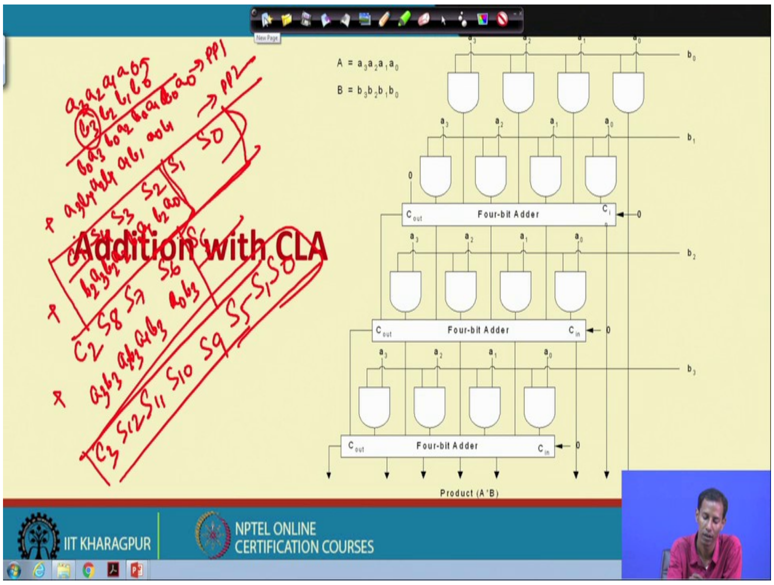
Architectural Design of Digital Integrated Circuits

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

Lecture - 29
Multiplier Architecture (Contd.)

So, welcome back to the course on Architectural Design of IC's. So, we are seeing different Multiplier Architecture. So, in the last class we have seen the; that means, we have seen that array multiplier architecture is also there. There is also based on that the fundamentals are that Shift and Add ok. So, there are different type of array ; that means, array multiplication ok. So, we will see what type of array multiplication is there and how they basically operate and what is the advantage and what is disadvantages of them. So, that we will discuss today.

(Refer Slide Time: 00:52)



Addition with CLA

A = $a_3a_2a_1a_0$

B = $b_3b_2b_1b_0$

So, this basically this array multiplication the basic thing is that whatever we have learnt that bitwise multiplication and then addition ok. So, here suppose I am considering 2 A and B 2 numbers 4 bit ok. At the very beginning what is happening? I am having this 4 AND gate over here and 4 AND gate over here. That means, if I just do it something like this, so at that time what is happening?

So, if I am multiplying a 3 a 2 a 1 a 0 with b 3 b 2 b 1 and b 0. So, what I have to do? So that means, initially what will happen? B 0 will be multiplied by each bit of this. So that means, now b 0 a 3 b 0 a 2 b 0 a 1 a is b 0 a 0 right. Then that will be 1 bit shift for b 1 and b 1 will start from here. So, it will be a 0 b 1 plus a 1 b 1 plus a 2 b 1 plus a 3 b 1 ok.

Then what will happen? This is 1 this is partial product 1 this is partial product 2. So, this will be added ok. So, I will get after addition what I will get? I will get sum, 4 sum 3 sum 2 sum 1 sum 0 and then therefore, there will be carry 1 also. Then again in the next what will happen? This I have to start this b 2 and if b 2 will start from here S 2 position. So, now, what I have to do? This b 2 a 0 b 2 a 1 b 2 a 2 and b 2 a 3; then again this I have to add. As there is this 2 position is 0 0; that means, now I do not have to consider these 2 particular case, I have to add only this 4.

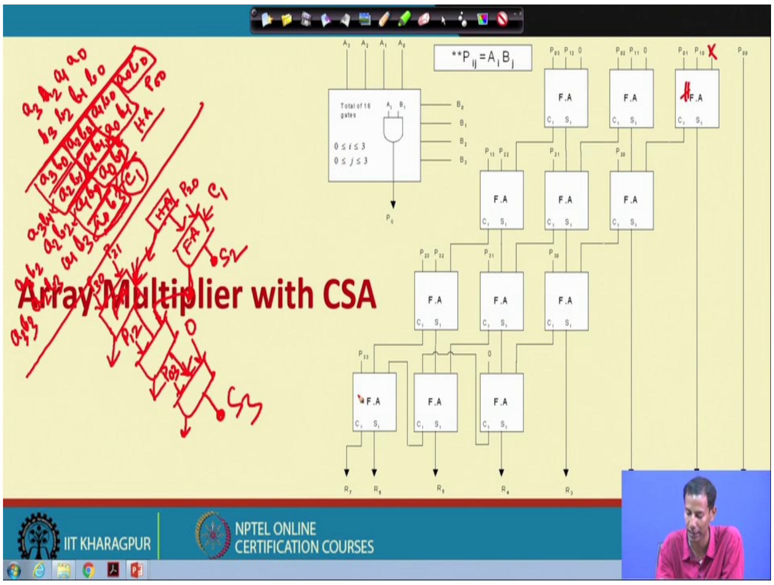
So, then again I can produce this S 5 S 6 S 7 S 8 as well as carry. Then again along with this will start from what? This fourth position; that means, 1 2 3 from S 6; then again I have to add this a 0 b 3 a 1 b 3 a 2 b 3 and a 3 b 3 ok. So, now, S 5 competition is already over. Now, after that again I have to do this. So, here what I will get? S 9 S 10 S 11 S 12 and then C 3 over here.

