

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

Lecture - 08
Algorithm to Efficient Architecture Mapping (Contd.)

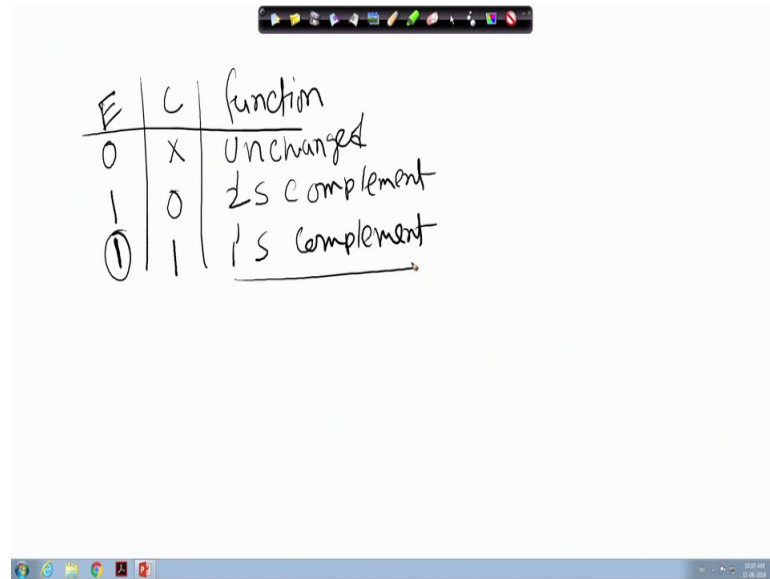
Welcome back to the course on Architectural Design of ICS. So in the last class we have seen that control twos complement circuit ones complement a circuit we have seen. Then we have seen control twos complement circuit; that means, we did in twos complement circuit how we can implement the ones complement circuit too. And after that we have seen sum of N natural numbers how we can do and we have drawn the circuit of sum of N natural numbers. That means, I can do it in different way; that means, I can do it parallelly or I can do it sequentially.

So, the parallel implementation of a sum of N natural numbers if the number of sum; that means, if I want to add more numbers, so at that time it becomes if I just do it in parallel manner, so at that time I require more of hardware; that means, more number of adders ok. So, for that reason whenever the N if I ask you to draw a circuit for N considering 1 0 2 4 or 2 0 4 8 number, so at that time it requires N minus 1 number of adders, so that means, as N increases, so that much of adder I require.

So, more the number of adders I am increasing to the circuit; that means, the consumption of the area and power that also becomes very very increases linearly with the number increases as it is N. So, for that reason what we have done we have done that sequential; that means, implementation of that algorithm.

So, we have seen that we have seen the algorithm and then we have develop the corresponding circuit for that. Let us go back to the corresponding. So, if I just want to add something; that means, suppose I just want to draw the previous circuit and let me tell you one thing. In the last class what we have that in twos complement circuit there was a silly things which is missing at that times. That is in the twos first I will start with that twos complement circuit and then we will come to that sum of N natural numbers. So, in twos complement circuit what we have seen that: suppose if I am having let us consider this thing.

(Refer Slide Time: 02:55)

