

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

Lecture - 43
Cordic Architecture (Contd.)

Hello every one. So, welcome back to the course on Architectural Design of ICs. So, in the last 2 classes we are basically describing or we are just seeing the Architecture of Cordic. So, using the cordic the use of pseudo rotation we basically the cordic algorithms is became simpler or it becomes simpler in terms of hardware cost, because of this use of this pseudo rotation or because of this pseudo rotation again it causes; that means, this 2^{-i} to the power minus i angle rotation which is pre defined ok.



So, based on that now any angle you can be calculated. So, what I said that and how we can calculate that also we have seen in the last class. So, whenever we are calculating or we are introducing this pseudo rotation in the cordic algorithm, there is some of the side effect of that or some of the factors, which are being associated. So, what is that; that means, effect or what are those effect which is basically has been introduced because of this considering the fact of this pseudo rotation.

And we have seen that in the circular rotation though it is this magnitude of r becomes addressed in the pseudo rotation and that has been increased by a factor. So, that is specifically known as scaling factor. So, now, we will see what is that scaling factor and what is value of scaling factor, how it is being calculated or how we are getting and why this scaling factor introduced in the cordic algorithm so, that we will see now.

(Refer Slide Time: 02:24)

The Scaling Factor

- The Scaling Factor is a by-product of the pseudo-rotations.
- When simplifying the algorithm to allow pseudo-rotations the $\cos\theta$ term was omitted.
- Thus outputs $x^{(n)}, y^{(n)}$ are scaled by a factor K_n where:
$$K_n = \prod_n 1/(\cos\theta^{(i)}) = \prod_n (\sqrt{1 + 2^{(-2i)}})$$
- However if the number of iterations are known then the **Scaling Factor** K_n can be precomputed.
- Also, $1/K_n$ can be precomputed and used to calculate the true values of $x^{(n)}$ and $y^{(n)}$.

 IIT KHARAGPUR  NPTEL ONLINE CERTIFICATION COURSES 26

So, the scaling factor is a by-product of the pseudo rotation. So, what I said and we have already seen that because of the pseudo rotation from the original circular rotation as this the corresponding vectors that has been increased or; that means, magnitude of the vectors that has been increased. And when simplifying the algorithm to alive the pseudo rotation in the cos theta term was omitted that also we have seen.

So, thus the outputs x_n and y_n are scaled by a factor K_n , where K_n is the summation of 1 by $\cos\theta_i$; why this is being scaled? Because we have omitted why because the equation is $\cos\theta$ into $x_1 - y_1 \tan\alpha_i$ so, we are in this particular we are considering the fact in this cordic. So, at the time we have omitted that $\cos\theta$ terms, we have considered only this $x_i - y_i \tan^{-1}\alpha_i$. Where α_i we have again that has be we have considered that as d_i into 2^{-i} to make that only sit and add base logic.

So; that means, because of this omitting this $\cos\theta$ terms. So, that will be gathered together. So, this will be gathered together for how many times or how many iteration I have done. So, as I have done i number of iteration so; that means, how; that means, the output of this x_n and y_n that will be scaled by the factor of K_n equals to summation of 1 divided by $\cos\theta_i$ which is nothing, but the; that means, the square root of $1 + 2^{-2i}$. So, why this is? Because here that is $\tan^2\alpha + \cos^2\theta$ means what; square root of $1 + \tan^2\theta$. So, $\tan\theta$ again that

has been how we are considered that is a in terms of 2 to the power minus i. So, tan square means that is 2 to the power minus of 2 i.

So; however, if the number of iteration are known, then the scaling factor K_n can be precomputed so; that means, as I know this how many; that means, how many stages I will use this. So, the scale factor also I can compute by using this equation ok. And along with that $1/K_n$ can be precomputed and used to calculate the true values of x_n and y_n . So, how we can calculate? Because whatever value I will get x_n and y_n that are been scaled by the factor of K_n .

So, K_n I know or I can calculate using this equation as I know this values of i, then the original values of x_n and y_n that can be calculated by dividing K_n ; because that has been this computed from the cordic algorithm whatever x_n and y_n I am getting that has been scaled up by the factor of K_n so, now, I have to scale down that. So, $1/K_n$ it will scale down to its original value ok. So, this is the scaling factor this is being introduced because of this pseudo rotation or this omitting of this cos theta term.

