CORDIC Background (2A)

-
-

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

CORDIC Background

1.A survey of CORDIC algorithms for FPGAs, Ray Andraka,
www.andraka.com/cordic.htm

Vector Rotation (1)

$$x' = x\cos\phi - y\sin\phi$$
$$y' = y\cos\phi + x\sin\phi$$

$$x' = \cos\phi \cdot [x - y\tan\phi]$$
$$y' = \cos\phi \cdot [y + x\tan\phi]$$

$$x_{i+1} = K_i \cdot [x_i - y_i \cdot d_i \cdot 2^{-i}]$$
$$y_{i+1} = K_i \cdot [y_i + x_i \cdot d_i \cdot 2^{-i}]$$

$$K_i = \cos\phi_i = \cos(\tan^{-1}(2^{-i}))$$
$$= \frac{1}{\sqrt{1 + 2^{-2i}}}$$
$$d_i = \pm 1$$

Restrict rotation angle ➡ $\tan\phi = \pm 2^{-i}$

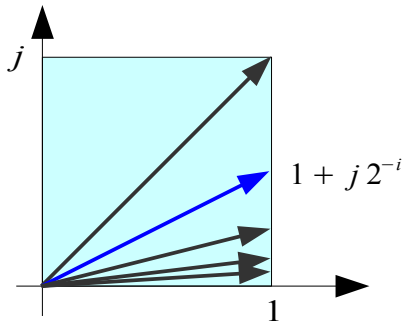
Multiplication ➡ simple shift

$y \cdot \tan\phi$ $\quad y \cdot 2^{-i}$

$x \cdot \tan\phi$ $\quad x \cdot 2^{-i}$

regardless of direction ➡ $\cos(\phi) = \cos(-\phi)$

Allowed Rotation Angles



$\tan\phi$ ➡ 2^{-i}

$\cos\phi$ ➡ $\frac{1}{\sqrt{1 + 2^{-2i}}}$

$K_i \leq 1$ $\quad (\ K_i \Leftarrow \cos\phi\ )$

Vector Rotation (2)

$$x_{i+1} = K_i \cdot \left[x_i - y_i \cdot d_i \cdot 2^{-i} \right]$$
$$y_{i+1} = K_i \cdot \left[y_i + x_i \cdot d_i \cdot 2^{-i} \right]$$

$$K_i = \cos \phi_i = \cos\left(\tan^{-1} \left(2^{-i} \right) \right)$$
$$= \frac{1}{\sqrt{1 + 2^{-2i}}} \qquad K_i \le 1$$

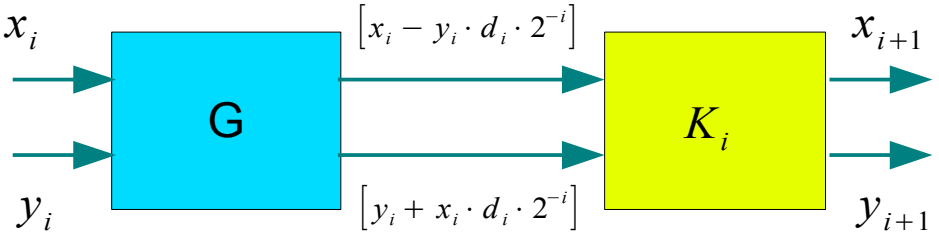
$$d_i = \pm 1$$

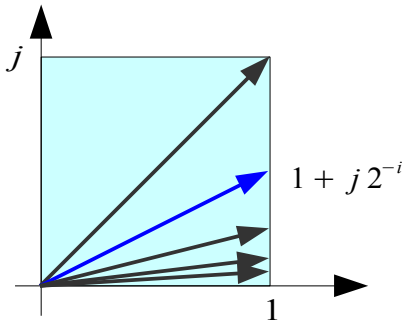
$$x_{i+1}^2 = K_i^2 \cdot \left[x_i^2 + y_i^2 \cdot 2^{-2i} - 2 x_i y_i d_i \cdot 2^{-i} \right]$$
$$y_{i+1}^2 = K_i^2 \cdot \left[y_i^2 + x_i^2 \cdot 2^{-2i} + 2 x_i y_i d_i \cdot 2^{-i} \right]$$

$$x_{i+1}^2 + y_{i+1}^2 = K_i^2 \cdot \left(1 + 2^{-2i}\right) \cdot \left(x_i^2 + y_i^2 \right)$$

