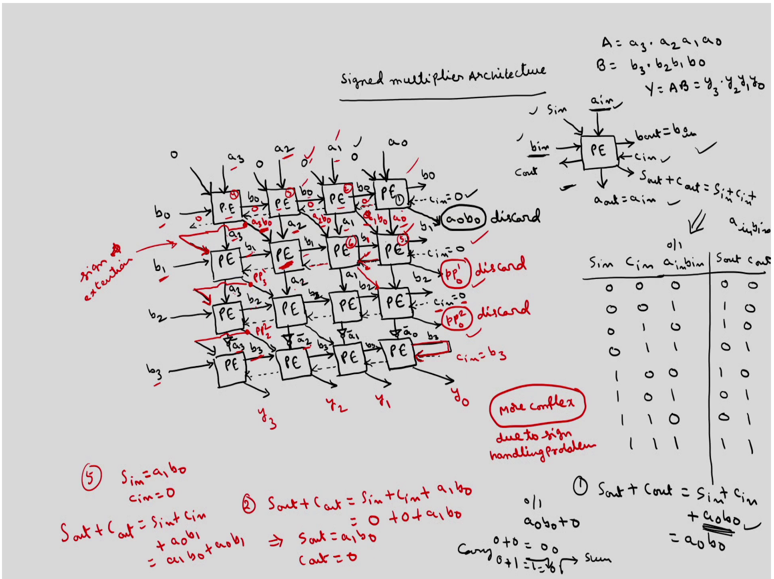
System Design Through VERILOG
Prof. Shaik Rafi Ahmed
Department of Electrical and Electronics Engineering
Indian Institute of Technology, Guwahati

Case Studies
Lecture - 26
Baugh-Wooley signed multiplier architecture

In the last lecture we have discussed about the algorithm for signed multiplication. So, today we will map the algorithm onto the architecture.

(Refer Slide Time: 00:49)



So, what is the architecture for signed multiplier? In order to map this algorithm onto architecture first we have to define a processing element. So, here I am going to define processing element as this. So, with a in is the input a out is the output and I will make a out is equal to a in.

Similarly b in is the input b out is the output I will make b out is equal to b in, then the carry input we are going to apply here c in c out and here we are going to use some input s in and this is s out. So, this s out plus c out we are going to write combinedly s out plus c out is equal to s in plus c in plus a in b in.

So, basically this is a full adder. So, this a in b in is a single value 0 or 1. So, the meaning of this one is s in is one input c in is another input s in is applied here c in is applied here and a in b in we can obtain by taking the AND gate between this a in b in by passing through the AND gate will get a in. So, by passing a in and b in through the AND gate we will get a in b in.

So, this is a single value a in b in a single value this can be 0 or 1 this is the output of the AND gate. If we add 3 bits you will get outputs as 2 outputs which is s out c out that is the meaning of this. So, 0 0 0 sum is 0 carry is 0, 0 0 1 sum is one carry is 0 exactly same as the full adder this processing element here is a basically a full adder.

So, 0 1 0; 1 0; 0 1 1; 0 1 because the carry will be generated 1 0; 1 0 0; 1 0; 1 0 1; 0 1; 1 1 0 also 0 1; 1 1 1 is 1 1 this is the meaning of this definition. Now, using this processing elements we have to construct the parallel multiplier basically we have taken the multiplication of A as a 3 dot a 2 a 1 a 0 B as b 3 dot b 2 b 1 b 0 and Y we are taking as AB this also I am storing only 3 bits y 3 dot y 2 y 1 y 0.

Here I have given in general as a in a in can be either a 0 a 1 a 2 a 3 similarly b in can be b 0 b 1 b 2 b 3. So, because we have 4 a's and 4 b's. So, totally we need 16 such processing elements. If I take the overall architecture of this one, this is one processing element; this is another processing element, another processing element. So, we are going to repeat the same processing element throughout the system. So, this is called modular architecture. So, this particular array is called systolic array.

We have total 16 such processing elements, I will explain the operation with the help of the algorithm that we have developed in the last lecture, this is our 16 processing elements. So, basically a's we are going to give in this direction, b's we are going to give in this direction. So, here this is a 0 we will give here.