(Refer Slide Time: 06:10)

To simplify the Givens rotation we removed the $\cos\theta$ term to allow us to perform pseudo-rotations. However, this simplification has a **side-effect**. The output values $x^{(n)}$ and $y^{(n)}$ are scaled by a factor K_n known as the Scaling Factor where:

$$K_n = \prod_{i=0}^{n-1} 1/\cos\theta^{(i)} = \prod_{i=0}^{n-1} (\sqrt{1+\tan^2\theta^{(i)}}) = \prod_{i=0}^{n-1} (\sqrt{1+2^{(-2i)}})$$

↳ $K_n \rightarrow 1.6476$ as $n \rightarrow \infty$
 $1/K_n \rightarrow 0.6073$ as $n \rightarrow \infty$
 n = number of iterations

However, if we know the number of iterations that will be performed then we can precompute the value of $1/K_n$ and correct the final values of $x^{(n)}$ and $y^{(n)}$ by multiplying them by this value.

So, now if I say that to simplify the given rotation we remove the cos theta terms to allow us to perform the pseudo rotation; however, this simplification has a side effect and thus the output values of x_n and y_n scaled by the factor put the same logic whatever I am telling you. And the; that means, if I considered that n ok; that means, up to infinity the values will be K_n values will be 1.6476. So, $1/K_n$ equals to what? 0.6073 ok.

(Refer Slide Time: 06:54)



Rotation Mode

- The CORDIC method is operated in one of two modes;
- The mode of operation dictates the condition for the control operator d_i ;
- In Rotation Mode choose: $d_i = \text{sign}(z^{(i)}) \Rightarrow z^{(i)} \rightarrow 0$;
- After n iterations we have:

$$x^{(n)} = K_n(x^{(0)} \cos z^{(0)} - y^{(0)} \sin z^{(0)})$$

$$y^{(n)} = K_n(y^{(0)} \cos z^{(0)} + x^{(0)} \sin z^{(0)})$$

$$z^{(n)} = 0$$
- Can compute $\cos z^{(0)}$ and $\sin z^{(0)}$ by starting with $x^{(0)} = 1/K_n$ and $y^{(0)} = 0$

So, now considering this fact we can basically run the cordic in the 2 mode ok. So, one of the 2 modes we can run so, this is the basic equation.

(Refer Slide Time: 07:19)

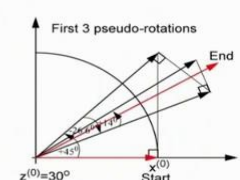


In Rotation Mode the decision operator d_i obeys the condition:

$$d_i = \text{sign}(z^{(i)})$$

Thus, we input $x^{(0)}$ and $z^{(0)}$ ($y^{(0)} = 0$) and then drive $z^{(0)}$ towards 0.

Example: calculate $\sin z^{(0)}$, $\cos z^{(0)}$ where $z^{(0)} = 30^\circ$

i	d_i	$\theta^{(i)}$	$z^{(i)}$	$y^{(i)}$	$x^{(i)}$
0	+1	45	+30	0	0.6073
1	-1	26.6	-15	0.6073	0.6073
2	+1	14	+11.6	0.3036	0.9109
3	-1	7.1	-2.4	0.5313	0.8350
4	+1	3.6	+4.7	0.4270	0.9014
5	+1	1.8	+1.1	0.4833	0.8747
6	-1	0.9	-0.7	0.5106	0.8596
7	+1	0.4	+0.2	0.4972	0.8676
8	-1	0.2	-0.2	0.5040	0.8637
9	+1	0.1	+0	0.5006	0.8657

If I just consider the; that means, how can I calculate the corresponding values? So, initially what is happening? So, this is the starting. So, initially you see y value is 0, but the as this is the position if I considered this is the initial vector in this particular x value will be what; x value is just nothing, but 1 by K_n ; that means, from the beginning what I am doing from the beginning I am putting the values of K_n that is 0.6073 instead of 1 over here.

So, if I put 1 so; that means, that will be whatever results I will get, that will be scaled by the factor of this 1.6476 so, instead of doing that if I put; instead of 1 if I put that as

0.6073. So, I will get the values something the original values of x and y at the after the 10th iteration. So, the same angle for 30 degree how we have calculated at the time? We have not calculated this x and y value. So, x and y value I will get as 0.5006 over here and 0.8657 over here so; that means, this is the corresponding values of sin theta and cos theta; that means, sin 30 and cos 30.

(Refer Slide Time: 08:56)

Vectoring Mode

- In Vectoring Mode choose: $d_i = -\text{sign}(x^{(i)}y^{(i)}) \Rightarrow y^{(i)} \rightarrow 0$
- After n iterations we have:

$$x^{(n)} = K_n \sqrt{(x^{(0)})^2 + (y^{(0)})^2}$$

$$y^{(n)} = 0$$

$$z^{(n)} = z^{(0)} + \tan^{-1} \left(\frac{y^{(0)}}{x^{(0)}} \right)$$
- Can compute $\tan^{-1} y^{(0)}$ by setting $x^{(0)} = 1$ and $z^{(0)} = 0$

So, what we have to do? Actually this is whatever we are doing considering this fact, this is basically rotational mode I am doing or this mode is basically known as rotation mode. Now I can run this in the vectoring mode. The same algorithm I can just change it in a different form. So, that it will be run on the or it can calculate or it can compute different function, where the mode is basically known as vectoring mode.