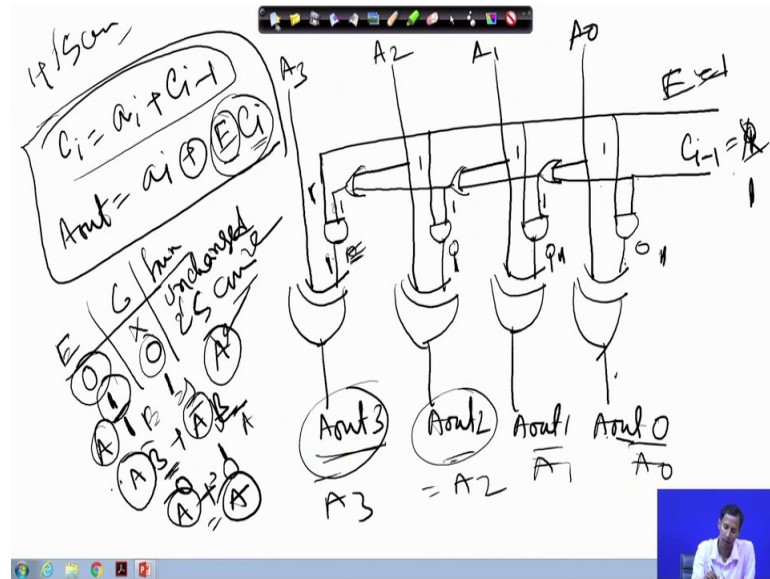


E	C	function
0	X	unchanged
1	0	2's complement
①	1	1's complement

So, this so if I consider this; that means, 2 control signal which is E and C and I need the functions that means here. Whenever this E equals to 0, so at that time the function; that means, the corresponding output will not be changed it will remain same as the input. So, whenever E is 1 and C is 0 at that time it will give me twos complement circuit. When E is 1 and C is 1, so at that time I will get ones complement circuit ok.

So, I think in the earlier; that means, class it was mentioned as 0, but it will not mentioned it is not 0 it is 1 ok. So, when both E and C are 1 so at that time it will give me ones complement circuit. The circuit will remain same. Now I will explain how it works basically.

(Refer Slide Time: 04:04)



So, the circuit was like C_i equals to A_i plus C_{i-1} and the A_{out} was A_i XOR with E C_i . So, this was the logic for designing the two complement controlled two complement circuit. So, if I just implement this logic, if I just drawn the circuit for this particular expression, so at that time how what I will get; that means, I need for 4 bit if I consider, so at that time how it will work at how it will look like so it will look like this.

So here this things, so at this particular I need 1 AND gate for this particular function, so each of this another input will comes from the AND gate as I am XOR with the corresponding bit position. So, this is A_0 and this is the one of the input of this particular AND gate that is this enable signal correct. So, where from then I will get this C_i . C_i basically is generated from this particular equation. So that means, for this as A_i is not there; that means, before hand of this A_i is not this directly it will come. So, from here I need the OR gate which will come to this each of these is connected with this OR gate ok.

So now, one of the input of this is this and another input of this is this; so here this to this; that means, the previous input along with the previous carry. So, here this things and this is the carry, so this is the C_{i-1} and this is $A_{out 3}$, this is $A_{out 2}$, this is $A_{out 1}$ and this is $A_{out 0}$. Now consider what are what you what was my functionality.

So functionality is if E is, if E is 0 and C whatever the value of C is the function will remain unchanged. That means, it will be same as input and output. So, if you put that,

that means, if this is the E so if E is 0 at that time what is the value of this each of this E is output is 0. That means, S in XOR gate we know that if any of the input is 0. So, at that time output is nothing, but another input right. So, that means, here if it is 0. So, at that time A3 will directly come to the A out 3 why because, if you see that what is the expression for XOR gate expression for XOR gate is $A \bar{B}$ plus $A \bar{B}$ corrects.

So, if I put that B equals to 0. So, that means, now this is A and plus this is 0 means this is 0 so this is A. So that means, if 1 of the input is 0 in XOR gate. So, the output will be same as the input, so as each of this bit is 0 when I put the value for E equals to 0. So, at that time the function remains. That means the input and output that are same. So, next is that whenever this at the time it is twos complement right. So, whenever E equals to 1, so that means, this of this input. That means, this bit is 1 and C i equals to 0, so at that time what happens.

So, 0 means here this bit is 0, so this will come to 0 ok. So, this is A 0 and then from here if you just see that each of this, that means carry generated from this particular position that is being forwarded and it is added to the corresponding. That means, it is ANDed and that means, what is the logic behind twos complement. That means, you just add 1 with the ones complement circuit right.

