Architectural Design of Digital Integrated Circuits

Prof. Indranil Hatai
School of VLSI Technology
Indian Institute of Engineering Science and Technology, Shibpur, Howrah

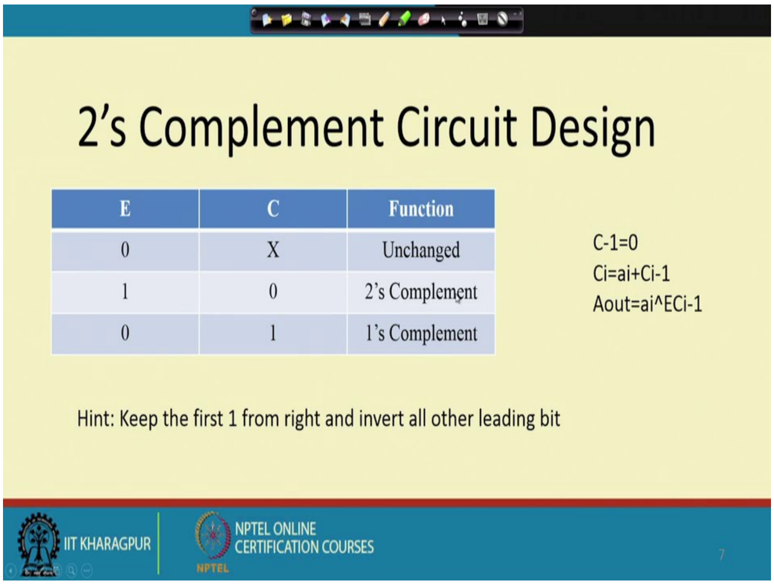
Lecture – 06
Algorithm to Efficient Architecture Mapping (Contd.)

Hello everyone welcome back to the course on Architectural Design of ICS. So, in the last class we have seen that 1’s complement circuit controlled one 1’s complement circuit. And we have we have discuss that 2’s complement we find out how we find out the 2’s complement circuit; that means, we find out the 1’s complement of circuit that we find out 1’s complement of the number first then we add 1 to that that becomes 2’s complement so; that means, now what from that particular point or from that particular perspective we can see that one circuit that can be act as 1’s complement as well as 2’s complement.

So, at that time how we can; that means, draw a circuit or how we can design a circuit, which can gives me both the things at a time; that means, that their circuit will remain same only I have to put some additional control logic to it. So, that I can get the in one particular circuit I can get 1’s complement as well as twos 2’s complement ok, so that we will see in today’s lecture.

So, if you follow this these 2’s complement circuit. So, enable and; that means, complement this is 2 control signal whenever this is 0 and cross at the time it will remain unchanged when enable equals to 1. So, at that time; that means, E signal is 1 at that time I will get 2’s complement.

(Refer Slide Time: 01:56)



2's Complement Circuit Design

E	C	Function
0	X	Unchanged
1	0	2's Complement
0	1	1's Complement

C-1=0
Ci=ai+Ci-1
Aout=ai^ECi-1

Hint: Keep the first 1 from right and invert all other leading bit

Or whenever this C signal is 1, so at that time I have to get it 1's complement; that means, now I am having 2 control signal which is E and C, so based on that if E is 1 if E is 1. So, at that time I will get 2's complement if C is 1, so at that time I will get 1's complement circuit.

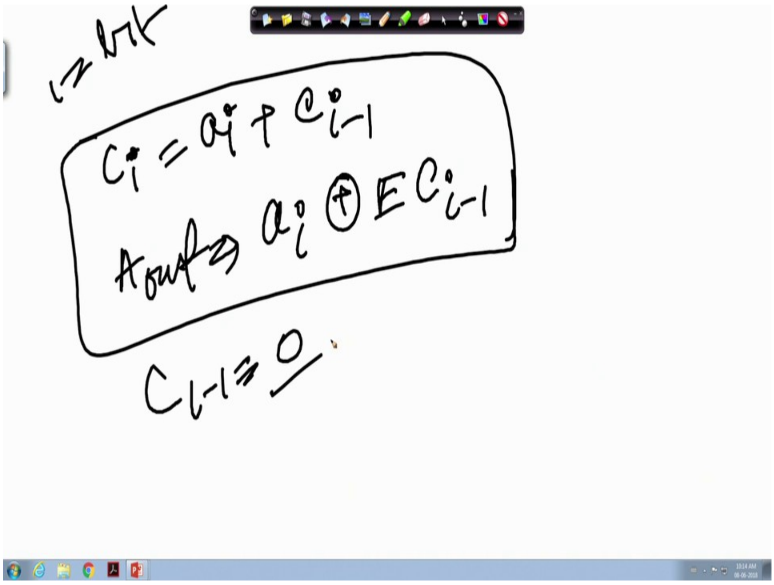
So, what is that; that means, hints here as I discussed that what ah; that means, I have to do this 1's complement and then if we add it by 1 then it will became this 2's complement circuit. So, in 1's complement circuit what we do we use XOR gate ok. So, in whenever and if you see this incrementer by 1. So, at that particular case also we will use the series of XOR gate ok; that means now can I conclude or can I; that means, make the expression in such a way.