So, initially in the rotation mode the sign d_i that is dependent on what? That is in dependent on the z whatever is the sign of z that basically defines this x and y ; that means which direction in x and y will be varying so, that basically defines at the time in the rotation mode. But in vectoring mode; that means, the d_i value is basically changed based on this sign of $x_i y_i$ to make this y_n to 0.

So, y_n to 0 if we do so, at the time this x_n will became square root of x_0 square or; that means, this will be x_i square plus y_i square where as this z_n will be as this z_i plus tan inverse of y_i divided by x_i .

(Refer Slide Time: 10:53)

In Vectoring Mode the decision operator d_i obeys the condition:

$$d_i = -\text{sign}(x^{(i)}y^{(i)})$$

Thus, we input $x^{(0)}$ and $y^{(0)}$ ($z^{(0)} = 0$) and then drive $y^{(0)}$ towards 0.

Example: calculate $\tan^{-1}(y^{(0)}/x^{(0)})$ where $y^{(0)} = 2$ and $x^{(0)} = 1$

i	$z^{(i)}$	θ^i	$y^{(i)}$
0	0	45	2
1	45	26.6	1
2	71.6	14	-0.5
3	57.6	7.1	0.375
4	64.7	3.6	-0.078
5	61.1	1.8	0.151
6	62.9	0.9	0.039
7	63.8	0.4	-0.019
8	63.4	0.2	0.009

First 3 pseudo-rotations

Start: $x_0=1, y_0=2$

End

Logos: IIT KHARAGPUR, NPTEL ONLINE CERTIFICATION COURSES

So, how we can do? if you just consider this another example of that. So, initially z equals to what? That is 0 ok. So, here at that time; what we have considered that z value equals to 30 degree, but here z equals to 0. So, the elementary angle first angle is 45. So, for that what is the y that we have to calculate ok. So, that how we can calculate based on that particular equation now we can calculate; that means, after n iteration I will get.

(Refer Slide Time: 11:36)

Shift-Add Algorithm

- Hence, the original algorithm has now been reduced to an iterative **shift-add** algorithm for pseudo-rotations of a vector:

$$x^{(i+1)} = (x^{(i)} - d_i(2^{-i}y^{(i)}))$$

$$y^{(i+1)} = (y^{(i)} + d_i(2^{-i}x^{(i)}))$$

$$z^{(i+1)} = z^{(i)} - d_i\theta^i$$
- Thus each iteration requires:
 - 2 shifts
 - 1 table lookup (θ^i values)
 - 3 additions

Logos: IIT KHARAGPUR, NPTEL ONLINE CERTIFICATION COURSES

So; that means, here now this particular d_i , now it will be changed based on the sign of $x_i y_i$ ok.

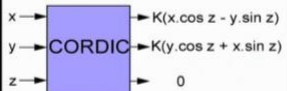
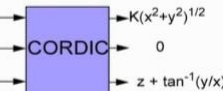
So, z equals to initial z we have considered 30 at that time, but here we are considering 0 at the time, and the corresponding values of y will be noted in this particular case. So,

initially it is when z_i equals to 0. So, initially the y value will be 2, then 1 then; that means, in this particular case z_i value is basically change to another; what is the value ok, what is the value over here? That basically depends on the corresponding values of this y and x ; y and x means this is the tan inverse of y_0 divided by x_0 means what? In the vectoring mode what I said that the decision operator d_i obeys the condition of minus of $x_i y_i$ right.




So, initially you have to or you can calculate this tan inverse of y_i divided by x_i . So, if I put that y equals to y_0 equals to 2 and x_0 equals to 1 that means, tan inverse of 2 what is the values of tan inverse of 2 that you can compute using this particular mode. So, initially y_0 equals to 2 then based on the equation this will be 1, based on this it will be 1 means if you just go back if you just consider that particular; that means equation so, it will be varying in this fashion and finally, whatever the array; that means, the value will be accumulated in z_i . So, that is my value of tan inverse of this particular function ok. (Refer Slide Time: 13:45)

Circular Coordinate System

- So far only pseudo-rotations in a Circular Coordinate System have been considered.
- Thus, the following functions can be computed:

Coordinate System	Rotation Mode $z^{(0)} \rightarrow 0; d_i = \text{sign}(z^{(0)})$	Vectoring Mode $y^{(0)} \rightarrow 0; d_i = -\text{sign}(x^{(0)}y^{(0)})$
Circular	 <p>For $\cos z$ & $\sin z$, set $x = 1/K, y = 0$</p>	 <p>For $\tan^{-1} y$, set $x = 1, z = 0$</p>

- However, more functions can be computed if we use other coordinate systems.