So, finally, I do not have to do this ANDing operation over here; that means, all the bit of b I have already done completed. So, now, finally what will be my output? S 5 S 1 and S 0; so, this is the final product output of this.

So, in this manner what I am doing? Initially b 0 is my ANDed with all the bits of a and b 1 that has shifted by 1 and all the bits are ANDed with a, they are being added. Where this 4 bit adder, there being; that means, now there they have been done by using this carry look ahead adder ok.

So; that means, now at the very beginning it is producing this 4 bit sum ok. So, apart from that that last sum that is directly coming out then again with that tree and carry out then that is again added with the b 2 a 3 b 2 a 2 b 2 a 1 b 2 a 0 and then again it is added I have over here and again the carry bit again it is coming over here. The last sum that is coming out from this and finally, I am getting the product something like this ok. So, this is the array multiplication where I am using this 4 bit adder which are made by carry look ahead adder. So, this is carry, this CLA means this is carry look adder ok.

(Refer Slide Time: 05:39)



Array Multiplier with CSA
**P_{ij} =A_iB_j
F.A
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, then next what I can do? The same thing; so, the same thing what I can do if I use carry save adder? So, instead of that we have used carry look adder. So, here I will use carry save adder. So, carry save adder means if I just again rewrite this particular equation a 3 sorry a 2 a 1 a 0 b 3 b 2 b 1 and b 0, then again what is there? That means, now a 3 b 0 a 2 b 0 a 1 b 0 a 0 b 0 right, then this is a 3 b 1 a 2 b 1 a 1 b 1 a 0 b 1 right, then again there is a 3 b 2 a 2 b 2 plus a 1 b 1 plus a 0 b 1 sorry b 2 b 2 b 2 b 2 ok, then again at this position a 3 b 3 then again this is a 2 b 3 this is a 1 b 3 and this is a 0 b 3 ok.

Now, you see that for this position there is no addition operation right, but for these 2 this is bit because this is ANDing operation this is 1 bit this is 1 bit. So, for these 2 bits if I want to add I need what? I need one half adder for these 2 and whenever I will do this half adder operation, so, at that time the carry will be generated and that will be propagated to the next level. That means, if I consider these as carry 1; so then that will come over here. So that means what? Now, for doing this I need what? I need 3 bit operation or along with that I have to add this carry 2 ok.

So that means, here you see that for this is the direct P 0. So, this is P 00 ok, then in this particular this if I if they; that means, if I consider not full adder half using half adder, so 0 carry in means this is nothing but one half adder circuit right. So that means this is half adder if I do not consider this. So, then that carry will again come over here; so that means, now initially what I will consider for this case? I can consider either of this 2 using one half adder ok. So, this is what? This is this P 11 means a 1 b 1 and P 02 means what? This is this sorry a 0 b 2 ok. So, if I add this a 1 b 1 a 0 b 2 sorry a 0 b 2 in the first using 1 half adder; that means what? In half adder then again from this particular half adder circuit then again that will be added with what? This a 2 b 0; that means, that is P 20 along with the corresponding C 1 which is the previous carry from the previous stage ok.

So, this is one full adder cell and it will be produced the corresponding this is S 2 and again this half adder this; that means, this half adder carry then this will be connected to the next and the carry generated from this then again that will be connected or that will be propagated to the next ok. So that means what as I am doing 4 of this addition operation over here, so I will get 2 carry. So, 1 carry from here 1 carry from here and both of them will be propagated to the next level then again you consider this two and this two along with the carry over here carry over here.

So that means, for this particular case I need what? One full adder considering these P 30 and here this is P 21 along with the carry of this to this; this is one full adder cell then again this along with these is the; that means, this again this will generate the carry ok.

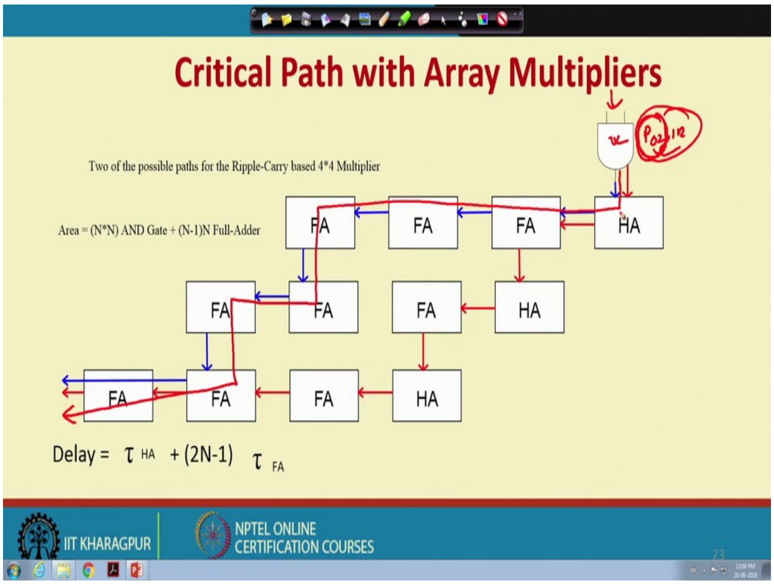
So, then again I need what? The sum of these, that will be pass to the corresponding this a 2 b 1 sorry a 1 b 1 to the sum and this will be the carry in for this particular case ok. Then again this will produce what? The sum which will be added with the corresponding P, this is 03 and the carry in over here that is 0 because no further carry is generated from the previous stage ok. So, finally, I will get S 3 over here and it will generate 3 carry at this point ok.