So, that I can make it common; that means, I can re use those XOR gate for 1's complement as well as this 2's complement circuit. So, can I do that, so that we will see in this particular lecture ok. So, for that and what is the hints here that is keep the first one from right and then invert all other leading bit so; that means, the corresponding equation for that that will be whenever we adding 1; that means, I have to generate the carry right.

So, generate the carry means how we can generate the carry, so for carry i, I can calculate by using this particular equation which is Ai plus C i minus 1 which is the previous output previous carry output; that means, previous stage carry output that we will be added with that would be or with the corresponding positions of the input and that you will get the corresponding carry out ok, for that particular stage and for the A out; that means, which is nothing, but that 2's complement or this 1's complement circuit.

So, that we will get by this Ai x or with E of C i minus 1; that means, the previous carry that is added with the another signal and that is x or with the corresponding input bits ok. So, if I just want to draw this particular equation; that means, if I just want to draw 4 bit this 1's complement come 2's complement circuit. So, at that time how it look like that we will see that architecture we will see now ok.

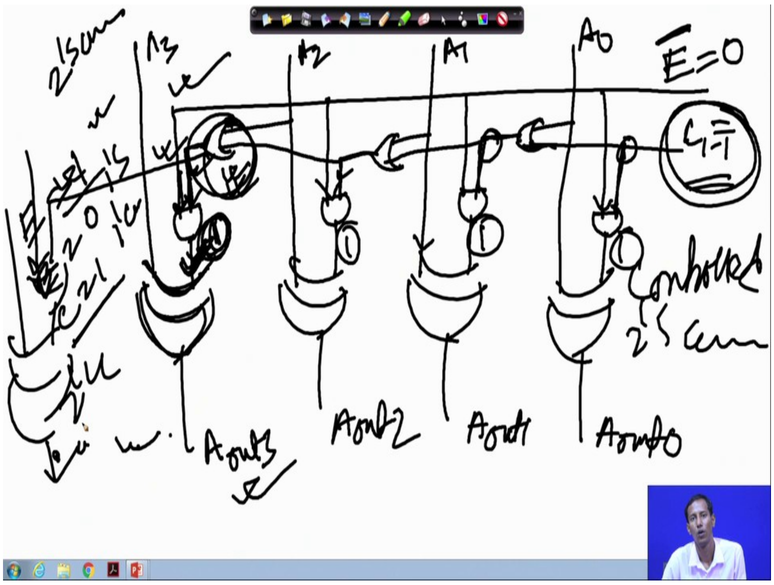
(Refer Slide Time: 05:19)



Now, so what was the equations, so if I just write the equation something like this. So, C Ci is A i plus Ci minus 1 and what is this A out that is A i this is A i XOR with E C i minus 1. So, this two equation basically is followed to calculate or to compute the corresponding architecture for that. So, here actually i means this is for i bit, so if I consider 4 bit.

So, at that time how the architecture will look like, so in the initial case and if I just go back in the initial case this Ci minus 1 that is equals to 0; that means, for the first stage as there is no carry. So, that is been carry in as considered as 0 ok, so now, for this first case.

(Refer Slide Time: 06:28)



So; that means, what I what I need I am having 4 x or gate for the from here I will get A out 3, this is A out 2, this is A out 1, this is A out 0 ok, now this is A 3, this is A 2, this is A 1, this is A 0, this is A 3, A 2 A 1 and A 0.