$K_i \le 1$ G > 1

$$G \cdot K_i = 1$$





$\tan \phi \Rightarrow 2^{-i}$

$\cos \phi \Rightarrow \frac{1}{\sqrt{1 + 2^{-2i}}}$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^{n} \frac{1}{K_i} = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

Vector Rotation (3)

$$x_{i+1} = K_i \cdot \left[x_i - y_i \cdot d_i \cdot 2^{-i} \right]$$
$$y_{i+1} = K_i \cdot \left[y_i + x_i \cdot d_i \cdot 2^{-i} \right]$$

$$K_i = 1 / \sqrt{1 + 2^{-2i}} \quad \Leftarrow \quad \cos(\phi_i)$$
$$d_i = \pm 1$$

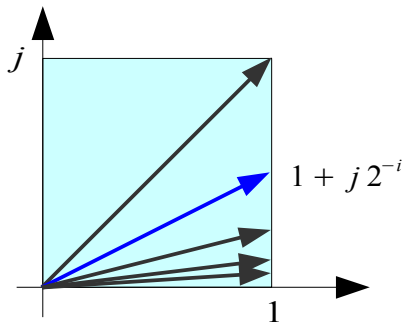
Without Scale Constants K_i

$$x_{i+1} = \left[x_i - y_i \cdot d_i \cdot 2^{-i} \right]$$
$$y_{i+1} = \left[y_i + x_i \cdot d_i \cdot 2^{-i} \right]$$
$$d_i = \pm 1$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^{n} \frac{1}{K_i} = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

$$1 / K_i = \sqrt{1 + 2^{-2i}}$$



$$\tan\phi \Rightarrow 2^{-i}$$
$$\cos\phi \Rightarrow \frac{1}{\sqrt{1 + 2^{-2i}}}$$

For correction

Multiplying K_i*'s as a processing gain*

$$\prod_{i=1}^{n} K_i = \prod_{i=1}^{n} \frac{1}{\sqrt{1 + 2^{-2i}}} \rightarrow 0.6073$$

Angle Accumulator

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

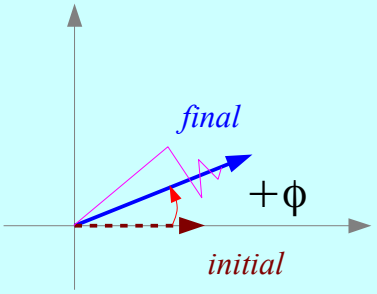
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

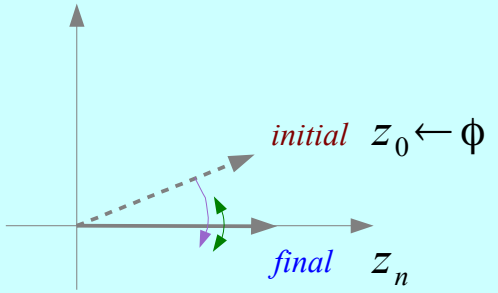
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



Subtract *angles at each step*

Vectoring Mode

$$z_0 \leftarrow 0$$

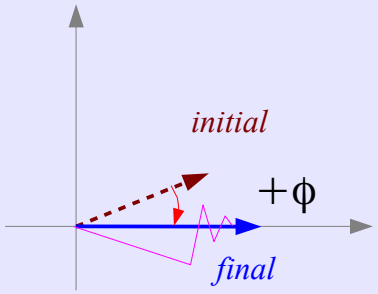
$$z_n \rightarrow z_0 + \tan^{-1}(y_0 / x_0)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