So, here you see that this is the half adder case or this is also considering this case half adder then, that carry is generated for full adder this 2 0 is considered in this case along with this sum of from this sum and the previous carry. So, it is generating S 1 as well as it is producing 2 carry.

So, in a next stage; that means, for this particular case. So, initially what I am considering? P 0 3 means a 0 b 3 and P 1 2 means a 1 b 2 that has been added with 0 then the sum that is added with carry in along with this a 2 and b 1 ok. Then the sum it is put to producing the sum then that is added with a 3 b 0 and the previous carry is coming over here. So that means, the initially it is 0 the carry in is 0 because I have to add 2 carry which is generating from the previous stage. So, 2 carry from the previous stage. Again as there are 3; that means, stage in this case.

So; that means, it will produce 3 carry. So, again 3 carry will come over here and that will be considered something like this in this manner, this is the array multiplication considering the carry save adder; that means, in each stage am saving the carry and am just propagating the carry to the next level.

(Refer Slide Time: 13:06)



Critical Path with Array Multipliers

Two of the possible paths for the Ripple-Carry based 4*4 Multiplier

Area = (N*N) AND Gate + (N-1)N Full-Adder

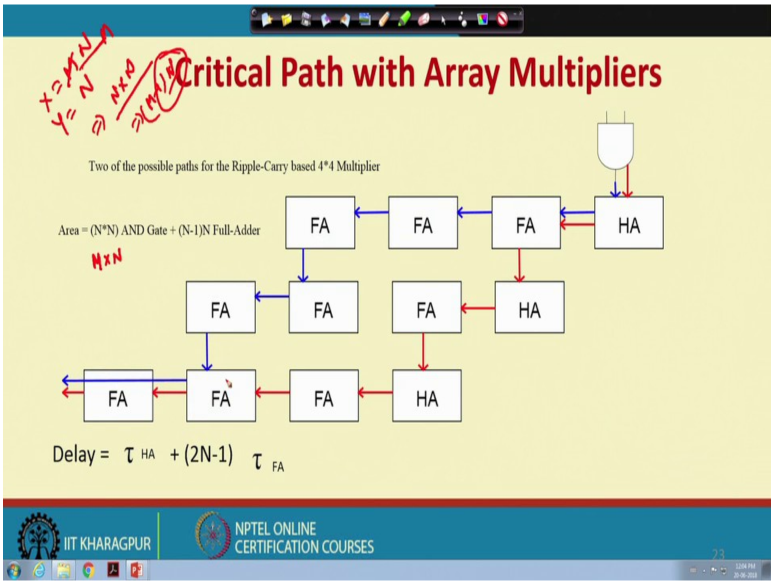
Delay = $\tau_{HA} + (2N-1)\ \tau_{FA}$

So, using that things also I can do. Then what is the corresponding; that means, delay for this curve; that means, what is the critical path for this array multiplication? The critical path is this; that means that is basically coming from these to these, to these then these then these to this; that means this is the longest path which I am getting. So why ANDing? ANDing is basically for generating the P 02 or P 12, I need one AND gate to generate the corresponding partial product. So, that AND gate delay will also come. Then as the carry is basically if I just go back, if I just go back as the carry is basically propagating from each of these stage ok. So, that is why it requires that type that time as the critical path for this array multiplication. So, in this case if I consider this when it is passing from these to these to these to these to then again these to these to this something like this ok.

So, the number of full adder cell it is passing from that is half adder then 1 2 3 4 5 6; 6 full adder cell. Apart from that 1 2 3 4 5 6 7 ok, then again 7, but here you consider the 2 half adder cells are there ok; so, 2 half adder? Here how many full adder? 1 2 3 4 5 6 full adders ok, but if I consider the another path where it is considering 1 2 3 4 5 full adder and 2 half adder. So, 2 half adder delay can be considered as 1 full adder delay. So, 6 full adder delay in this particular case along with 1 half adder delay and in this path also 6 full adder and 1 half adder and 1 and delay ok. So, this delay so, the area is if I consider N bit, so at that time the area will be N star N AND gate, why N star N AND gate; that means, I need to produce that number of partial products ok.

So, if the number is if the number is; that means, if X as of N bit and Y is of N bit, so at that time this AND gate requirement will be M into N. As both are of N so that is why the AND gate requirement is this and this is there is N minus 1 into N number of full adders ok.