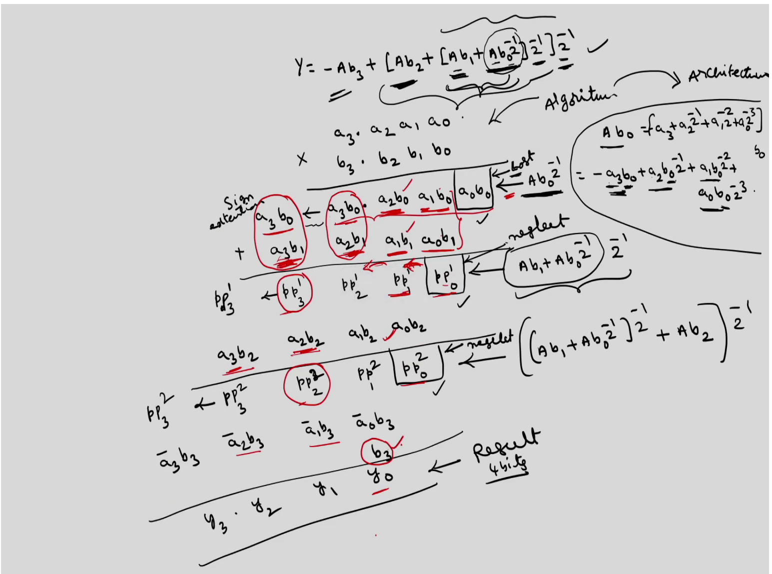
So, this line is also a 0 because a out is equal to a in this is also a 0 this is also a 0, but here a 0 we have to apply through the NOT gate, I will tell you what is this a 0 bar in fact, a 0 bar. So, I will tell the reason why this should be a 0 bar. Similarly, if I apply a1 here this is also a1 this is also a1 this is also a1, but for the last one I will apply a1 bar this is a1 bar this is a 2 output is also a 2.

Because I am defining this processing element such that output of a is equal to input of a a 2, but here you have to give a 2 bar. Similarly, here a 3 bar similarly here b0. So, this entire line is b 0 this is also b 0 this is also b 0 if I take output here also this will be b0. This center line is b 1 line b 1 b 1 b 1 and b 1 similarly b 2 b 2 b 2 b 2.

So, totally we need 16 product terms, that is why we require 16 processing elements, this is b 3, this is b 3, this is b 3, b 3, this is also b 3. So, this is about a's and b's connection ok and what about this sum and carry connections is this is c in here I will make this as 0 and this is c out this is c out this c out plus this s out is given by this relation.

So, the initial carries for all these things except the last stage I will give as 0's. Now this c in I will connect to b 3 I will explain why this has to be connected to b 3. Now some bits here I am going to give 0 here I am going to give 0 sum in this is also 0, this is also 0. Then the output of each one we are going to connect here, here this will give some output, this will give some output, here also we will be having sum in.

(Refer Slide Time: 11:44)



Now, to get this the final connections of this one. So, we will refer back to the algorithm that we have discussed in the last lecture. So, this was the algorithm that we have developed using

Horner's law. So, to perform the signed multiplication of two 4 bit numbers a 3 a 2 a 1 a 0 b 3 b 2 b 1 b 0.

So, the procedure is here first you have to compute Ab 0 and then Ab 0 into 2 raised to the power of minus 1 in that we have to lose this we have to neglect this and we have to extend the sign of these bits. Similarly, step by step procedure first we will find out this Ab 0 into 2 raised to the power of minus 1 then we add this again 2 raised to the power of minus 1 we add Ab 2 again 2 raised to the power of minus 1 finally, we add minus Ab 3.

So, a 0 b 0 basically here this should be a 0 b 0, you can see that here what is the output here? If I call this as the processing element for this definition of the processing element what is the output of this first processing element s out plus c out is equal to s in plus c in plus a 0 b 0 for this particular processing element. If I call this as processing element 1 for processing element 1 what is s out plus c out is equal to s in plus c in plus a 0 b 0 because the inputs are a 0 b 0 and what is s in 0 what is c in 0. So, this is simply a 0 b 0.

So, what does it mean? So, only sum bit is a 0 b 0 there is no carry at all because only this a 0 b 0 is single bit this can be 0 or 1. So, sum bit is a 0 b 0 this is sum bit. So, this we are going to discard because if I want to store 8 bit of the result here I am storing only 4 bits of the result. If I want 8 bits of the result I can store these values also.

