok. So that means, 127 into 2 that is 4 and 254 nanosecond, I require as a computation time for this particular circuit. That means I have to wait for 254 nanosecond; for to produce the results after applying the input over here, for the starting of this total computation ok.

So, if you have that liberty in the corresponding actually it depends on the application where, you are for which application you are designing the corresponding functions or corresponding of this expressions. So, if in one particular system you have the liberty that you can wait for 254 nanoseconds. So, at that time you can go for this particular implementation. But in some case suppose in a critical case, this kind of architecture you

require where you cannot wait. So that means, time is a that means, very much critical condition. So, at that time you cannot wait for 254 nanosecond.

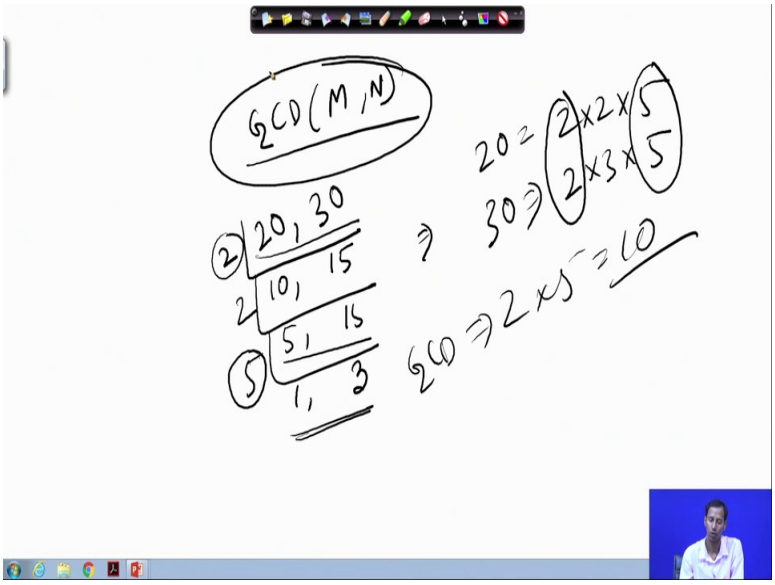
So, at that time if I just implement this particular logic using this N minus 1 number of multiplier; so at that time, what will happen? So, that can cost me suppose this is a multiplier 1 multiplier or this total 2 nanosecond. So, N minus 1 number of multiplier, I require and you log 2 base N that is the number of stages I require for implementing the parallel implementation. Why because if I am having 8 number I have 2 add.

So, at the time I will have log 2 base 8 means 3 stages right of adder or 3 stages of multiplier. So, here if as I have chosen this N equals to 128; 128 means that is 5 6 2 to the power 5 6 7 sorry 7 ok. So that means, 7 stage will be there. So, in each stage if I consider for each of the multiplayer, it is equals 2 nanosecond so that means total 14 nanosecond will be the time requirement for parallel implementation of this. So, hardware device this is reduced.

But that means, 14 nanosecond it is required means after applying the input once, after 14 nanosecond, I will get the output. But here I have to wait for 254 nanosecond. So, that means in one particular case, where I require the time as the critical one. So, at the time I have to go for this, I have to; that means, allow that ok. It will consume more area, but I can save the time here and in terms actually if I say in terms of here I am purchasing the time you with the cost of area and here I am purchasing the area in terms of this time.

So, these are the 2 different implementation techniques for one particular architecture or one particular circuit implementation ok. So, this is sum of sorry product of N natural numbers, then again we will see one another architecture or another function if I want to implement. Then how we can start with and then how can we build the architecture for that that we will see next.

(Refer Slide Time: 13:46)

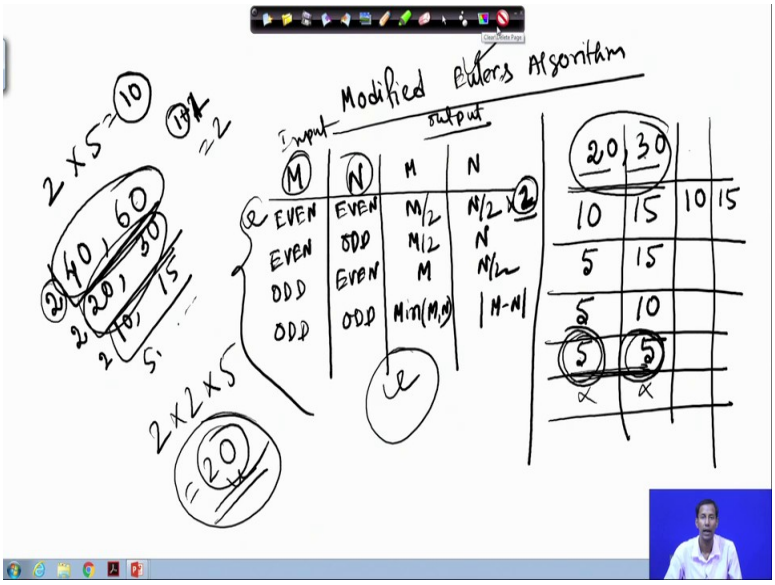


Suppose, I have to design 1 GCD function ok. GCD function means greatest common devisor among 2 numbers. So, I have to find out GCD of let us consider M and N. So, I think this is the basic mathematic operation in we have learned in our that means, in our high school days. So, GCD of M and N means suppose I am having 2 numbers ok. So, 20 and let us consider 30. So, initially how we can find out 2 here 10 and then this is 15; then here 2; this is 5; this is 15; 5, here 1 and here it is 3 ok. So, then what is the; that means, common term over here. So, 5 is common in both the cases and 2 is common for both the cases.