(Refer Slide Time: 15:40)



Critical Path with Array Multipliers

Two of the possible paths for the Ripple-Carry based 4*4 Multiplier

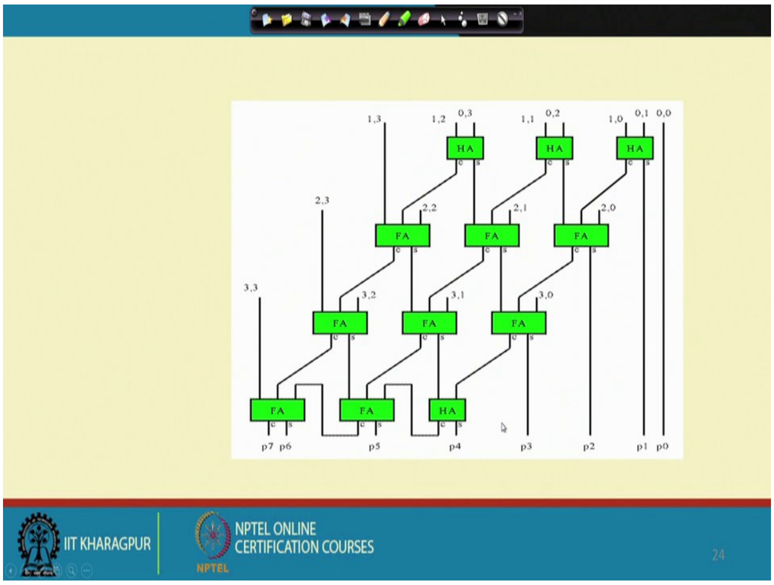
Area = (N*N) AND Gate + (N-1)N Full-Adder

$\mathrm{Delay} = \tau_{HA} + (2N-1)\,\tau_{FA}$

So, why is so? Because I am having this N of; that means, this is the multiplier and this is the multiplicand block ok. So, that is why I need this N minus 1 into N number of full adders because I have to add and that is in a 2 D matrix right. If there is M and there is, so at that time it will be M minus 1 into N and the delay is one half adder plus 2 N minus 1 into this that number of full adder delay is the total delay requirement for this particular case ok.

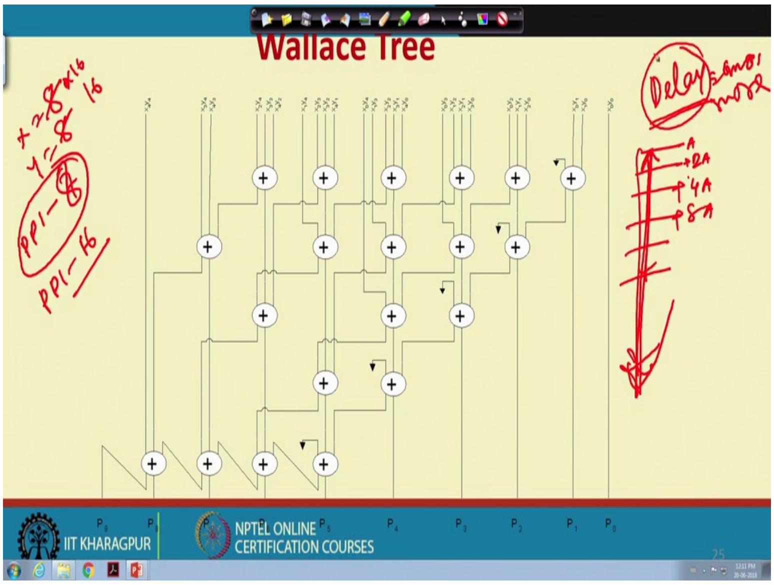
So that means, this if I as this is N cross N, so that is why this is 2 N minus 1, otherwise, it will be what? M plus N minus 1 ok, if this that means, y of N and x of M. So, at that time this number will be M plus N minus 1 number of full adder delay ok. So, this is the critical path of this array multiplication.

(Refer Slide Time: 17:35)



So, this is the; that means, full adder; that means, the array multiplication we have seen.

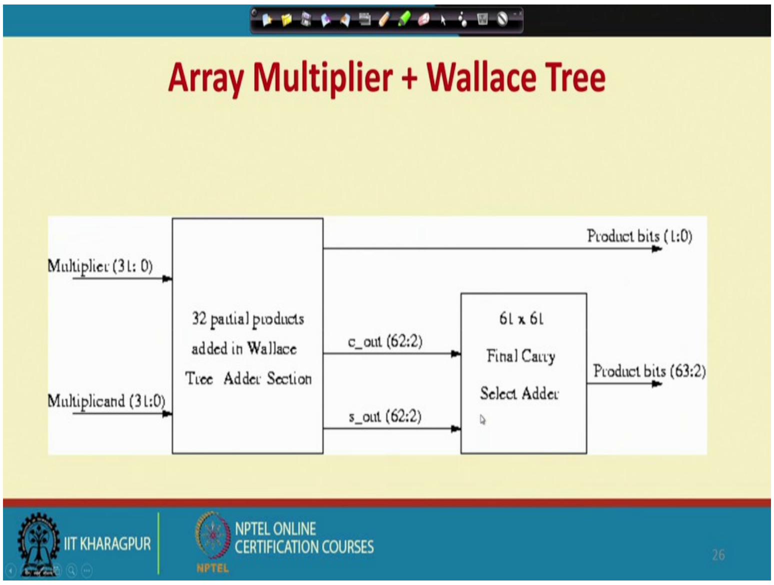
(Refer Slide Time: 17:39)


Wallace Tree

So, this; that means, what we have seen this in this fashion. So, this is just the; that means, some modification to that this is known as this Wallace tree; so that means, like a tree structure we are just adding each of this partial products ok. So, here each of this is ground means here you see these are the nothing but this full adder cells and this grounded means the one of the input is connected to 0 and at the final, at the final stage what I need? I need one 4 bit addition operation which is basically where the carry is basically rippling ok.