So, that we are basically doing whenever we are putting this E equals to 1 and C equals to 0 at that time we are just. That means, the carry generated from the previous the stage or previous section that is being forwarded to the left hand side and that is added or XOR with to get the corresponding twos complement circuit ok, and for if I need one, that means ones complement circuit.

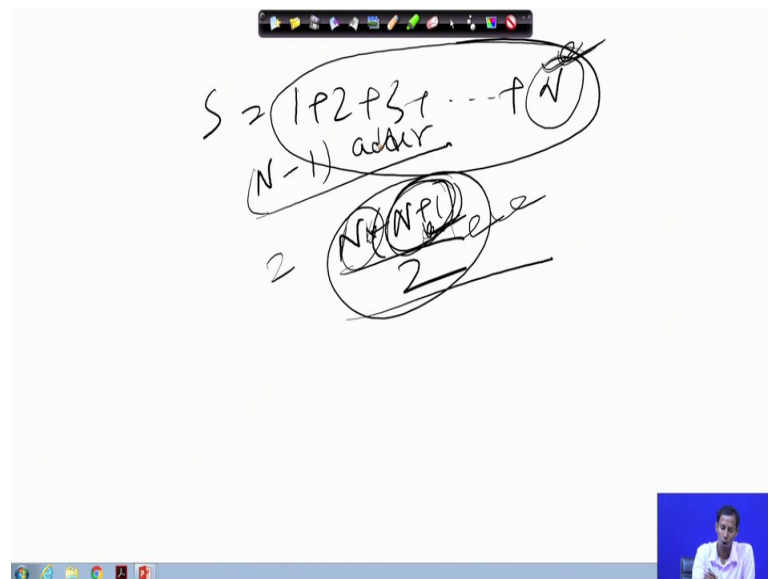
So, at that time what the control value I need that is 1 and 1. So that means, at that time instead of 0 this will be 1. So, ones complement circuit means in the previous class we have seen that ones complement circuit means if on instead of 0 if this bit output is 1. So, automatically this will be just inverted of the input. That means, instead of 0 here if I just put it 1. So that means, what will happen now this will be active. So, at that time XOR gates output that will be $A \bar{A}$ ok, so here I need 1 each of this should be 1.

So, E I have set already 1 so now, C i minus 1 that is equals to 1, so 1 and 1 this will be 1. So, it will be inverted so that 1 of the OR gate input is if it is 1, so at that time no matters what is the other input it will not depend on that if 1 of the input is 1. So,

automatically this will be 1 ok. So that means, now 1 and 1 it will be giving you 1 this will also give you 1 and this will also give you 1. So, each of each of these AND gate output if it is 1.

If 1, so at that time this A out will be just bar of each of the input ok. So that means, using this particular circuit now I can implement these twos complement circuit as well as ones complement circuit ok. So, in the next this was the; that means, functionality or this was the circuit for twos complement circuit and along with the ones complement circuit in the same module, right.

(Refer Slide Time: 11:34)



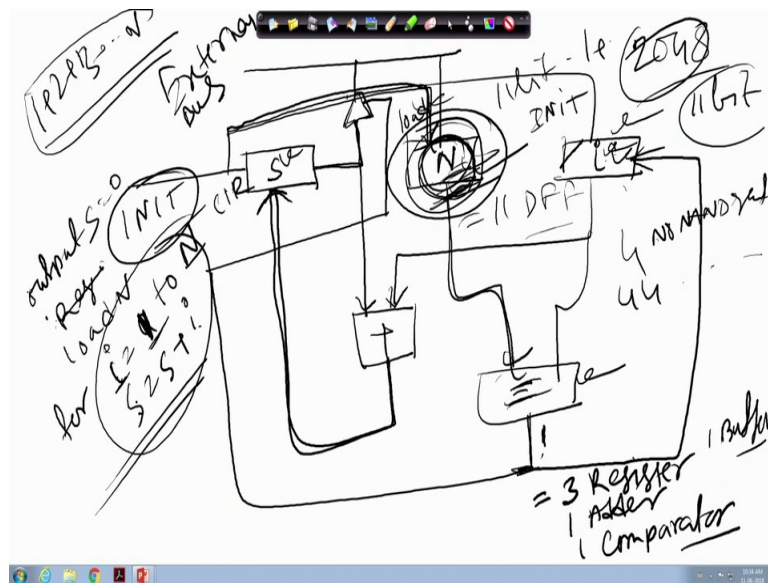
So, after that what we have seen we have seen one another example that is sum of sum of N natural numbers up to N ok. So, what I said that for this I require parallel implementation of that that requires N minus 1 number of adders and another implementations technique is that this expression also I can write as N minus N plus 1 by 2. So, this also here I require what, I require 1 incremental and I require 1 multiplier block.