So, here, what I am discarding this is sum we are going to discard and what is this carry this carry is 0 because this is only single value which is either 0 or 1 which is sum itself. So, this is something like a 0 b 0 plus 0 if I perform this a 0 b 0 can be 0 or 1. So, what is sum bit and carry bit 0 plus 0 is 0 only sum bit is 0, carry bit is also 0, if you want to write 2 bit equivalent 0 plus 1 is in decimal this is 1 what is the binary equivalent 0 1 means this is sum bit and this is carry bit. So, this is 0.

Now, what about this value the output of this? So, I will write now these values. So, for this processing element what are the inputs this carry is 0, this sum is 0. So, output is simply the inputs this is b 0 is the input and a1 is the input. So, output of this one is a 1 b 0 of course, this carry is 0.

And what about the carry here also same thing for this processing element if I call as 2 for processing element 2 also what will be the input s out plus c out is equal to s in plus c in plus what are the a's and b's for this second processing element a is a 1 b is b 0 a 1 b 0. But this s in is 0 plus this c in is 0 plus a 1 b 0.

So, always only this bit is sum bit only carry bit is 0. That means, implies s out is simply a 1 b 0 and c out is 0. So, this c out is also 0 this s out is a 1 b 0. Similarly you can find out output of this processing element as the sum bit will be 0 carry bit will be the input is a 2 b 0 this is a 2 b 0.

Similarly, this will be a 3 b 0 and this carry output is 0. So, you have generated a 1 b 0 a 2 b 0 a 3 b 0 a 0 b 0 in that a 0 b 0 I have discarded. So, whatever this the first I mean this row a 0 b 0 a 1 b 0 a 2 b 0 a 3 b 0 I have generated using the first 4 processing elements in that a 0 b 0 is discarded.

Now, in the second row what is to be added to this one? So, we have to add this I have discarded for a 1 b 0 I have to add a 0 b 1 for a 2 b 0 a 1 b 1 and so on. And this a 3 b 0 has to be extended to sign ok this a 3 b 0 has to be added with a 2 b 1 as well as a 3 b 1 ok. Now, I will show that one now this thing is a 1 b 0 is generated a 0 b 1 you have to add. So, coming for this processing element.

So, what are the inputs a 0 is this input this b 1 ok. So, sum in for this if I call this one as 1 2 3 4 the 5th processing element for 5th processing element you can see a sum in is equal to s in we can call as sum in is s in, s in is equal to a 1 b 0 a 1 b 0 and c in is we have applied as 0. So, therefore, s out plus c out is equal to s in plus c in plus what is a and b that is applied for this 4th one a is a 0 b is b 1. So, plus a 0 b 1.

So, this will be c in is 0. So, this is a 1 b 0 plus a 0 b1 ok this will get some sum term as well as carry term ok, what is sum bit and what is carry bit of this? So, if I add these two sum bit is pp 0 1 and carry bit we have to propagate and we have to add here. So, sum bit of this stage is pp 0 1 that also we are going to discard.

So, this sum bit of this processing element 5 is pp 0 1 this also we are going to discard and carry bit we are going to this carry bit is not 0. So, this is carry of this bit is carry of a 1 b 0 plus b 1 a 0 that we are going to add in the next stage. So, this carry of this one we are going to add in the next stage ok, this carry whatever is generated sum is discarded.

So, now in the third stage we have to add a 2 b 0, a 1 b 1 and then the carry of the previous one sum of that one is pp 1 1 and carry again we have to propagate to next stage. You see here if I take the processing element 6. So, what is the output sum bit? So, of course, here we have these two connect to sum here.

So, what is the sum bit of this p 6? So, this is carry of this one previous bit and these bits are the sum bit is a 2 b 0 and this is a 1 b 1 a 2 b 0 plus a 1 b 1 a 2 b 0 plus a 1 b 1 plus the carry this is going to perform in the processing element 6, a 2 b 0 a 1 b 1 plus the carry from the previous stage this one this carry from the previous stage this is not 0.

So, what is this sum bit of that one? Sum bit of this one will be this pp 1 1 carry bit you have to propagate to the next stage. Again this sum bit has to be added with a 0 b 2 a 0 b 2 this is a 0 b 2, whose sum bit is pp 0 2 this have to neglect or discard pp 0 2. So, pp 0 2 will be generated here this also we are going to discard. So, like that we will proceed. So, at the last stage here we see this a 3 b 0 has to added with a 2 b 1 a 3 b 0 has to added with a 3 b 1 also a 3 b 0 has to be added with a 3 b 1 a 2 b1.