So, this is tree structure for doing the multiplication operation and this structure is known as Wallace tree architecture that is nothing but the array multiplication considering the CSA ok.

(Refer Slide Time: 18:54)



Array Multiplier + Wallace Tree

Multiplier (31: 0)

Multiplicand (31:0)

32 partial products added in Wallace Tree Adder Section

Product bits (1:0)

c_out (62:2)

s_out (62:2)

61 x 61 Final Carry Select Adder

Product bits (63:2)

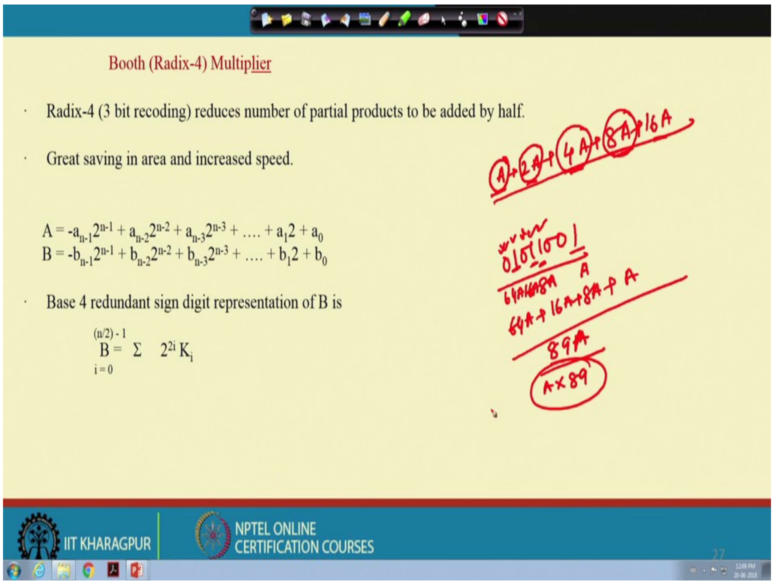
IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

26

So, this is the; that means, this is the block diagram of this array multiplier with Wallace tree; that means, here this array multiplication is being done over here and this final stage addition operation that has been done through this carry select adder to finally, find out the corresponding product bits of this. So, initial bit has been generated from this and the other this; that means, 60 as this is 32 bit multiplier; 32 into 32. So, that has been generated from this final carry select adder ok. So, the final carry select adder has been generated from this ok.

(Refer Slide Time: 19:50)



Booth (Radix-4) Multiplier

- Radix-4 (3 bit recoding) reduces number of partial products to be added by half.
- Great saving in area and increased speed.

$A = -a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + a_{n-3}2^{n-3} + \dots + a_1 2 + a_0$
$B = -b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + b_{n-3}2^{n-3} + \dots + b_1 2 + b_0$

- Base 4 redundant sign digit representation of B is

$$B = \sum_{i=0}^{(n/2)-1} 2^{2i} K_i$$

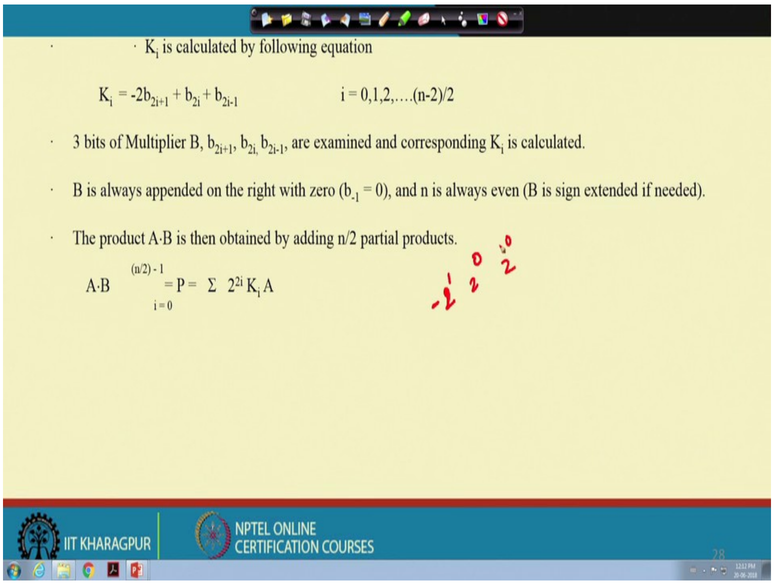
So, then there is another method of doing the multiplication operation that is Booth’s multiplier ok. So, in Booth’s multiplier initial actually in multiplier what we do, in multiplier what we do? That means, what I said that we basically added something like this, something like this we do and based on this; that means, what will be the this values, based on the multiplicands value if that bit is based on this each of this particular bit position ok, the value of this I will select what will be the corresponding value I need to add ok.