So, when this N value is large at that time this will also give you the parallel implementation. That means, it will not depend for so much whatever is the value or whatever is the length of N. It will not wait for that much of clock, within 1 clock if I just want to add of, if I just want to get the result of this at that time I have to implement this kind of architecture or this kind of architecture. If I have, that means at that time what is

happening I need the circuit this related to. That means it is require more here I require multiplier and here I require incremter circuit.

So, but if I need, that means if my major concern is that area so at that time what I can do at that time, I can implement or I can I can follow the sequential implementation of that ok. So, sequential implementation of that at that time what I have to do, I have to write the algorithm for that. So, the algorithm we have seen and then corresponding algorithms architecture also we have seen.

(Refer Slide Time: 13:22)



So, what we have seen in the, that means in the last class. That means, I am having three registers where I am having S N and I, this three register I was having ok. So, and then I require one particular adder circuit right. So, from where this S and I value is basically added and then it is going back to the corresponding S circuit, and I need 1 circuit where I need to compare weather that is N and I both are same if it is same.

So, at that time what it will do it will go here and it will stop the corresponding counting this is nothing, but a counter. So, this will stop count or it will just initiate from the starting value again ok. And this will come here and it will say that I need. That means, I am having 1 buffer over here I was needing 1 external bus and this is the thing. So, this is the corresponding, that means whenever this basically this particular signal also says that basically as this is running in a loop.

So, at that time how unless and until these two values are same at that time the summation will be over ok so that means, the loop will break and it will produce the result at the output gate or I need output bus. So, for I need the output should be, that means the loop has done its job, so now you just produce the result at the output bus. So, how can I do that so this is the terminating condition? That means, whenever N and I value they both are same. So, at that time it will come so this that means this is just a single bit as this is a comparator. So, this is this will produce one single bit output, so single bit output will come here.

And it will activate this buffer that this now at that time this S will be gone to the external bus. So, this is the external bus and what I need, I need I am having this is initial value ok. So, here I think it will be clear here it will be load and here it will be N ok. So, for that reason what I need basically the algorithm was something like. That means, output is S and initially S equals to 0 set to 0 and this is 1. That means, oh sorry load the value of N then this is the logic for this is if I equals to sorry 1 to N then S equals to S plus I. That means, I need to add 1 plus 2 plus 3 plus something like up to N right.

So, that is why I will be changed to 1 to N and at that time S will in this case S will be added with the corresponding I right. That means, initially the I equals to 1. So, S value is initially I said S value equals to 0 so 1 plus 0. So that means, then again it will come back here 1 then 1 in the next cycle it will be 2. So, 1 plus 2 something like this it will just do it and then.

So, this is the logic for computing 1 plus sum of this particular N natural numbers. So, this particular is very much it will give you or it will produce you the corresponding whatever you desired to design. That means the sum of N natural numbers. So, my point is that here now can I optimize this circuit or can I that means from this particular algorithm can I see something; that means, I can reduce it more ok.