So, the next if you just go back A out is dependent on what A i XOR with E of C i minus 1 so; that means, to calculate this particular things I need 1 AND gate in each of this corresponding XOR gate input so; that means, here what I need, I need AND gate here I need 1 AND gate, here I need 1 AND gate and here I need 1 AND gate. Where each of the input of this and gate that is fix to E then what, what is the other input the other input is the carry out of the previous stage.

So, that carry out how we can; that means, calculate the carry out I can calculate by using A i XOR with C i minus 1; that means, for that what I require, I require OR gate as you see here I; that means, this is the OR operations sorry this is or operation A i OR with C i minus 1 so; that means, now this is this and actually if I start from here so; that means, this will be the another case here also I need 1 OR gate ok.

So, 1 of the input is this as there is a there is no; that means, previous of this input. So, that is why OR gate is not needed for this last stage ok, but the rest of the case it is required. So, then what will be the other input is this one for this one and for this what will be the case that is C i minus 1 ok. So, now, if you see; that means, I have drawn the I have calculate the carry out for the previous carry out I will get from this, and then I will end with E and I will get a this in this fashion.

So, now if I just want to verify this particular circuit that, whether that is what I; that means, what was my target if E equals to 1. So, at that time it will be act as a 2's complement if E equals to 0. So, at that time it will act as a 1's complement circuit 0 and C equals to 1. So, at that time it will act as a 1's complement circuit, so is it doing the same thing.

So, initially you check E equals to; that means, if E equals to 1. So, at that time each of this particular bit is 1, 1 means now this will depend on the corresponding carry of this so; that means, if increment the circuit at that time what will happen this is nothing, but the incremental circuit ok. So, this incremental circuit means what. So, at that time what will happen I am, I will just added by 1 here. So, whenever I am adding this; that means, E is 1, so E then the AND gate output will depend on what this particular input.

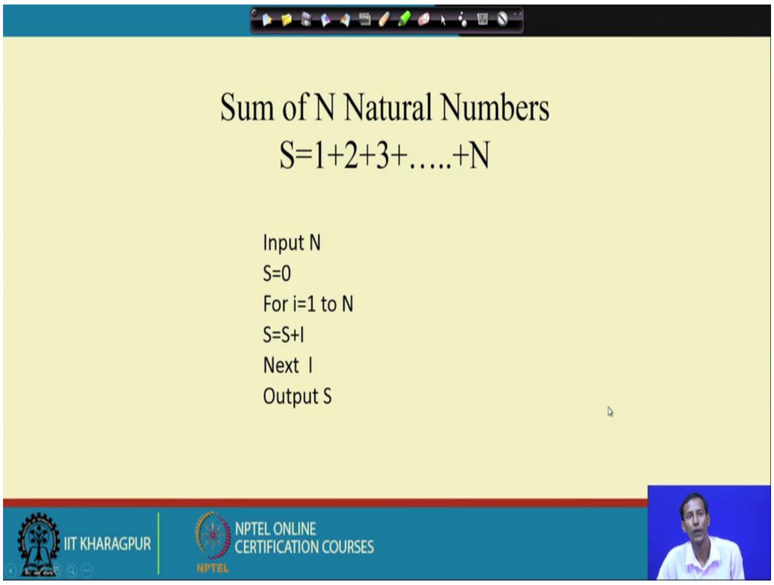
So, this will be the output of AND gate will be this one. So, as I am following or as I am doing this putting this carry in this particular chain towards the MSB. So, at that time this carry will be propagated and then that will be XOR with the corresponding bit position of A 3, A 2, A 1 and A 0 and E will get the A out; that means, the corresponding 2's complement circuit ok. So, this is the controlled 2's complement circuit. So, now if I just want to; that means, make this as if I want to make it; that means, work that as in 1's complement circuit. So, at that time what I need, I need that this particular position should be 1.

So, this particular position to be 1, so then what I need to do, for that what I need to do to calculate that I have to do; that means, if this is 0 then this will be what 0 so, but I need for this inversion I need this as 1 so; that means, now at that time I have to make it the bar of that so; that means, if I put it as this E bar at that time, so then it will be one and there I need to change the corresponding this value to. So, that it does not effect the corresponding.