So, if this position is 1, so at the time I have to add A, if this position is 1; that means, I have to add 8 A; if this position is 1; that means, I have to add 16 A; if this position is 1 that time I have to add 64 A. So, finally, what I have to do based on this finding the 1? I have to add this 16 A plus 8 A plus A and finally I will get 89 A. So 89 A means, A along multiplied with 89.

So, the final results I will get, but in Booth’s that means, multiplier what is the problem in this with this array multiplication is that we are considering each of this bit position ok. So, if I just go back, here each of this bit position we are considering to find out the corresponding partial products. So, the more the number of the that means, if I consider ok, if I consider x of 8 and y of 8 so at that time the partial products that I need to generate of 8 combination. If this is 16 sorry, if this is 16 cross 16, so at that time PP 1 to sorry 6 this is for 8 up to 16 generate and then I need to add so; that means, now there will be 16 stage which I need to add this. In each of this is basically I have to add, something like this I have to add.

So, more of this depth means the delay will be more, delay will be more ok. So that means, if I consider the word length more means I am increasing the delay. So, people are thought that why cannot we reduce the number of partial if I can reduce this depth; that means, at that time I can reduce the delay. So, how can I reduce this particular depth so that I can get the benefit or I can reduce the delay? So, that is why people have done what? This people have done this Booth's multiplication ok. So, this says that there is a decoding technique; that means the multiplicand that has been decoded. So, how it has been decoded? It is basically based on this base 4 redundant sign digit representation of B ok.

(Refer Slide Time: 24:18)



- K_i is calculated by following equation

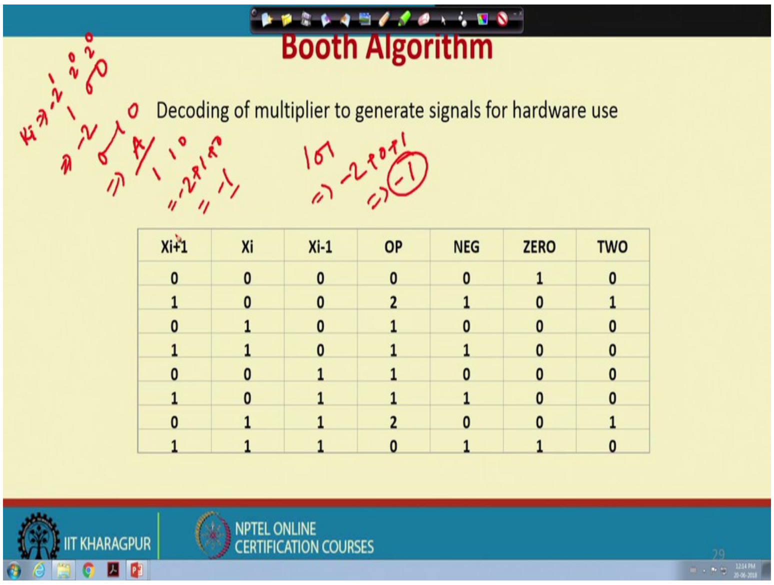
$$K_i = -2b_{2i+1} + b_{2i} + b_{2i-1} \qquad i = 0,1,2,\ldots.(n-2)/2$$

- 3 bits of Multiplier B, b_{2i+1}, $b_{2i,}$ b_{2i-1}, are examined and corresponding K_i is calculated.
- B is always appended on the right with zero ($b_{-1} = 0$), and n is always even (B is sign extended if needed).
- The product A·B is then obtained by adding n/2 partial products.

$$A \cdot B = P = \sum_{i=0}^{(n/2)-1} 2^{2i} K_i A$$

So, how is that? The corresponding value has been calculated based on this. The position is a like this 2 b 2 y plus 1 plus b 2 y plus b 2 y minus 1 ok. So that means what? That means what? Based on this particular and this is this Booth's multiplication is basically used for sign multiplication ok; signed and unsigned both. So, if it is unsigned number that also at that time also you will get the correct result and if it is signed number at that time also you will get the correct results. So, how they it has been decoded? The decoding technique has been done based on this particular equation where this says that if that is the position is something like this that is 2 to the power minus of 2 to the power 1 then 2 to the power 0 and then 2 to the power again that is 0 ok.

(Refer Slide Time: 25:35)