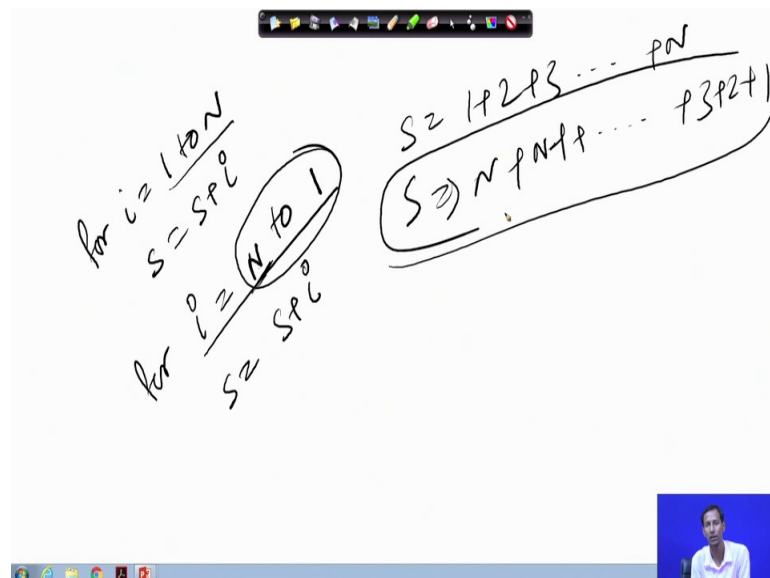
So, how can I do that here you see this particular N register that is once it is loaded. So, after that it is not doing anything it is just sitting idle. That means, once it is loaded then each of the cycle it is coming here and then it is just comparing the corresponding I value, why it is sitting idle. That means, in each of the clock cycle I is basically doing its jobs. That means, this is counter, that means I is now next 1 then plus then 2 then 3 then 4 then 5 something like this it is increment increasing in each of the clock, thus S

registers how it works in each of the clock it is by basically changing its value from the older value to the newer value newer sum value.

But this N is not doing only at the that means, starting of this INIT signal it is loaded from the external, but bus what is the value of N and then it is sitting idle that that value for N that is not at all changing in each of this clock cycles. So that means, can I reduce or can I remove this N. So, remove why I need to remove it because if this N value is let us consider that means, 2048. So, 2048 means I need 11 bit for considering this 2048. So that means, I need eleven bit registers over here.

So, if I can reduce this registers That means, obviously, I can reduce the number of transistors. That means I need 11 bit sorry 11 D flip flop. So, if each of the D flip flop requires 4 number of NAND gate. So that means, total 44 number of NAND gate I can reduce ok. So, in terms of area or in terms of power also I can get the benefit. So, that at that point looking at that particular point now my. That means intention is that can I optimize this particular circuit keeping the functionality same. So, at the time there will be a small change in this particular section. So, what will be the change if I just. So what was the logic initially for $I \text{ equals to } 1 \text{ to } N$ $S \text{ equals to } S \text{ plus } I$.

(Refer Slide Time: 21:02)



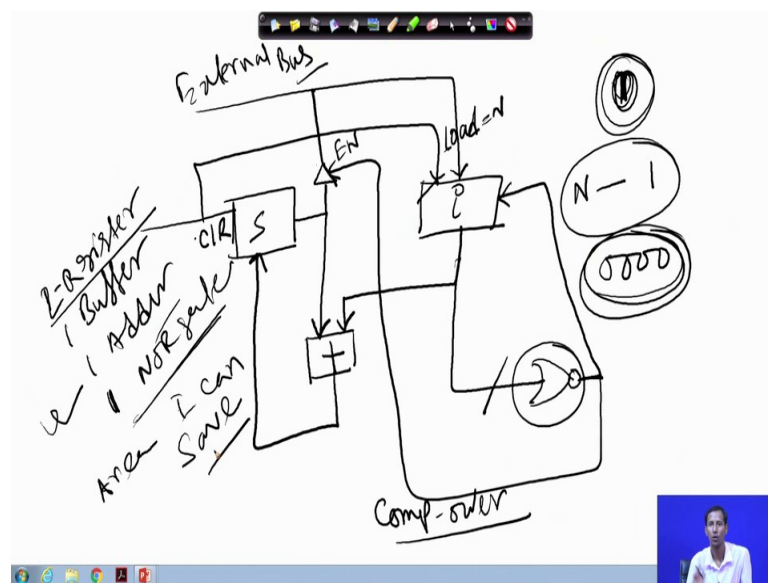
So, instead of doing that if I just change the logics something like this for $I \text{ equals to } N \text{ to } 1$ $S \text{ equals to } S \text{ plus } i$. So that means, this means what I that the here what I need, I need one incremented circuit which is or up counter circuit; that means, it is increased by 1 in