If this bit is 1; that means, at that time it will be dependent on that and then I can get the corresponding value for this ok. So, this is for controlled basically 1's complement circuit whenever, I have to implement at that time I need to change something. So, that it can work as a ones complement, but this is controlled 2's complement circuit by which we can get; that means, this is for a 4 bit for N bit this will just continue that mean this to particular, this particular position will continue and then that at the final stage also we will get something these values here AND gate and that will be or with the carry end will be generated from the OR gate.

And then that will be ANDed with you enable signal, and you can get the corresponding N bit some output sorry this 2's complement value out ok. So, this is for controlled 2's complement circuit implementation ok. So, then next we will see another example of let us go back to the slide ok. So, then next what we will do, we have to find out sum of N natural numbers suppose this is my job to do.

(Refer Slide Time: 15:21)



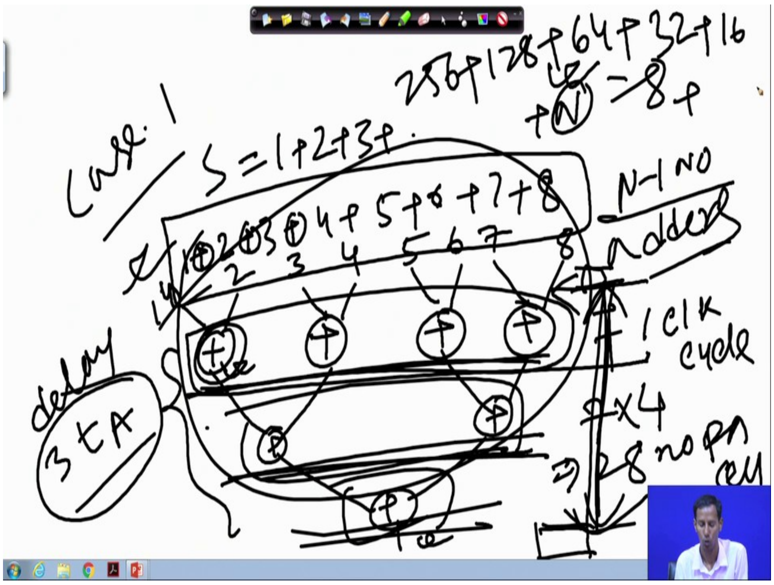
Sum of N Natural Numbers

$S=1+2+3+.....+N$

```
Input N
S=0
For i=1 to N
S=S+I
Next I
Output S
```

I have to find out sum of N natural numbers so; that means, sum equals to 1 plus 2 plus 3 plus up to dot dot dot up to N. So, N I can specify that N value to 100 N value 50 200 1000 any value I can say. So, how I can implement this particular circuit?

(Refer Slide Time: 16:02)



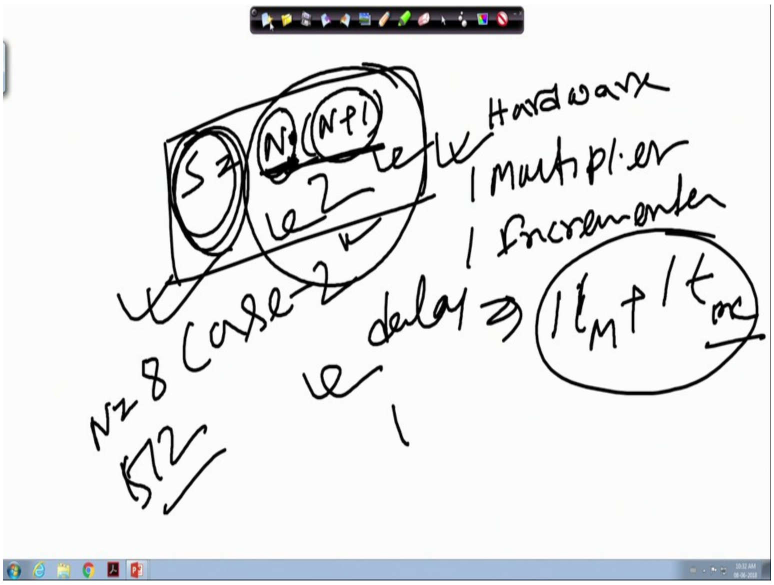
So first you consider that means S equals to 1 plus 2 plus 3 plus dot dot dot dot up to N, I have to. So, if I just; that means, this is means what? 1 that is added with 2, then that is added with 3, then that is added with 4, then that is added with 5, if I consider let us say N equals to 8; 5 6 then 7 then 8 ok.