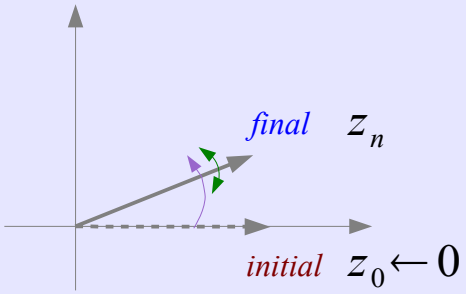
$$d_i = -1 \quad otherwise$$

Vector View



Minimize the residual y component

Accumulator View



Add *angles at each step*

Rotation Mode

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

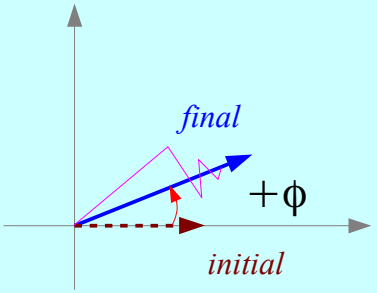
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

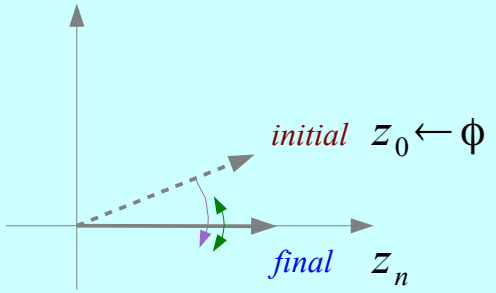
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



Subtract *angles at each step*

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

$$d_i = +1 \quad otherwise$$

$$x_n = A_n \left[x_0 \cos z_0 - y_0 \sin z_0 \right]$$

$$y_n = A_n \left[y_0 \cos z_0 + x_0 \sin z_0 \right]$$

$$z_n = 0$$

$$A_n = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}}$$

Vectoring Mode

Vectoring Mode

$$z_0 \leftarrow 0$$

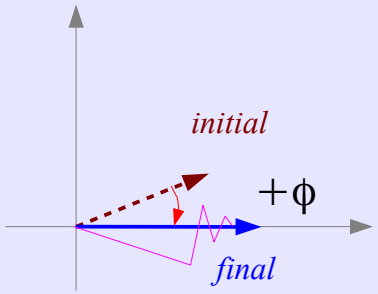
$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

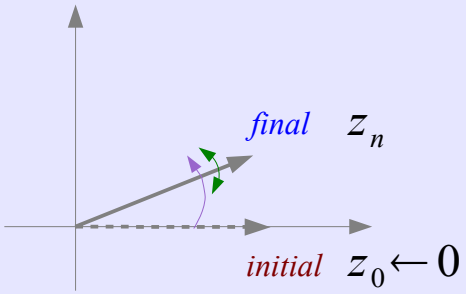
$$d_i = -1 \quad otherwise$$

Vector View



Minimize the residual y component

Accumulator View



Add *angles at each step*

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

$$d_i = -1 \quad otherwise$$

$$x_n = A_n \sqrt{x_0^2 + y_0^2}$$

$$y_n = 0$$

$$z_n = z_0 + \tan^{-1}(y_0/x_0)$$

$$A_n = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}}$$

Angle Accumulator – Rotation Mode

Rotation Mode

$$z_0 \leftarrow \phi \quad (input)$$

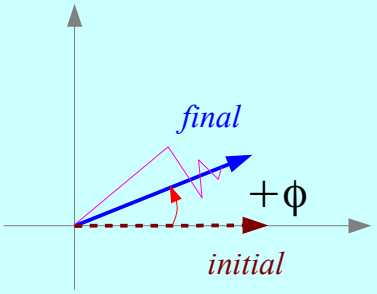
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

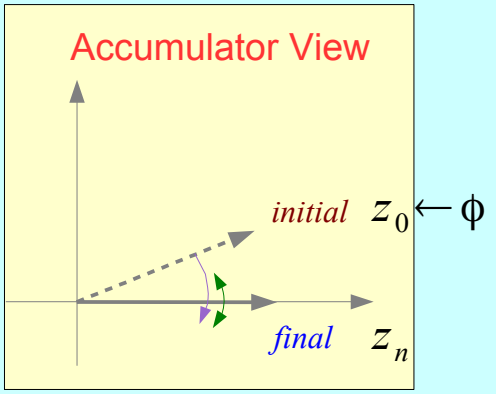
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