So that means this circular coordinate system. So, this is basically known as circular coordinate system in circular coordinate system considering this pseudo rotation, in the rotation mode whenever this d_i is dependent on the sign of z_i . So, if I put this three value x y and z where z is the input angle I am putting, and x is the 0.6073 if get, then k value at the output, you will get this corresponding value which is this $x \cos \theta$ and $y \sin \theta$. Y value equals to 0 means y equals to 0 means so, this is 0. So, $x \cos \theta$ and in this particular location this $x \sin \theta$ so; that means, what $\cos \theta$ and $\sin \theta$ I am getting ground this particular cordic.

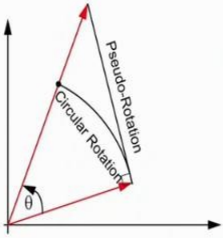
So, that is why we have put that y equals to 0 and x equals to 0.6073. If you put x and y , you can calculate $x \cos \theta$ minus $y \sin \theta$ $y \cos \theta$ plus $x \sin \theta$ so, but in the vectoring mode depending on the sign x and y , you can get at the at the; that means, from the first output you can get square root of x^2 plus y^2 and in this particular it will be \tan^{-1} of y divided by x . So, z initially you have put we have put here z initially we have put initially as 0 means 0 that is \tan^{-1} of y by x and here what we have done? That is y equals to 2 and x equals to 1 right.

So, at this particular point, I will get what; I will get K square root of 4 plus 1 that is 5. So, square root of 5 K I will get at that particular output; that means, the circular coordinate or the circular cordic if I run in the rotation mode I can calculate $\sin \theta$ $\cos \theta$, in the vectoring mode I can compute this \tan^{-1} of θ and then \tan^{-1} of x by y sorry y by x as well as I can also compute the square root of x^2 plus y^2 square ok. So; that means, that 2 function sorry not this function if I require suppose for a particular function I require square root operation. So, using this method I can calculate. So, square root of this means if I just put y equals to 0 means only x I will get, if I put x equals to 0 only y I will get at that particular output ok.

So, is this that only; that means, circular mode only or this cordic can be used in the circular mode only; this is not the thing; we can compute different kind of arithmetic operation considering this rotation in this is circular rotation right. So, considering the fact on the other rotation other rotational method we can compute different function.


(Refer Slide Time: 17:25)

With rotations in a Circular Coordinate System we are limited to the number of functions that can be calculated.



However, we shall see that by considering rotations in other coordinate systems we can calculate more functions directly, such as multiplications and divides, which allow us to calculate even more functions indirectly.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES



So, what are those? So, this is the circular rotation and this is the pseudo rotation we are making.

(Refer Slide Time: 17:32)

The slide is titled "Other Coordinate Systems". It contains two bullet points:

- Linear Coordinate System: Accompanied by a diagram showing a vector in the first quadrant of a Cartesian coordinate system. A smaller vector is shown inside it, and an arc indicates the angle between them, labeled "Linear Rotations".
- Hyperbolic Coordinate System: Accompanied by a diagram showing a vector in the first quadrant. A hyperbolic curve is drawn in the first quadrant, and a smaller vector is shown along this curve, labeled "Hyperbolic Rotations".

The slide footer includes the IIT Kharagpur logo and the text "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a man in a red shirt is visible in the bottom right corner.

But other coordinate system is that this linear coordinate system ok. So, linear coordinate system means, it is from here and then from it is there. And then hyperbolic rotation means the corresponding rotation that is follows this hyperbolic curve in the linear rotation it follows this linear; that means, this the change in this magnitude or in this vectors, that are in the it follows the linear route. But here if it is follows this hyperbolic rotation. So, at the time this will be hyperbolic coordinate system ok.

(Refer Slide Time: 18:17)

The slide contains the following text:

The advantage of using other coordinate systems with the CORDIC algorithm is that it allows more functions to be calculated. The drawback is that the system becomes more complex. The set of rotation angles used for the Circular Coordinate System are no longer valid when using the CORDIC algorithm with a Linear or Hyperbolic system. Hence, two other sets of angles are used for rotations made in these systems.