So; that means, this 8 numbers I have to add, so what we can do we can use the adder something like this where I can put 1 2, 3 4, 5 6, 7 8. Then this again I have to add then this to this so; that means, here you see how many number of adders I require? Total 7. So, this is; that means, the case or this is for any N bit number of addition in this particular fashion I need N minus 1 number of adder.

So, then there is another implementation style for this and what is that? That means, the corresponding delay for that; that means, as there are 3 levels of adders. So, the delay becomes 3 t adder delay as there are 3 level of adders. So; that means, for this particular implementation I need N minus 1 number of adders and the delay with 3 adder delay ok. So, if each of this bit if I consider that as 4 bit or 8 bit.

So, at that time I require for if this each of this bit is like if I consider that as 4 bit or 8 bit 4 bit, if I consider then at that time what I, how many full adder cell I require? 7 into 4 that is 28 number of full adder cell I require. So, is there any other; that means, different that methodology or different techniques to implement the same thing? Yes there are or there is ok, so this expression, this expression.

(Refer Slide Time: 19:06)



If I can write in an another form where that is S equals to N into N plus 1 by 2 this is the expression of summation of N natural number. So, if I just want to add this oh sorry, if I just want to implement this particular circuit.

So, at that time what I require? I require 1 incremental circuit here, and then I require 1 multiplier over here and divide by 2 means I will just discard the LSB that is hard shift (Refer Time: 19:40) if I do if I discard the LSB that will be divided by 2. So, for divide by 2 I do not need any extra hardware, but here the hardware wise or area wise what I require is 1 multiplier and 1 incrementer circuit. And what is the delay is 1 multiplier plus 1 incrementer.

So; that means, this is another; that means, implementation style by which I can make the corresponding, I can or I can implement the corresponding expression for N summation of N natural numbers. So, in these two particular case in this case if this considered as case 2, and the previous this is case 1 ok. So, what is this two case? Here, what I am doing? I am calculating the values simultaneously or parallelly ok

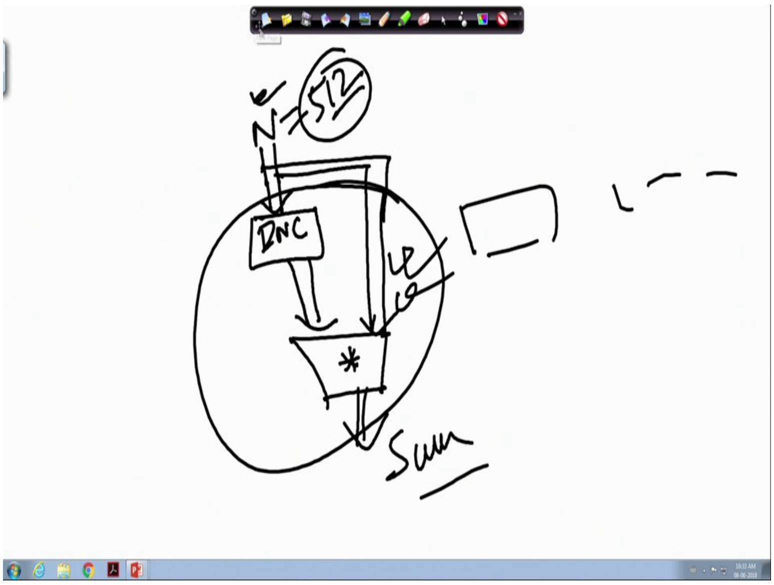
So; that means, I am putting all these values what I am doing? I am putting all these values parallelly to this particular circuit. And then I am getting the corresponding output, if I consider this as this; that means, with register we put at this particular level and register we put at this output level. So, this is the critical path which is nothing, but this 3 delay 3 t; that means, adder delay and this is; that means, from this particular path to this that I can compute within 1 clock cycle ok.

So, the next thing also the same thing here also I can implement this particular function in 1 particular case. Then actually where I need to if you just see 1 plus 2 plus 3 up to dot dot N using this fashion I can do. And here also this 1 star; that means, N dot N plus 1 by 2 also I can implement, where I will use this particular circuit? Where I will use this particular circuit? Though actually the delay for this particular circuit is more than the previous case; but this style is also I have I can use somewhere.