Subtract *angles at each step*

$z_i < 0$

Increase Angle $d_i = -1$

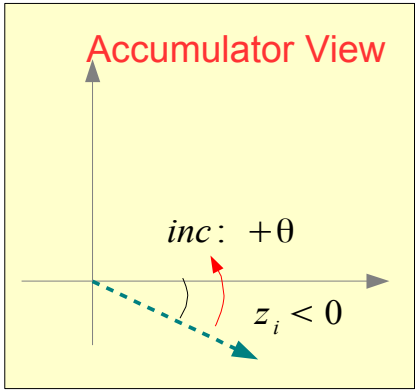
$$z_{i+1} = z_i + \tan^{-1}(2^{-i})$$

$z_i \geq 0$

Decrease Angle $d_i = +1$

$$z_{i+1} = z_i - \tan^{-1}(2^{-i})$$

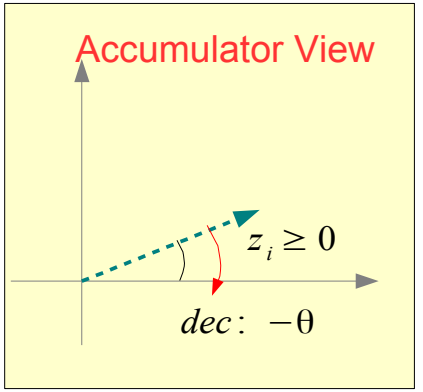
Accumulator View



$z_i < 0$

Increase Angle $+\theta$

Accumulator View



$z_i \geq 0$

Decreases Angle $-\theta$

Angle Accumulator – Vectoring Mode

Vectoring Mode

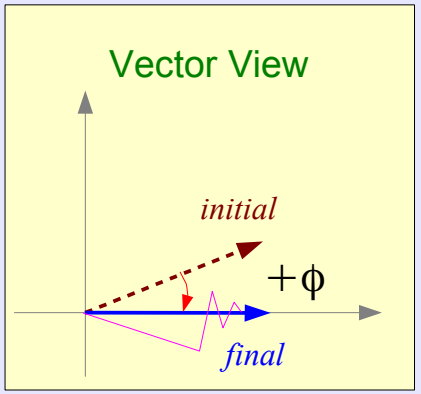
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

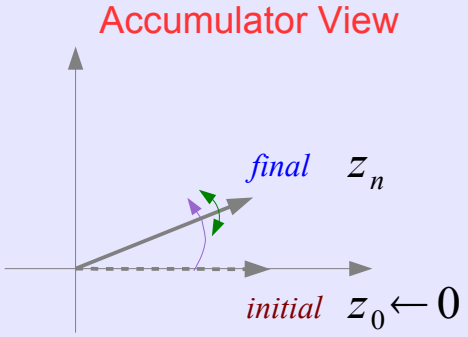
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