So that means, here or else if I just write it 20 equals to 2 into 2 into 5; that means, 4 into 5 and 30 just write it something like this 2 into 3 into 5. Then, what are the common factors among these two numbers that are 2 and 5. So that means the GCD value is 2 and 5 that is equals to 10. So, 10 is a number which basically the greatest number which can divide this 20 as well as 30.

So, I have to implement the circuit corresponding which will give me I will put the values of m and n. So, it will give me or it will produce me the GCD of that. So, how can I do that? So, for that I need to follow the modified Euler's algorithm ok; so according to the modified Euler's algorithm.

(Refer Slide Time: 16:03)



So,. So, it says that whenever this I am having 2 numbers M and N ok; when this is M is even and N is also even. So, at that time M will be M by 2; N will be N by 2. This is the input and this is the that means, in each stage how the out that means, value will change for even M and N. If this is even and this is odd then this will be M by 2 and this will remain unchanged. N will be same as N. If this is odd and this is even, so at the time it will be N and this will be N by 2. And both are odd, so at that time what will happen? This will be minimum of M and N and this will be mod of M minus N.

So, this is the algorithm for finding out the GCD. So, then we see if we see that let us consider the previous case 20 and 30. So, in both this case, what will happen? 20 and 30 means both are even. So, in the next it will be by 2 by 2. So, here it will be 10 here it will be 15; sorry, here if I just write and then next here it will be 10 and 15. So, then here it is even this is odd.

So, in the next what will happen? This will be 5 and this will be 15 ok. Then both are odd; both are odd means then this will be minimum of M and N. So, among M and N minimum is 5 and mod of m minus N is here it is 10. So, then again this is even; sorry this is odd this is even; so that means 5 and here, this will be 5 and when both the numbers are same at the time the computation will be over. So that means, as if I mean that M and N sorry both are same. So, at that time there will be no other computation.

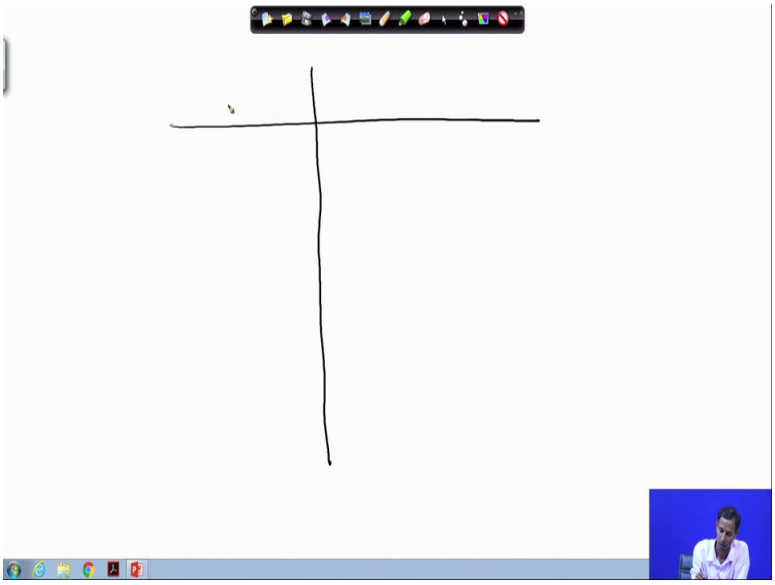
So, now according to this what I said that when both the numbers are even at the time only it will be multiplied with 2; the output values that will be multiplied with 2 ok. Output values means finally, whatever; that means, reminder is coming over here so that will be multiplied is with this 2 for finding each of this even/even case means what here how many times we are finding this even/even case, only for this particular case. So that means, only once I have to multiplied with this particular here for 5 and 5 this value is same; so that means, and how many times I am finding out even and even? That is only 1 times so; that means, this will be multiplied with 2 and I will get the results of 10.

Now, let us consider another case where I can find more of this ok. So that means, now if I consider another of if I consider another example of that so; that means, here if I consider 40 and let us consider this as 60 ok. So that means, for this particular case, initially it will be 20 and 30; then 10, 15 the same something 5, something like this. So, 20, 30 mean it will be like the same. So, for this particular case I am finding what? I am finding both are even. So, this is already stored 1 times.

So, again I am finding both are even so that means, now it will be 1 plus 1. So; that means, 2. So, 2 into 2 into 5 that is equals to 20 is the greatest common divisor among this two. If you see 40 and 60, the common device are among them is 20 ok. So, that means using this particular logic or this particular things I can get the GCD function of 2 numbers which is M and N. Now this is the logic ok. So, this is the algorithm description of finding out the GCD function of 2 numbers based on this particular algorithm.