Where I can use? When the N value is very large ok; that means, if the N value in previous case I have set to 8 if the N value is let us say; that means, 512. So, at that time designing this particular architecture will became too much of complex; that means, at this particular stage I need 256 adder then add this to add this particular case; I need 128 adder, then I need 64 adder, then I need 32 adder, then I need 16 adder something like this I require.

So, many levels and I require this kind of structure tree structure of the adders. So, describing using HDL or any other language that will became too much complicated whenever I am following this particular method. So, at that time instead of that when the N value is more at that time what will happen? If I specify this, only here what I need? I need if I just draw this.

(Refer Slide Time: 24:06)



What I require? I need 1 I need to put this as N this is incrementer circuit then that will come over here. So, this is incrementer circuit which is multiplied with this, so here I will get the corresponding sum.

So, whenever this N value is 512 so I do not need to; that means, change here only the thing is that I need to load the value of N only once, I do not need to for 512 or 1 0 2 4 or 2 0 4 8 I do not need to describe any other; that means, adder oh which I have to do in the previous case. So, for those type of application or for those type of things where circuit implementation, I will follow this kind of architecture or this kind of circuit implementation.

So, then is there any other technique or other method by which I can implement the same function yes there is ok. So, what is that that is the iterative method so; that means, iterative method means using only 1 adder I will run that particular things; that means, this particular; that means, I will in each of the stage I will add I will store it somewhere and then again I will add to the next that storing value that will again I will add with the next value.

So, something like this then whatever N value I have loaded up to those values unless and until it is reaching to those particular values, it will try to in each of the clock cycle it will try to add and then accumulate add and then accumulate add and then accumulate. So, whenever we will get the final result, so at that time it will freeze and it will go to the corresponding output.

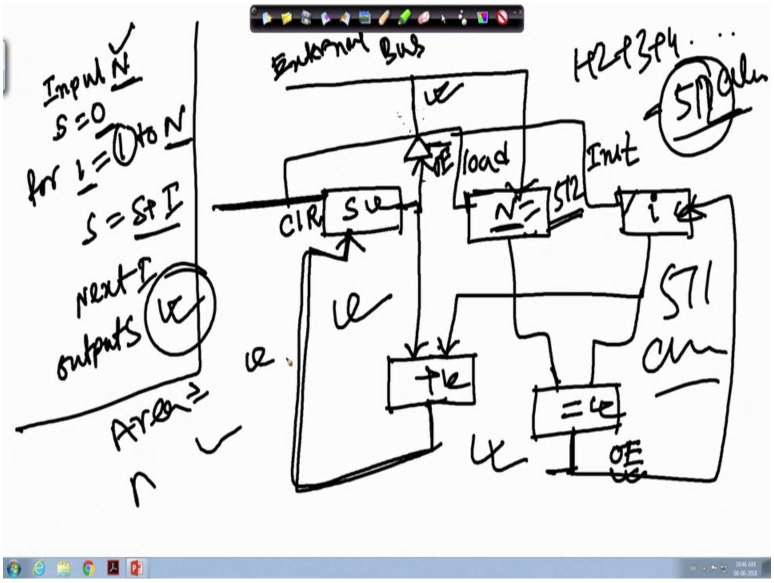
So, to ah; that means, compute this iteratively the algorithm is something like this the input is N so; that means, the here you see this input is basically N initial; that means, the sum is 0 and then for i equals to 1 to N S will be sum will be; that means, modified with previous sum plus whatever value is there that is the i, and whenever this job is finished then change the corresponding i value to the next and when all this is finished then output you just assign to the; that means, the S value that will be assigned to the corresponding output, so from this particular algorithm.

So, if I just want to map this particular algorithm into the architecture at that time what do I require. if I just follow or if I just closely; that means, see I am having 3 variables what are those I am having I needing this N over here then I need S and then I need I so;

that means, I need 3 variables for computing this particular ah; that means, this logic or this particular algorithm.

If I want to implement then I need 3 variables and then for this particular I need 1 adder and rest of the things we will just whenever we will draw the corresponding architecture at that time we will see that ok. So, if I just want to map it to the architecture for this particular algorithm.

(Refer Slide Time: 28:41)



So, what is that that is input is N initially S is 0 then for i equals to 1 to N S equals to S plus I then next I and put the output as S. So, this is my algorithm for computing the sum of N natural numbers so; that means, what I said that I have I am having 3 input variables sorry not input variables I need 3 registers or 3; that means, for N as there are 3 variable N S and i. So, I need 3 register N for N S and i.