$$d_i = -1 \quad otherwise$$



Minimize the residual y component



Add *angles at each step*

$y_i < 0$

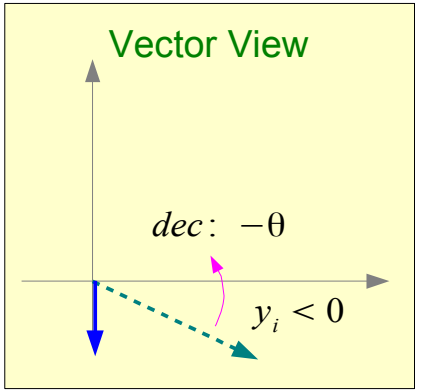
Decrease Angle $d_i = +1$

$$z_{i+1} = z_i - \tan^{-1}(2^{-i})$$

$y_i > 0$

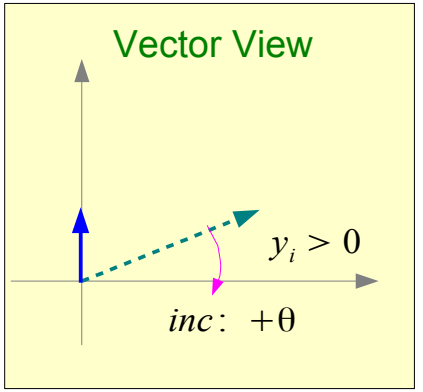
Increase Angle $d_i = -1$

$$z_{i+1} = z_i + \tan^{-1}(2^{-i})$$



$y_i < 0$

Decrease Angle $-\theta$

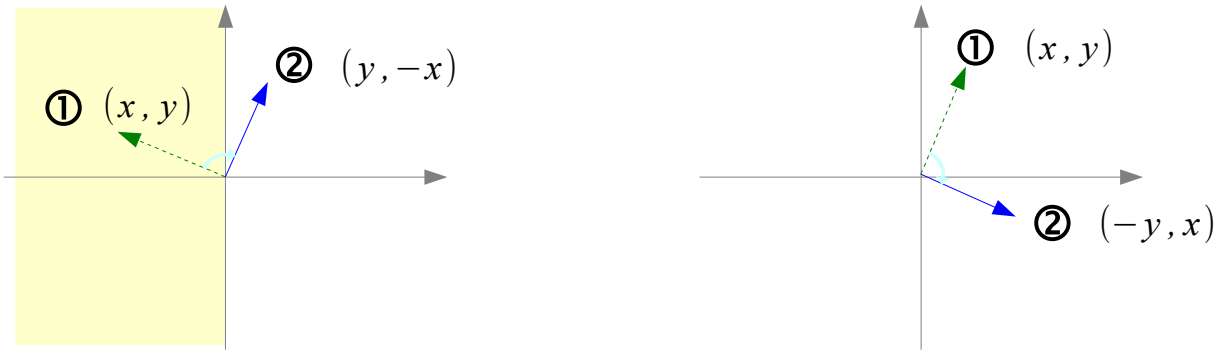


$y_i > 0$

Increases Angle $+\theta$

Initial Rotation $\pm\pi/2$

Positive Phase (y > 0) ➡ *Rotate by –90 degrees*



Negative Phase (y < 0) ➡ *Rotate by +90 degrees*

Resulting Phase ➡ [-90, +90]