So, a 3 b 0 a 3 b 0 has to be added with a 2 b 1 a 3 b 1 a 2 b 1 a 3 b 1 a 2 b 1 is this one, a 2 b 1 is this one we have to add here. Then we are adding this a 3 b 0 we are adding with in this processing element a 2 and then correspondingly b 1 the output of this processing element is a 3 b 0.

So, in this processing element what addition will be performed a 3 b 0 plus a 2 b 1 of course, the carry from the previous stage also then other one is a 3 b 0 we have to also add with a 3 b 1 right a 3 b 1. So, this sign extension this a 3 b 1 also you have to add. So, what we have to do is same a 3 b 1 you take from here and you connect to the this one, this will show the sign extension of sign extension connection.

Now, the same a 3 b 0 is going to add with a 2 b 1 as well as a 3 b 1 this is a 3 b 1 this is a 2 b 1. So, that is what we have here a 3 b 1 a 2 b 1. Similarly this pp 3 1 has to be add with a 2 b 2 a 3 b 2. So, that we have to do in a similar manner. So, this is pp this one. So, this has to be added with both a 2 b 2 as well as a 3 b 2. So, this is the connection sign extension, similarly here also this has to be added with this.

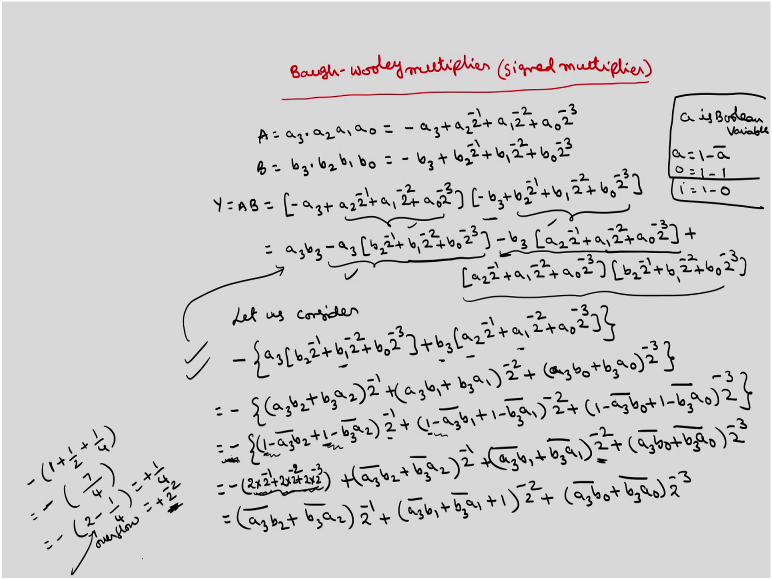
So, this connection will be this value will be pp 3 1 and this will be pp 2 2 this is pp 3 1 this is pp 2 2 you can easily check this. So, the pp 3 1 has to be added with two values that is a 2 b 2 as well as a 3 b 2 a 2 b 2 a 3 b 2 similarly pp 2 2 is a 2 bar b 3 a 1 bar b 3 and this is a 2 bar b 3 and a 3 bar b 3.

Now, there is one extra term like this b 3 has to be added along with this pp 11 a 0 b 0 and we have to add b 3 also whose output is y 0. So, this addition of the b 3 nothing, but you can see that this is b 3. So, instead of making this c in is equal to 0 if I connect this b 3 to c in, so c in becomes b 3. So, that the third bit that will be added will be b 3. So, this one will be implemented by connecting this b 3 bit to the carry in carry in is equal to b 3.

So, here you will get y 0 y 1 y 2 y 3. So, you see about the signed multiplier architecture we just have mapped this algorithm onto architecture this algorithm whatever we have developed in the last lecture. So, we just mapped this on to the architecture, these are 4 bits if you want to retain these bits also then you can increase the accuracy. Otherwise you can neglect also. So, this is about the complete signed multiplier architecture, but the drawback of this one is this is more complex due to sign handling problem.