each clock cycle. So, here I need decrementer circuit. So the initial value of this; that means, I don't need any values or any storage for keeping the value N what is the value for N. So, here in the counter itself in the earlier logic what we are doing we are we are, that means from the external bus we are knowing that ok.

Users have selected the N value is something like this then each of the clock cycles we are just basically checking, whether that I value the up counter value that has been reached to the N. So, whenever it has reached to the N the computation will be over and the corresponding output will, will go to the external bus. So, instead of doing that here in the counter itself if I load the initial value of N. And then we start decrementing it whenever we will. That means here the logic will something like what 3 dot dot dot my logic, logic initially was something like this, but now my logic has become something like N minus 1 sorry N minus 1 dot dot dot dot plus up to 3 plus 2 plus 1. That means, a have just reversed the order ok.

The functionality is remaining same, but I have just reverse the order in order to or to; that means, reduce the corresponding architecture. So, if I just change this logic from i equals to N to 1 and S equals to S plus 1 S plus I so at that time what will be the architecture for this.

(Refer Slide Time: 23:44)



So, the architecture for this is now what I need 2 registers now ok, this is S and this is I so I is now 1 decrements circuit. So, what I need here this is the load values, so that

means so this will load the value of N and this is the clear signal initially S. That means, this is the storage sum storage. So, initially it will be set to 0 and then this will come to the adder circuit ok, and then again it will come back to this particular things storage. And here what I missed, this is the external bus what is remain in this particular circuit how can I set the corresponding, that means the terminating condition of the loop.

So, in the previous what was that whenever this i and N both are same so at that time the combination will be over? So, now, here what is the terminating condition? That means, whenever this corresponding decremented value that has, that means come to this particular 1 sorry, whenever it will reached to 0. That means, up to 0 I need to add. So, whenever it will be come to 0, so at that time the whole competition will be over.

So, that means, the 0 value will not the indicting or. That means, as I said that N to 1 it will just count; so that means, now here I need whenever how can I change this all the bits are 0 0 0 0. So, for that reason what I need one NOR gate ok. So NOR gate when all the input bit is all the input bits are like 0. So, at that time it will produce 1 that is the logic ok. So that means, now this will come to this and this will again come to that ok. That means stop the counter stop counting and this signal will come here to says that this is a completion over signal or you just here it will be like enable, enable this a buffer.

So, that the corresponding S value gone to the external bus. So that means, here is the in the that means, previous slides if I just see this circuit architecture so at that this particular things what I require area wise what I require, 3 register 1 adder 1 comparator right and 1 buffer to it. So, in this particular thing in this particular architecture what I require, I require 2 register if I consider for this S and I 1 buffer, 1 adder and instead of comparator what I require 1 NOR gate. So that means, now I can reduce 1 registers over here and instead. That means, in comparison to the comparator 1 NOR gate also will require lesser number of hardware ok.

So that means, now I can say that area wise I can save just doing. That means, a simple trick I can reduce the hardware. So, reducing the hardware, that means obviously, that will benefit in terms of power ok. So, this is a basic or this is a. That means, very simple things how we can do the optimization whenever we are implementing it from the algorithm to the architecture.

So, thank you for today's class. Again, we will see some interesting these circuit implementation techniques in the next few classes.

Thank you.