$$x' = -d \cdot y$$
$$y' = +d \cdot x$$
$$z' = z + d \cdot \frac{\pi}{2}$$

$$d = +1 \quad if \quad y < 0$$
$$d = -1 \quad otherwise$$

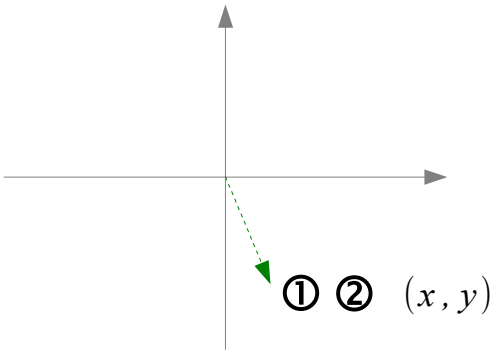
No magnitude change

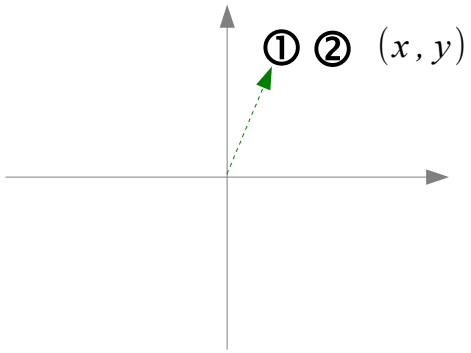
$$x' \leftarrow y$$
$$y' \leftarrow x$$

Consistent

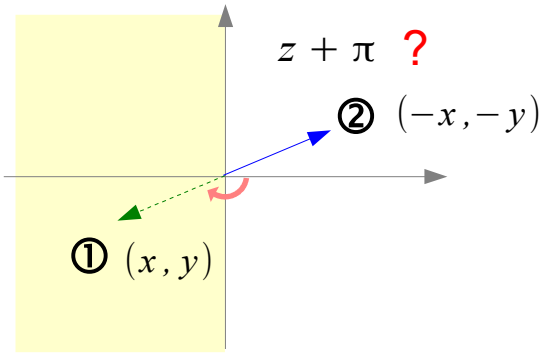
Initial Rotation $0, +\pi$

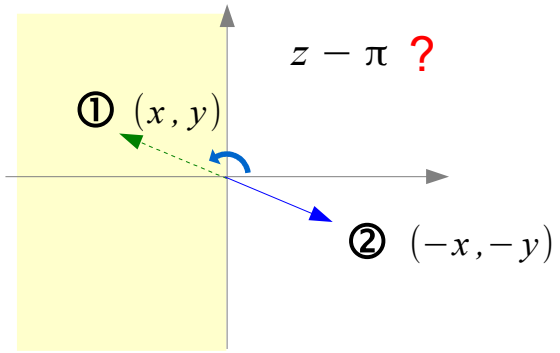
Positive x (x > 0) ➡ *Rotate by* *0 degrees*





Negative x (x < 0) ➡ *Rotate by +180 degrees*





Resulting Phase ➡ [-90, +90]

$$x' = +d \cdot x$$
$$y' = +d \cdot y$$
$$z' = z \quad if \quad d = 1$$
$$z' = \pi - z \quad if \quad d = -1$$

$$d = -1 \quad if \quad x < 0$$
$$d = +1 \quad otherwise$$

No magnitude change

$$x' \leftarrow y$$
$$y' \leftarrow x$$

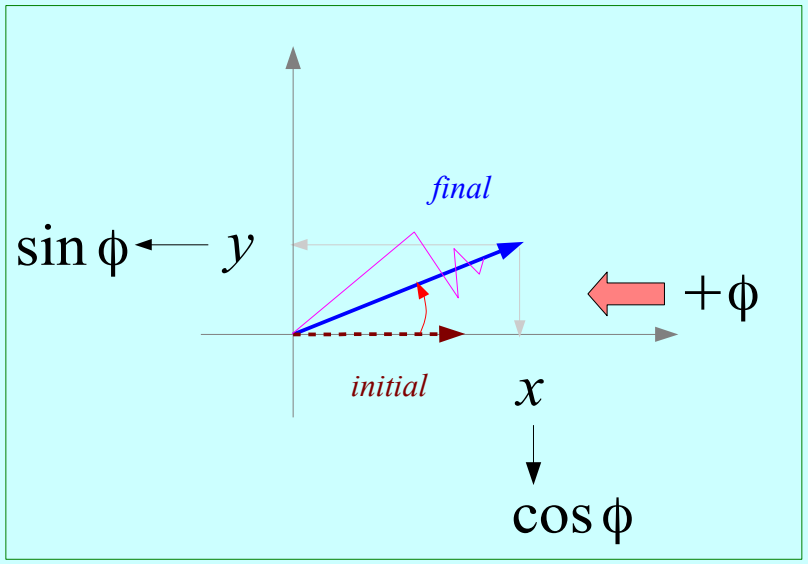
Convenient wiring in FPGA

Application Modes (1)

Rotation Mode

Input angle is given

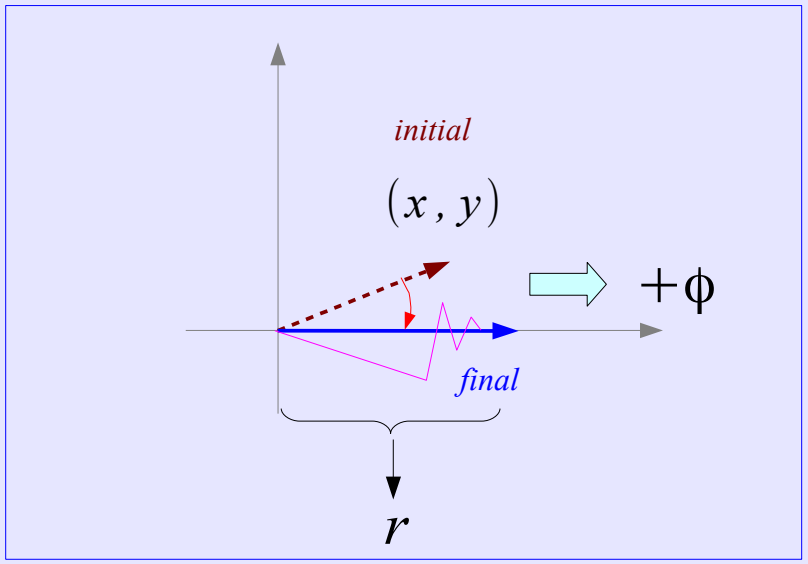
- **sin** and **cos**
- $(r, \theta) \rightarrow (x, y)$
- General vector rotation