Booth Algorithm

Decoding of multiplier to generate signals for hardware use

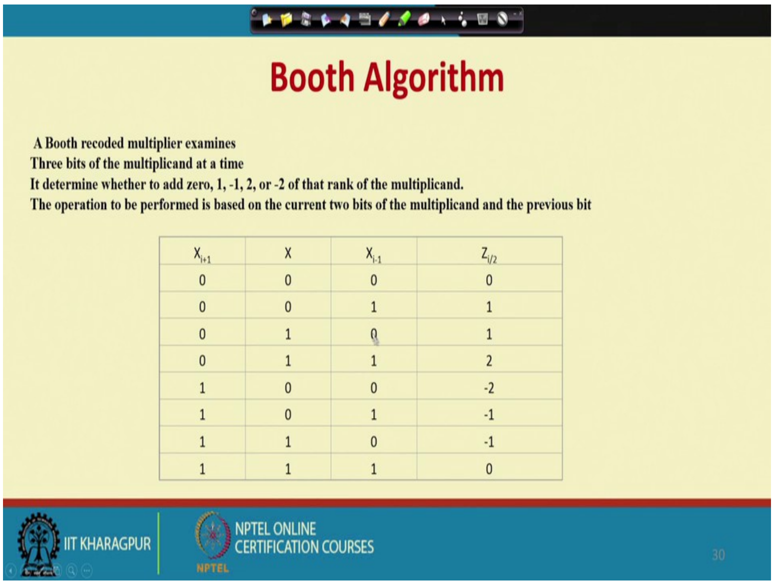
Xi+1	Xi	Xi-1	OP	NEG	ZERO	TWO
0	0	0	0	0	1	0
1	0	0	2	1	0	1
0	1	0	1	0	0	0
1	1	0	1	1	0	0
0	0	1	1	0	0	0
1	0	1	1	1	0	0
0	1	1	2	0	0	1
1	1	1	0	1	1	0

So that means, now based on these the corresponding decoding technique is ok, the decoding technique is what I said? 0 0 0 the operation is basically 0 ok, then is the decoding technique using this Booth's multiplier. If that is 1 0 0, then at that time that will be 2 along with 1; that means, negative is 1; that means, this will be minus 2. Why? Because what I said that decoding technique says that that is the final; that means, this equation is 2 to the power 1 and 2 to the power 0 and 2 to the power 0.

So, if this bit is 1 0 0; that means, this will be minus 2; then again 0 for 0 1 0 1 0 for 0 1 0 this is nothing but A or just the 1. For 1 1 0, for 1 1 0 what will be the case? 1 1 0 this is minus 2 plus 1 plus 0 means minus 1. So, here you see for 1 1 0, 1 with negative of 1 and in the same fashion for 0 0 1 that is 1 no negative, for 1 0 1 again 1; for 1 0 1 what will happen minus 2 plus 0 plus 1 that is equals to minus 1 ok. So, for 1 0 1; 1 with negative of 1, for 0 1 1; that is 2 with positive, 1 1 1 that is 0 ok, but 0 means it is 0 ok.

So, based on this; that means, the major; that means, the weight which I need to consider that is that can be computed using this particular equation. So, the Booth's decoding, whenever we will follow this Booth's multiplier at that time this decoding table can be formed if you can remember this particular equation ok.

(Refer Slide Time: 28:01)



Booth Algorithm

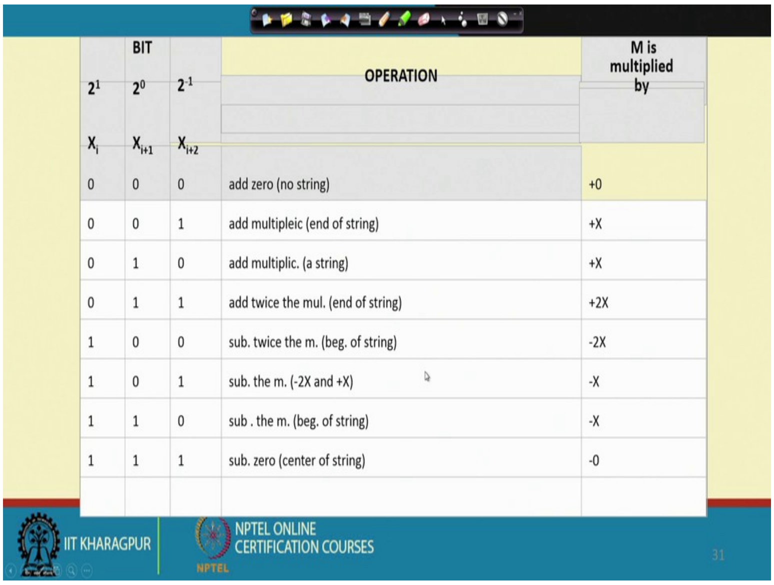
A Booth recoded multiplier examines
Three bits of the multiplicand at a time
It determine whether to add zero, 1, -1, 2, or -2 of that rank of the multiplicand.
The operation to be performed is based on the current two bits of the multiplicand and the previous bit

X_{i+1}	X	X_{i-1}	$Z_{i/2}$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	2
1	0	0	-2
1	0	1	-1
1	1	0	-1
1	1	1	0

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES 30

So, now, if we take the example of this, if we take the example of this, so, this is the; that means, operation I need.