So, we have to take the sign bit we have to extend the sign of this one. So, because of that this is more complex is to avoid this sign handling problem. So, there is one efficient signed multiplier called Baugh Wooley multiplier. So, next we will discuss about the Baugh Wooley multiplier and using Baugh Wooley multiplier we can implement the filters.

(Refer Slide Time: 27:28)



This is also signed multiplier, but this will take less complexity how the complexity of this multiplier is reduced. So, we will see now using some mathematical background. So, we will take the same A is 4 bit number a 3 dot a 2 a 1 a 0 in binary form and in decimal form this is minus a 3 plus a 2 raised to the power of minus 1 plus a 1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3 and B is b 3 dot b 2 b 1 b 0 in decimal form minus b 3 plus b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3.

So, we want Y is equal to AB if I take the decimal values this will be minus a 3 plus a 2 2 raised to the power of minus 1 plus a 1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3 into minus b 3 plus b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3.

Here we will do some manipulation thereby we cleverly handle the sign involvement this is equal to be minus a 3 minus b 3 becomes a 3 b 3. So, there is no sign here. So, the only times where the sign is involved is minus a 3, if I multiply with this one and minus b3 if I multiply with this one these two are having the negative signs.

Minus a 3 with b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3 minus b 3 a 2 2 raised to the power of minus 1 plus a 1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3.

So, remaining thing is all positive times. So, what is that plus remaining thing that is left is this into this a 2 2 raised to the power of minus 1 plus a1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3 into b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3.

So, the only problem with these two terms which are having minus sign. So, I will consider only this minus sign terms first I will do some manipulation later I will come to this final expression. So, let us consider this minus I am taking outside a 3 this one b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3.

This becomes plus because I have taken minus outside b 3 a 2 2 raised to the power of minus 1 plus a 1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3 here I am going to use some trick. So, this is equal to minus sign first I will multiply as it is what are the coefficients of 2 raised to the power of minus 1 is a 3 b 2 plus b 3 a 2, this whole thing is 2 raised to the power of minus 1 2 raised to the power of minus 2 is this term and this term plus a 3 b 1 plus b 3 a 1 this whole thing is 2 raised to the power of minus 2 and 2 raised to the power of minus 3 term will be a 3 b 0 plus b 3 a 0.

Now, here I am going to write this as if a is a Boolean variable in general a is a Boolean variable I can write a as 1 minus a bar always because if a is 0 a bar is 1. So, this 1 minus 1 becomes 0 if a is 1, if a is 1, a bar is 0 this is equal to 1 minus 0. So, always you can write a as 1 minus a bar this is what we are going to write here. This is minus of this I will write down as 1 minus a 3 b 2 whole bar because this I will take as single term.

Similarly 1 minus b 3 a 2 bar into 2 raised to the power of minus 1 plus 1 minus a 3 b 1 bar plus 1 minus b 3 a 1 bar into 2 raised to the power of minus 2 plus 1 minus a 3 b 0 bar plus 1 minus b 3 a 0 bar whole thing into 2 raised to the power of minus 3. Now, what is the constant term? So, remaining thing is we will have minus of this I will take minus outside. So, if I take the constant term this is minus of.

So, the first term is 2 raised to the power of minus 1 1 into 2 raised to the power of minus 1 and this one is 2 raised to the power of minus 2 plus 2 raised to the power of minus 2 plus 2 raised to the power of minus 3. And this minus sign if I take inside this becomes plus this minus of minus becomes plus this minus of this minus becomes plus this minus of this minus becomes plus all becomes plus.