We shall see shortly that the CORDIC equations can be generalised for the 3 coordinate systems and that this involves the introduction of two new variables to the equations. One of these new variables ($e^{i\theta}$) represents the set of angles used to represent rotations in the appropriate coordinate system.

The slide footer includes the IIT Kharagpur logo and the text "NPTEL ONLINE CERTIFICATION COURSES". A small video inset of a man in a red shirt is visible in the bottom right corner.

So, the advantage of using this other coordinate system with the cordic algorithm is that it allows more function to be calculated, but the drawback is that the systems became

more complex. That means the circular rotation mode in the cordic that is the simpler one. The using the cordic algorithm with a linear or hyperbolic system, because circular cordic mode the pseudo rotation method is only applied for the circular coordinate system that is the circular rotation mode. It is not useful for the; or that will not be used for the linear system or this linear that coordinate system or hyperbolic coordinate system. So, hence 2 other set of angles are used for the rotations made in this systems.

(Refer Slide Time: 19:13)

Generalized CORDIC Equations

- With the addition of two other Coordinate Systems the CORDIC equations can now be generalised to accommodate all three systems:

$$x^{(i+1)} = (x^{(i)} - \mu d_i (2^{-i} y^{(i)}))$$

$$y^{(i+1)} = (y^{(i)} + d_i (2^{-i} x^{(i)}))$$

$$z^{(i+1)} = z^{(i)} - d_i e^{(i)}$$

- Circular Rotations: $\mu = 1, e^{(i)} = \tan^{-1} 2^{-i}$
- Linear Rotations: $\mu = 0, e^{(i)} = 2^{-i}$
- Hyperbolic Rotations: $\mu = -1, e^{(i)} = \tanh^{-1} 2^{-i}$

The slide also features the IIT Kharagpur and NPTEL Online Certification Courses logos at the bottom, along with a small video inset of a presenter.

So, we will see that shortly. So, that if we consider this particular equation as the base equation. So, circular in the circular; that means, here what we have considered that another term over here which is mu. So, mu is basically changed for the circular rotation mu is 1 means this equation remains same and e i equals to tan inverse of 2 to the power i; the pseudo rotation becomes tan inverse of 2 to the power i.

But in linear rotation this mu is 0 that means, only this y i and this z i value is changed. There will be no term for this and this e i value will be changed in the factor of 2 to the power minus i. But in hyperbolic rotation this mu is minus 1 that means, at the time this will be plus and this e i will be calculated as tan inverse of tan inverse h minus; that means, tan inverse hyperbolic minus inverse of 2 to the power minus i.

So; that means, at the time this elementary angle will be calculated based on this hyperbolic of tan; that means, tan hyperbolic tan hyperbolic of this 2 to the power minus i.

(Refer Slide Time: 20:45)

Summary of CORDIC Functions		
	Rotation Mode: $d = \text{sign}(z^{(i)}); z^{(i)} \rightarrow 0$	Vectoring Mode: $d = -\text{sign}(x^{(i)}y^{(i)}); y^{(i)} \rightarrow 0$
Circular $\mu = 1$ $e^{(0)} = \tan^{-1}2^{-i}$	$x \rightarrow \text{CORDIC} \rightarrow K(x \cdot \cos z - y \cdot \sin z)$ $y \rightarrow \text{CORDIC} \rightarrow K(y \cdot \cos z + x \cdot \sin z)$ $z \rightarrow \text{CORDIC} \rightarrow 0$ For $\cos z$ & $\sin z$, set $x = 1/K, y = 0$	$x \rightarrow \text{CORDIC} \rightarrow K(x^2 + y^2)^{1/2}$ $y \rightarrow \text{CORDIC} \rightarrow 0$ $z \rightarrow \text{CORDIC} \rightarrow z + \tan^{-1}(y/x)$ For $\tan^{-1} y$, set $x = 1, z = 0$
Linear $\mu = 0$ $e^{(i)} = 2^{-i}$	$x \rightarrow \text{CORDIC} \rightarrow x$ $y \rightarrow \text{CORDIC} \rightarrow y + (x \cdot z)$ $z \rightarrow \text{CORDIC} \rightarrow 0$ For multiplication, set $y = 0$	$x \rightarrow \text{CORDIC} \rightarrow x$ $y \rightarrow \text{CORDIC} \rightarrow 0$ $z \rightarrow \text{CORDIC} \rightarrow z + (y/x)$ For division, set $z = 0$
Hyperbolic $\mu = -1$ $e^{(i)} = \tanh^{-1}2^{-i}$	$x \rightarrow \text{CORDIC} \rightarrow K'(x \cdot \cosh z - y \cdot \sinh z)$ $y \rightarrow \text{CORDIC} \rightarrow K'(y \cdot \cosh z + x \cdot \sinh z)$ $z \rightarrow \text{CORDIC} \rightarrow 0$ For $\cosh z$ & $\sinh z$, set $x = 1/K', y = 0$	$x \rightarrow \text{CORDIC} \rightarrow K'(x^2 - y^2)^{1/2}$ $y \rightarrow \text{CORDIC} \rightarrow 0$ $z \rightarrow \text{CORDIC} \rightarrow z + \tanh^{-1}(y/x)$ For $\tanh^{-1} y$, set $x = 1, z = 0$