Vectoring Mode

Finding the resulting angle

- **$\tan^{-1}$**
- Vector Magnitude
- $(x, y) \rightarrow (r, \theta)$
- **$\sin^{-1}$** and **$\cos^{-1}$**



Application Modes (2)

- Inverse CORDIC functions
- Extension to Linear functions
- Extension to Hyperbolic functions

A. Sine and Cosine

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

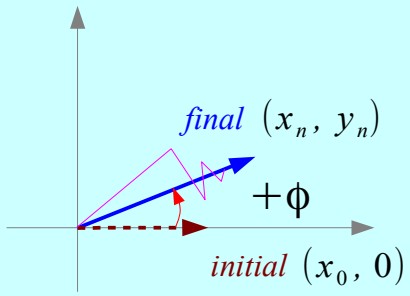
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

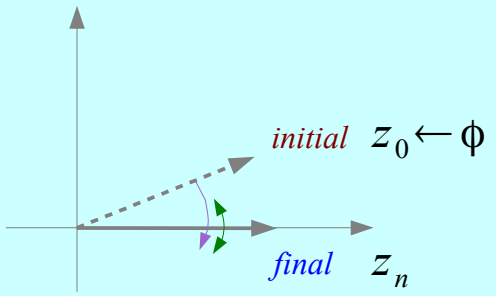
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



***Subtract** angles at each step*

Finding Sine and Cosine

$$(x_0,\ 0) \rightarrow (x_n,\ y_n)$$

$$x_n = A_n \cdot x_0 \cos z_0$$

$$y_n = A_n \cdot x_0 \sin z_0$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^{n} \frac{1}{K_i} = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

Unscaled Sine and Cosine

$$x_0 \leftarrow \frac{1}{A_n} = 0.6073$$

$$x_n = \cos z_0$$

$$y_n = \sin z_0$$

Modulated Sine and Cosine

$$x_0 \leftarrow \{\prod_{i=1}^{n} K_i\} \cdot x_0 = 0.6073 \cdot x_0$$

$$x_n = x_0 \cdot \cos z_0$$

$$y_n = x_0 \cdot \sin z_0$$

LUT → a pair of MULT

CORDIC → rotation operations

Single MULT

B. Polar to Rectangular

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

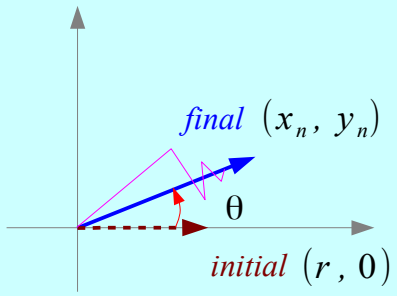
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

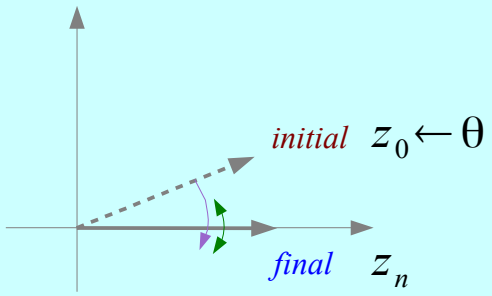
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



Subtract *angles at each step*

Finding Sine and Cosine

$$(x_0,\ 0) \rightarrow (x_n,\ y_n)$$

$$x_n = A_n \cdot x_0 \cos z_0$$

$$y_n = A_n \cdot x_0 \sin z_0$$

$$x_0 \leftarrow r, \quad z_0 \leftarrow \theta$$

$$(r,\ 0) \rightarrow (x_n,\ y_n)$$

$$x_n = A_n\, r \cos\theta$$

$$y_n = A_n\, r \sin\theta$$

$$x_0 \leftarrow r \cdot \frac{1}{A_n}, \quad z_0 \leftarrow \theta$$

$$\left(\frac{r}{A_n},\ 0\right) \rightarrow (x_n,\ y_n)$$

$$x_n = r \cos\theta$$

$$y_n = r \sin\theta$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^{n} \frac{1}{K_i} = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