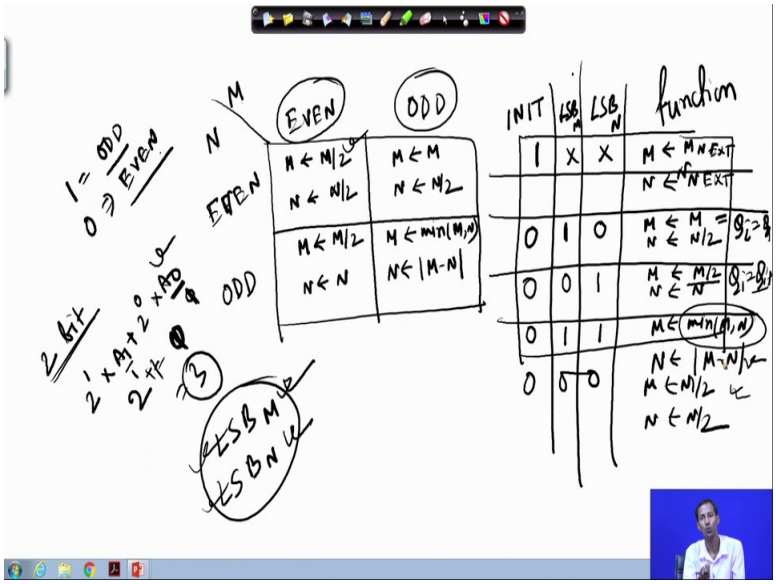
Now, I have to implement this as a circuit or architecture for this particular algorithm; how can I do that or where from I have to start with?

(Refer Slide Time: 22:33)



So, for that reason what I have to do? Initially I have to make the corresponding operation table ok. Sorry.

(Refer Slide Time: 22:41)



So, operational table means it is just same as this; so M and N. So, if this is even and this is for odd case; this is for even and this is for odd case Whenever M and N both are even, so at the time what happens? M that value equals to M by 2 and N that is equals to N by 2.

When M is odd N is even so at the time M equals to M and N equals to N by 2. When this N is odd M is even, so at the time M equals to M by 2 and N equals to N. When both are odd, so at the time M equals to minimum of M and N and N is mod of m minus N ok. So, this is the operation table for this particular algorithm right.

So, that means what I need to do I need to if I just plot it that means, I need to know about that means, what are the signals or how do I define that M and N value is even or odd, how can I know? If you see the LSB of M or N; LSB of any variables if that is 1 that mean that is the number is obviously, that will be odd; if the LSB is 0; obviously, that number will be even in binary number system. Why? Because at the LSB suppose if I consider 2 bit and how can I find the decimal number that is weighted basically representation. Suppose 2 bit if I consider that is 2 to the power 1 into A I plus 2 to the power 0 into A 0 ok. So, if A 0 bit is 0 means what? Sorry this bit is 0 means this is here you will get 0 and whatever suppose this is 1. So, you will get 2.

If this bit is 1 and 2 to the power 0 means 1. So obviously, you will get 1. So, it will be 2 plus 1 that is equals to 3; so that means, now from that particular point I can say if this bit is 0. Obviously, it will be even if this bit is 1. So, 1 will be added to whatever number is there. So, that will become odd ok.

So, that means for that reason if I consider only the LSB of M and LSB of N. For calculating these even and odd values I get that means, these now this LSB of M and LSB of N that will be the control signal for what operation I need at what point of time. That means, this will give me this will indicate that this is the even number or M is even or M is odd. So, we will find out based on this 2 particular function LSB of M and N.

So, that means this will be the control signal which will produce the corresponding functionality changes in each of the cycle ok. So, that means I require this initial signal LSB of M and LSB of N ok; here what this is the function. So, initially what will happen? Initially this is whenever I have to initialize.

So, that means at that time M will be loaded to M next and N will be loaded to N next sorry N next ok. So, this is for 1 cross 1 cases. Whenever this is 0 and this LSB of M equals to 1 and this is 0 LSB N equals to 0; so at the time what will happen? M equals to what M and N equals to N by 2. Then again 0, this is 0, this will be 0 and 1 M equals to M by 2 N equals to N and for 0 1.

One case this is sorry M equal minimum of M and N and N equals to mod of M minus N and for both the case for 0 0, it will be M will be M by 2; N will be N by 2 for 0 0 0. M equals to M by 2 and N equals to N by 2. So, for M equals to M means what? Basically it will consume it will that means, the input and output will be same that means, here Qi will be the Qi. And whenever this by 2 means what Qi will be Qi plus 1; that means, just discard the LSB position and then this i require N as the other require for other function this I need that means, other circuitry to produce the minimum M and N and mod of M minus N ok. So, this is the operation table.

Now it is the same way how we basically do earlier case we have find out the operation tables. So, from that operation table we have got the truth table. So, this is something like truth table. So, from that truth table now we will try to find out the expression. So, from that expression that expression will be converted to the architecture.

So, for today this is it. Then again, in the next class we will continue with this particular example.

Thank you.