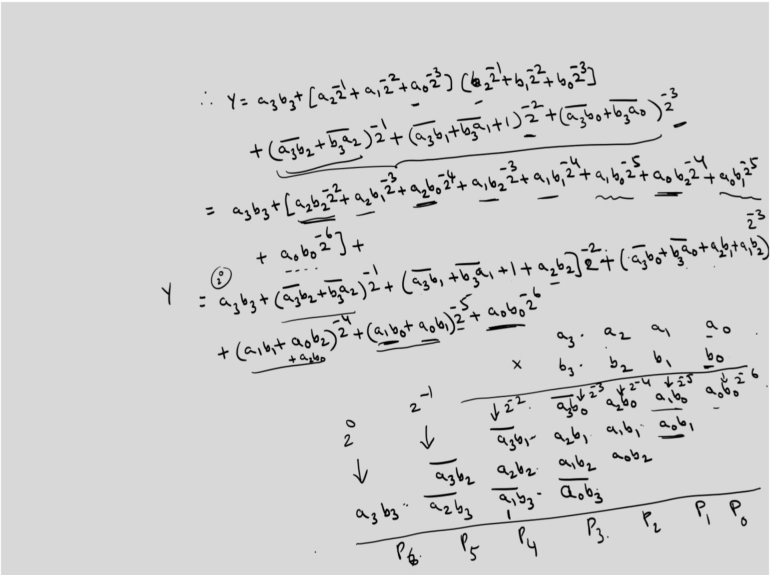
So, we will get a 3 b 2 bar plus b 3 a 2 bar into 2 raised to the power of minus 1 plus a 3 b 1 bar plus b 3 a 1 bar whole bar into a 2 raised to the power of minus 2 plus a 3 b 0 bar plus b 3 a 0 bar into 2 raised to the power of minus 3, sorry there is a mistake here. In fact, this is a 2 2 into 2 raised to the power of minus 1 this is not singly 2 this is 1 plus 1 is total 2 2 into 2 raised to the power of minus 1 this term is in fact, this is minus of 1 plus 1 is 2, 2 into 2 raised to the power of minus 1 here also 1 plus 2 plus 2 into 2 raised to the power of minus 2 plus 2 into 2 raised to the power of minus 3.

So, what happens now? So, this will be 2 into 2 raised to the power of minus 1 minus of becomes 1 plus this becomes 1, this becomes 2 raised to the power of minus 1 which is 1 by 2 this is 2 raised to the power of minus 2 which is 1 by 4. So, this is equal to minus of 4 plus 2 plus 1 is 7 by 4 this can be represented as 2 minus of 1 by 4. So, this 2 is overflow and we have seen that the overflow in twos complement can be neglected in between. So, this will it becomes simply plus 1 by 4 which is equal to plus 2 raised to the power of minus 2.

This is the logic that we are going to use to reduce the complexity of the system. Because one of the properties of this 2’s complement number is intermediate overflows can be neglected that is what I have explained in the last lecture with example also. So, this entire term becomes only 2 raised to the power of minus 2. So, what happens is here.

So, this is equal to a 3 b 2 bar plus b 3 a 2 bar into 2 raised to the power of minus 1 plus 2 raised to the power of minus 2 here all this term is also there. So, we will get 1 plus extra term this is a 3 b 1 bar plus b 3 a 1 bar plus 1 because this entire thing is equivalent to 1 by 4 which is equal to plus 2 raised to the power of minus 2 that if you take common this becomes 1 plus a 3 b 0 bar plus b 3 a 0 bar into 2 raised to the power of minus 3.

(Refer Slide Time: 38:23)



So, this is this entire value if we substitute this value here. So, we will get Y expression as a 3 b 3 is signed term plus we have two more terms. So, this term as it is a 2 plus a 2 2 raised to the power of minus 1 plus a 1 2 raised to the power of minus 2 plus a 0 2 raised to the power of minus 3 into b 2 2 raised to the power of minus 1 plus b 1 2 raised to the power of minus 2 plus b 0 2 raised to the power of minus 3 plus the remaining thing we have simplified to this value.

So, what is that a 3 b 2 bar b 3 a 2 bar a 3 b 2 bar plus b 3 a 2 bar into 2 raised to the power of minus 1 plus a 3 b 1 plus b 3 a 1 plus 1 whole bar into 2 raised to the power of minus 2 plus a 3 b 0 b 3 a 0. If you further simplify this and finally, we will get the algorithm from this expression we are going to develop a algorithm then that algorithm we are going to map onto the architecture. So, what are this a 3 b 3 is as it is plus what are the coefficients of 2 raised to the power of minus 1 minus 2 minus 3 and so on.

So, this is equal to here this is a 2 b 2 2 raised to the power of minus 2 plus a 2 b 1 2 raised to the power of minus 3 plus a 2 b 0 2 raised to the power of minus 4 plus a 1 b 2 2 raised to the power of minus 3 plus a 1 b 1 2 raised to the power of minus 4 plus a 1 b 0 2 raised to the power of minus 5 plus a 0 b 2 2 raised to the power of minus 4 plus a 0 b 2 2 raised to the power of a 0 b 2 2 raised to the power of minus 4 plus a 0 b 1 2 raised to the power of minus 5 plus a 0 b 0 2 raised to the power of minus 6 plus this entire thing I am not writing this again.