So, based on that so, we are getting three different equation and again this rotation vector whatever consideration we are taking for the rotation vector rotation mode and the vectoring mode, that will remain same; that means, the corresponding sign of d i that will be depended on this z for the that that rotation mode and for x i y i for the vectoring mode so, that will remain same.

So, then what are the outputs I will get in the circular mode we have already seen, in the linear mode x will be x. So, in the y output in the; that means, in the y output what I will get? Y plus x of z x into z; that means, the multiplication easily I can get; or the multiplicative result I can get using this running this cordic in the linear domain and along with that rotational mode. And I can get z plus y by x if I run that cordic in the vectoring mode; that means I can do multiplication as well as division if I run the cordic in the linear coordinate system.

If I run it for hyperbolic coordinate system so, at that time here what we are getting? We are getting cos, but here we will get the function as cos hyperbolic and the again that for the pseudo rotation here I will get the scale scaling factor. So, here scaling factor will be different because the summation of this here that is not cos theta here it will be cos hyperbolic theta, it will be omit right. So, depending on that cos hyperbolic theta. So, here this particular function the corresponding values of K that will be changed, and again whatever is the value of here. So, here we are getting tan inverse of y by x.

So, here we will get tan inverse hyperbolic of y by x, and here in the in this particular x output what we are getting? There is square root of x square plus y square. So, here what

we will get? Here we will get x square minus y square root of x square minus y square. That means, running the cordic for 6 different mode, I can get different function to be implemented very easily, which is cos theta, sin theta, tan theta, square root then multiplication division hyperbolic function and then square root of here plus here minus. So, these are the function which we can compute using the cordic running on different combination.

(Refer Slide Time: 23:45)

When using the CORDIC algorithm for Hyperbolic rotations the scaling factor K is different from the one used for Circular rotations.

The Hyperbolic scaling factor is denoted K_n and is calculated using the equation:

$$K_n = \prod_{i=0}^{n-1} \left(\sqrt{1 - 2^{-2i}} \right)$$

$K_n \rightarrow 0.82816$ as $n \rightarrow \infty$
 $1/K_n \rightarrow 1.20750$ as $n \rightarrow \infty$
 $n = \text{number of iterations}$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES


So, for hyperbolic what is the scaling factor? For hyperbolic the scaling factor is 0.82816 so, $1/K_n$ will be 1.20750. So, in case of that circular rotation the K_n value was 1.647 something and $1/K_n$ was 0.6073, but here you see this value is 0.8 to the K_n value is the 0.82816 and $1/K_n$ is 1.20750; that means, the x will be given as 1.20750 to get the proper output at x and y ; that means, the sin theta original sin theta and cos theta can be sin hyperbolic theta and cos hyperbolic theta can be calculated, if I put x equals to this 1.20750 this value and where n is the number of iteration.

(Refer Slide Time: 24:45)


Other Functions

- Although the CORDIC algorithms can only compute a limited number of functions directly, there are many more that can be achieved indirectly:

$\tan z = \frac{\sin z}{\cos z}$	$\tan^{-1} w = \tan^{-1} \frac{\sqrt{1-w^2}}{w}$
$\tanh z = \frac{\sinh z}{\cosh z}$	$\sin^{-1} w = \tan^{-1} \frac{w}{\sqrt{1-w^2}}$
$\ln w = 2 \tanh^{-1} \left \frac{w-1}{w+1} \right $	$\cosh^{-1} w = \ln(w + \sqrt{1-w^2})$
$e^z = \sinh z + \cosh z$	$\sinh^{-1} w = \ln(w + \sqrt{1+w^2})$
$w^t = e^{t \ln w}$	$\sqrt{w} = \sqrt{(w+1/4)^2 - (w-1/4)^2}$



NPTEL ONLINE
CERTIFICATION COURSES



So, then again you can although this cordic algorithm can compute a limited number of function directly, but there are many more other can be achieved indirectly. So, $\tan z$ can be calculated divided using this $\sin z$ by $\cos z$; then \tan hyperbolic z you can calculate using \sin hyperbolic z by \cos hyperbolic z means what; cordic you run for to calculate this \sin and \cos , you run that for circular mode circular in the circular; that means, this circular rotation mode you are getting \sin and \cos .