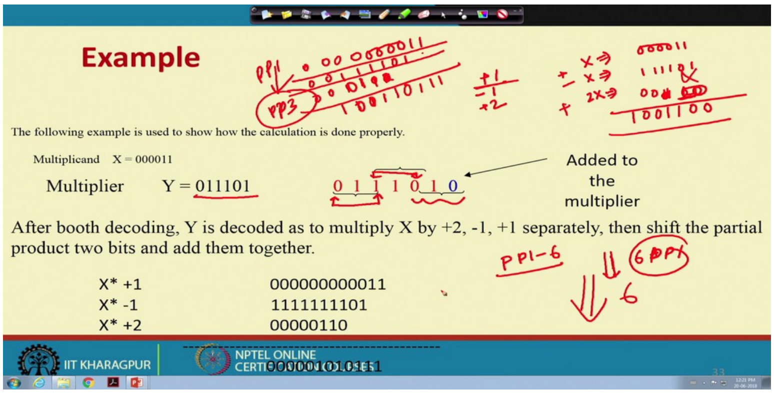
(Refer Slide Time: 28:04)



	BIT		OPERATION	M is multiplied by
2^1	2^0	2^{-1}		
X_i	X_{i+1}	X_{i+2}		
0	0	0	add zero (no string)	+0
0	0	1	add multipleic (end of string)	+X
0	1	0	add multiplic. (a string)	+X
0	1	1	add twice the mul. (end of string)	+2X
1	0	0	sub. twice the m. (beg. of string)	-2X
1	0	1	sub. the m. (-2X and +X)	-X
1	1	0	sub . the m. (beg. of string)	-X
1	1	1	sub. zero (center of string)	-0

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES 31

(Refer Slide Time: 28:13)



Example

The following example is used to show how the calculation is done properly.

Multiplicand X = 000011

Multiplier Y = 011101

0 1 1 1 0 1 0 ← Added to the multiplier

After booth decoding, Y is decoded as to multiply X by +2, -1, +1 separately, then shift the partial product two bits and add them together.

```
X* +1        000000000011
X* -1        1111111101
X* +2        00000110
-----------------------------
             000001010111
```

So, if we take 1 example of this, so, at the time that will be much clearer to you. Suppose, this the multiplicand X and the this is the multiplier Y, so, initially whenever we will do or we will follow this Booth's multiplication, so, at that time what I need to do? I need to add 0 at the LSB position. After that what we will do? We will consider this the combination of 3 over here for the LSB side first.

So, for 0 1 0 for 0 1 0 what is the combination? That is plus 1 and then again actually I have to overlap for the last for the MSB of this or the last bit position I have to overlap. So that means, at that time whenever I will consider the next set 3 bit set at that time I will consider from this to this ok, so, for 1 1 0 that is minus 1.

Again for the next set what I will consider? I will consider these 3 and for these for 0 1 1 that is plus 2 ok. So, now, for X plus 1 means what? That means, it will be the operation is that the multiplicand that will be plus 1 ok. So, plus 1 means this is nothing but the X ok. So, in the next it is just the 2's compliment of that ok, so that means, in the next that is minus X and in the next that is plus X plus 2 plus 2 means that is 2 of x. So, X means what? X means if from this if I consider, so X means 11.

So, then minus of X means in 2's compliment format if I just write that is 0 and then all 1 ok. For twice of X means what? The two 0 will be appended at the very last LSB and it will be something like this. Now, I have to add all these 3 for this 3 set and here what I will get? 0 over here, then 0, then again here 1, then again here 1, then 0, then 0, then 1 ok, so, this is the final result. What is that? Ok sorry. This is not something like this actually I forgot that, actually I have to do this shifting operation 2 ok.

So that means, at these first position this is 000011 ok. So, whenever I am considering this; that means, at that time 2 bits shift I have to do because 3 bit decoding we are using right. So, this minus X will start from this particular position that is 1 0 0 1 1 1 1. So, this is all 0. So, then again again 2 bit shift and this 2 X will be started from this particular position, ok. So, if I just want to add then 1 0 1 1 0 1 1 ok?

Sorry this is plus 2 means this is only 1; so that means, 1 it will be 1 1 1 0 1 then again 1 over here for 1 this is 0 1 ok. So, now, I will get 1 0 1 0 1 1; actually this is 2 X means not here what I am doing I am 4 not 4 this will be 1 1 0; just 1 bit; that means, 1 0 at the LSB position. If I have to do 4 at the time I will I will append 2 0 at the LSB position I have done that mistake at that time. So, that means now if you see what my point is that; that means, how many products; that means, partial products now I am generating though I have considered this y of 6 bit? So using the array multiplier if I want to do this, so at that time how much partial products I will get? I will get total 6 set; so that means, now the adder depth delay if I just say ok. So, that will be 6 at that time 6 partial product set I have to add. So that means, the depth of that will be 6, but here as I have used this Booth's multiplication decoding technique at that time this partial products set that has been reduced to 3 only.

So that means I can reduce the depth. So, whenever I am reducing the depth at that time I can reduce the delay too. But, here at what particular cost? So, initially I need one decoder table for this Booth's decoding operation, according to the Booth's algorithm I need this decoding table at the very front of this multiplier or this is basically nothing but the array multiplication. After this decoding what will be the corresponding partial product generation? So, that it will come and we will see that ok. So, this is the technique for this Booth's multiplication.

So, here we are considering radix 4. So, that is why we are considering something like these. If you just want to change the radix, so at that time I you can consider more of this; that means, 4 bit you can consider and at the time what where will be the change where will be the change? At that time, at that sorry at that case at that time there will be the change in this particular location. So, we will consider more on to this Booth's algorithm and if I change want to change the radix and at that time what will happen, so, that we will discuss in the next class. So, this is it for today.

Thank you.