Now, overall final this value is a 3 b 3 plus. So, what are the total 2 raised to the power of minus 1 coefficient this is 2 raised to the power of 0 coefficient, what is 2 raised to the power of minus 1 coefficient? This one 2 raised to the power of minus 1 here nothing is there, so only this one.

So, this is a 3 b 2 bar plus b 3 a 2 bar 2 raised to the power of minus 1 plus 2 raised to the power of minus 2 is this one and here also we have this term ok. So, this total term is a 3 b 1 bar plus b 3 a 1 bar plus 1 plus this term a 2 b 2, this one will be this is 2 raised to the power of minus 2 2 raised to the power of minus 3 is this, this and this.

So, what are the total terms a 3 b 0 bar plus b 3 a 0 bar and here 2 terms a 2 b 1 plus a 1 b 2 this whole thing is 2 raised to the power of minus 3 and here there is no 2 raised to the power of minus 4 only these 2 terms 2 raised to the power of minus 5 is these 2 terms 2 raised to the power of minus 6 is only this term.

So, what is the coefficient of 2 raised to the power of minus 4 is a 1 b 1 plus a 0 b 2 a 0 b 2 is 2 raised to the power of minus 4 2 raised to the power of minus 5 is a 1 b 0 plus a 0 b 1 and a 0 b 0 2 raised to the power of minus 6 this is the complete expression for y.

Now, I can map this on to the algorithm. So, if I take the algorithm a 3 dot a 2 a 1 a 0 into b 3 dot b 2 b 1 b 0. So, in case of Baugh Wooley multiplier. So, what is 2 raised to the power of 6 coefficient a 0 b 0 here a 0 b 0. Simply this itself is p 0 and 2 raised to the power of minus 5 is here you will get 2 terms one is a 1 b 0 ok and if I say multiply this with this one and then we will get a 2 b 0 ordinary multiplication b 0 a 0 a 1 b 0 a 2 b 0 a 3 b 0 what is this wherever this a 3 or b 3 is there we will take the complement.

Similarly, with b 1 this one is a 0 b 1 a 1 b 1 a 2 b 1 a 3 b 1 bar because a 3 is there. So, whenever a 3 is there or b 3 is there we will take the complement. Similarly, with b 2 a 0 b 2 a 1 b 2 a 2 b 2 a 3 b 2 bar similarly with b 3 because b 3 is there all complements. So, this is b

3 a 0 b 3 bar because b 3 is sign bit a 1 b 3 bar a 2 b 3 bar and a 3 b 3 both are sign. So, there is no complement.

So, if we see here now. So, this one is coefficient of 2 raised to the power of 0 this one is 2 raised to the power of minus 1 because here dot will be there this is 2 raised to the power of minus 2 this is 2 raised to the power of minus 3, 2 raised to the power of minus 4, 2 raised to the power of minus 5, 2 raised to the power of minus 6. We can easily verify this 2 raised to the power of minus 6 is a 0 b 0 this we will get as p 0 final output and p 1 is 2 raised to the power of minus 1 term a 1 b 0 a 0 b 1 a 1 b 0 a 0 b 1 this is b 1.

2 raised to the power of minus 4 term is a 1 b 1 a 1 b 1 a 1 b 1 plus a 0 b 2 a 0 b 2 and there will be one more term like a 2 b 0 here somewhere a 2 b 0 is missing. So, here a 2 b 0 is also there this a 2 b 0 is also there. So, plus a 2 b 0 into this one this is p 2. Similarly, 2 raised to the power of minus 3 is we have a 3 b 0 bar b 3 a 0 bar a 2 b 1 b 2 a 1 this is p 3 2 raised to the power of minus 2 is here we have to add a1 extra also this one a 3 b 1 bar b 3 a 1 bar then a 2 b 2.

So, this is p 4 then 2 raised to the power of minus 1 is this a 3 b 2 bar a b b 3 a 2 bar p 5 and last one is this is p 6. So, totally we will get 7 bit output. So, this is about the algorithm of this one. So, we can map this on to the architecture. So, we will discuss this architecture and then corresponding VERILOG code in the next lecture.

Thank you.