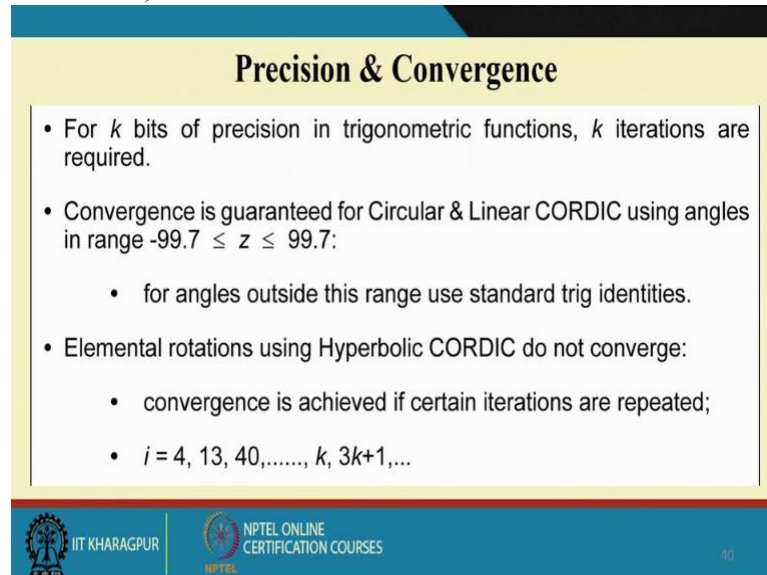
Then again if you just if you just do this division means what? You just put that value in this x and y if you put \sin and \cos . So, sorry in this mode if you put x and y ; that means, this \sin and \cos automatically you will get \tan inverse of this at this particular output. That means, linear vectoring mode you will get $\tan z$ same hyperbolic, then again linear vectoring mode you are you are getting \tan hyperbolic z . Then this \ln function that can be recalculated as what; \tan inverse of h minus \tan hyperbolic inverse of w minus 1 divided by w by plus 1.

So, \ln function can be computed how? So, in this particular mode what we are calculating \tan hyperbolic of y by x . So, y and x if I put as this w minus 1 and w plus 1 so easily I can get this \tan hyperbolic of sorry $\ln w$. So, $\ln w$ actually this will calculate not only that apart from that, we have to make that that output 1 bit left shift ok. So, to make it 2 of this so, after that easily you can calculate this \ln function then e to the power z function also you can calculate, e to the power z function is \sin hyperbolic z and \cos hyperbolic z than this ωt can be calculated then square root of w that also can be

calculated so; that means, you can implement different kind of function if you are using the cordic algorithm ok.

So, this is the beauty of cordic algorithm and that is why cordic algorithm has a fast application area for this.

(Refer Slide Time: 27:39)



Precision & Convergence

- For k bits of precision in trigonometric functions, k iterations are required.
- Convergence is guaranteed for Circular & Linear CORDIC using angles in range $-99.7 \leq z \leq 99.7$:
 - for angles outside this range use standard trig identities.
- Elemental rotations using Hyperbolic CORDIC do not converge:
 - convergence is achieved if certain iterations are repeated;
 - $i = 4, 13, 40, \dots, k, 3k+1, \dots$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | NPTEL | 40

So, then again what is the; that means, precision and convergence if we have to calculate. So, what is the precision? For k bits of precision in trigonometric function, k iteration are required ok. And this conversion convergence is guaranteed for circular and linear cordic using angle in the range of 0.997 to minus 0.997 to plus 0.997. Why plus 997 to minus 997, where from this particular terms comes? This particular terms basically comes from the elementary angle which we have considered. If you summed up all this particular; that means, elementary angle elementary angle was, 45, then; that means, then 26.6 then 14.1, then 17.1 these angle if you just summed up so, we will get the values of 99.7.

So, that is why this minus 99.7 to plus 99.7 any values of z can be calculated. So, now, then how can I compute this 180 degree or; that means, more than this? We know the trigonometric property of the sin, cos or tan or this you know the trigonometric property. So, from that we can basically change the initial angle to another value, which will be within this range.

So, if you change and if you just put to your system so, automatically you make the corresponding arrangement; that means if there is a sign change or if the sin became cos or cos we can sin. So, at the time that; that means, that modification you have to do in the input side and the output side, but you can calculate any of the values, which is basically

covering the within the range that is minus 99.7 to 99.7. So, for angles outside this range use standard trigonometric identities so, trigonometric identities we know.