So, initially what I will do, I will put that like this, so this is S, this is N and this is i. what i is basically is a up counter which is basically counting from 1 to N. So, what I require initially I need the control signal for each of these registers. So, initially S is what S is 0, so this will be clear initially this registers will be clear. for N what I require it will be load; that means, initially I need to load the values of N to this particular register.

And this for i whenever this is started this is this has to start with 1 so; that means, then this will be what this will be initialize, initialize the counter with values of 1 ok. So, then next what I have to do, I have to do this S plus i so; that means, I need 1 adder circuit. And then output of that that is again connected with S then S; that means S is basically connected to this and this value is again connected to this; that means, in 1 cycle the previous S is coming here that is added with i. And then again it is going back to; that means, the corresponding output that has go back to the sum registers, so from this is the external bus.

So, this will load from the external bus only I can load the values of N and again I need 2; that means, deliver whenever this job is finished. So, at that time I need to deliver the output to the external. So, what I require is that I require sorry, I require 1 buffer over here ok. So, let me just to want to I need 1 buffer over here ok.

When this total job is finished, so at that time this buffer will produce the corresponding this S value will go out via through this external bus. So, what will be the condition for this will be enable. So, that this S value can go out to the external bus and what is this is a loop running right. So, what is the terminating condition, so terminating condition is that; that means, it has to reach to the N value. So; that means, I need 1 equals to comparator whenever this i equals to N. So, at that time what will happen it will try to stop ok.

So; that means, whenever this value is equals to N, so at that time this will come and this will again start from 1, and this particular signal is the terminating condition. So, this if I just say that is; that means, output enabled. So, this signal will again be connected with this particular signal; that means, whenever this signal is 1. So, this will be enabled and S will be at that time it will go to the external bus.

So, this is the corresponding algorithm was that for finding out the N natural numbers the corresponding iterative circuit for calculating N natural numbers is that. So, now, if I; that means, have to if I the put that N equals to 512. So, in each cycle it will this circuit will basically run and it will try to compute 1 plus 2 then 3 then 4 something like this; that means, as this count value is basically increasing initially it is 1 then next cycle it is 2 then 3 then 4 and where I am doing the addition.

So, initially it was 0 added with 1, so 1 is stored here then in the next it is 2. So, 1 is coming from here, so 1 plus 2 then this is 3 so 1 plus 2 is already stored here. So, it comes with 1 plus 2 plus 3 then next it is 4, so air 1 plus 2 plus 3 that is stored here, so 1 plus 2 plus 3 comes here in the next cycle so 1 plus 2 plus 3 plus 4 that again comes with this.

So, in this fashion or in this manner this basically runs and how many times or how many times it will run; that means, up to this 511 clock cycle it will run ok. So, that; that means, sorry this 512 clock cycle it will this particular circuit will run and in the 512 clock cycles, I will get the output at the external says; that means, external bus if I select this N input as 512. So, then if you if you see here area wise; that means, what is the; that means, benefit of the circuit area wise what I require only I require 3 registers and 1 adder and 1 comparator.

So, 5 for 512 I need 511 for case 1 and for case 2 that is fixed 1 multiplier and 1 incremental, but here what I require only 1 adder and 1 comparator and registers are like they are also you require registers. So, if I say that here I require 3 register and delay wise what is the maximum delay. So, delay wise is that this registers delay or this and this adder delay that is the maximum delay which I can get from this particular circuit, so delay wise also it is very much faster.

Then this case 1 and case 2, but where actually whenever we have we are gaining something at that time we have to lose somewhere. So, what is the loose point I am losing here the time; that means, for total competition time of the corresponding circuit. So, there within 1 clock cycle I can finish the job, but here for 512 say number of samples if I want to add then up to 511 clock cycle, I have to wait. after 511 clock cycle then only I will get the corresponding output ok.

So, in next class or in the; that means, next few classes we will see. So, this is the; that means, corresponding algorithmic architectural implementation of this particular algorithm can I optimize this particular circuit can I do that, if I can do that then how or what is the approach where I will look into. So, that I can find out this is the point where I can look into and I can optimize this circuit, so that we will see in the next class.

Thank you for today’s class.