So, you change that based on that and then you can easily compute that. Elementary rotation using hyperbolic cordic do not converge because hyperbolic function is totally different from the circular function; circular this cordic method. So, convergence is achieved you certain iteration are repeated that is if i equals to 4, 13, 40. So, $3k + 1$ if the; that means, if we choose that as the $3k + 1$; that means,. So, at the time there is a chance for this hyperbolic cordic to be converged ok.

(Refer Slide Time: 30:42)

FPGA Implementation

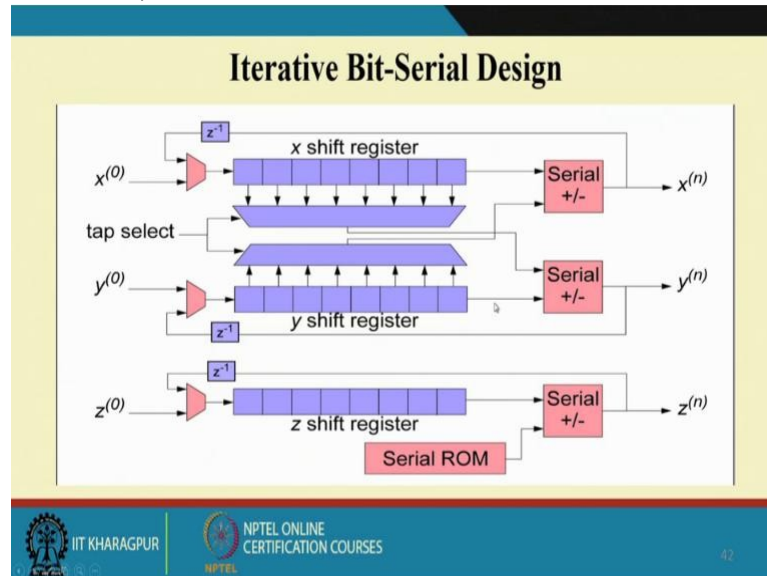
- The ideal CORDIC architecture depends on **speed vs area** tradeoffs in the intended application.
- A direct translation of the CORDIC equations is an iterative bit-parallel design, however:
 - bit-parallel variable shift shifters do not map well into FPGAs;
 - require several FPGA cells resulting in large, slow design.
- We shall consider an iterative bit-serial solution to illustrate:
 - a minimum area architecture;
 - one implementation of variable shift shifters.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, then for FPGA implementation of the cordic, it depends actually whenever we have seen that these 2 architecture which is the parallel one and which is the serial one. So, at that time you see there is a that tradeoff between speed and area right; in 1 that area may be higher, but speed wise if I use pipeline and so, at the time speed wise it will be improved right. So, in sequential or iterative the area wise it is minimized, but time wise it is time consuming, because throughput wise it is lesser than the n times lesser by n times ok.

So, then a direct translation of the cordic equation is an iterative bit parallel design; however, bit parallel variable shift register do not map well into FPGAs. So, this is on the aspect of FPGA implementation if you see. So, at the time this problems will come.

(Refer Slide Time: 31:46)



But it depends on that means, your consideration that what type of architecture you will choose what is the specification you have to make first you have to consider or concentrate on that particular fact.

So, to meet that particular constant, how you will run or how you will choose your architecture; whether you will choose bit serial architecture; that means, 1 bit by 1 bit processing you require or all the bits together you require all that means, parallel architecture you require, or the serial architecture you require. So, it depends upon your corresponding specification and specification is totally different for different different application ok. So, depending on your particular work, you have to be choose the 1 which particular that which architecture you will choose for the implementation of your hardware.

So, here means in this particular in this course what I am trying to give you just the overview of the architectures it is not that only this um; that means, whatever architecture I am I have just show 2 or 3 architecture. So, this that only this architecture for cordic is there it is not the fact. There are so, many architectures are available for implementing the cordic. This is the basic concept of cordic now you can modify your architecture in order to improve the performance of the cordic.

So, how to improve or what will be the idea that is totally depended on you. So, for that reason if you just want more; that means more information or if you just; that means, more you are looking more for this particular area; that means, only the cordic architecture. So, at the time you just Google it or you also can just let me know that I

need more information I am I am need to be concentrate on this particular cordic architecture.

So, at the time I can also share the available literature with you or the idea is also if you are having it, then you can also discuss with me. So, with that thank you for today, then again you will try to see 1 or 2 of the architectures new architectures how people have developed and what are the advantage they are getting so, that we will see in the next class.

Just to give the overview of this is that this is not the only architecture because this course is all about architectural design of ICs ok. So, what are the other architecture and how people have developed step by step so, that we will take as an example and then that we will see in the next class. So

Thank you for today's class.