C. General Vector Rotation

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

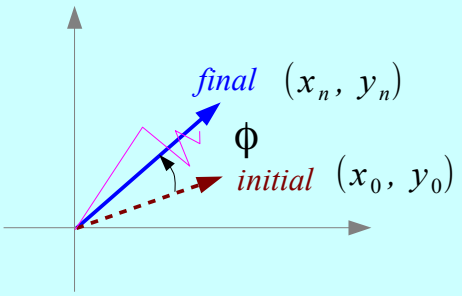
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad if \quad z_i < 0$$

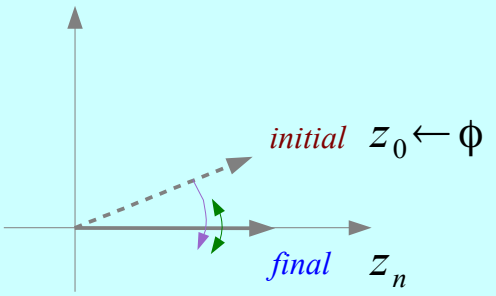
$$d_i = +1 \quad otherwise$$

Vector View



Minimize the residual angle

Accumulator View



***Subtract** angles at each step*

Motion Correction and Control System

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = A_n \cdot \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

* n-dim rotation
→ tree architecture

Unscaled Rotation

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = A_n \cdot \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} \frac{x_0}{A_n} \\ \frac{y_0}{A_n} \end{bmatrix}$$

A pair of MULT

$$\Rightarrow \quad \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

D. Arctangent

Vectoring Mode

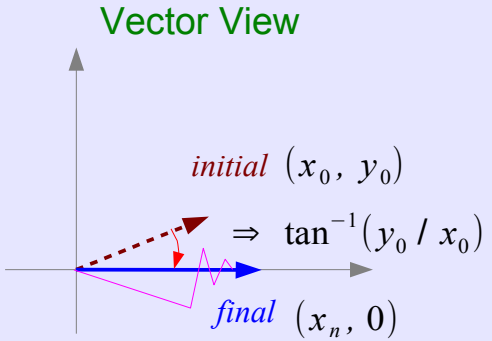
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

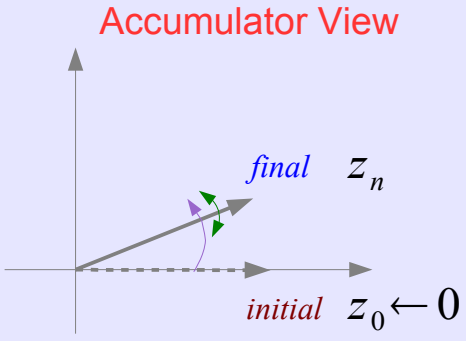
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

$$d_i = -1 \quad otherwise$$



Minimize the residual y component



***Add** angles at each step*

Input

$$(x_0, y_0) \rightarrow ratio \quad \frac{y_0}{x_0}$$

$$(0, y_0) \rightarrow ratio \quad \pm\infty$$

Output

Angle Accumulator Value

→ CORDIC gain does not affect

$$x_n = z_0 + \tan^{-1}\left(y_0 / x_0\right)$$

E. Vector Magnitude

Vectoring Mode

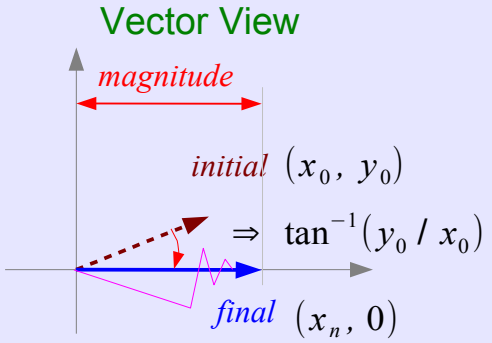
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0 / x_0)$$

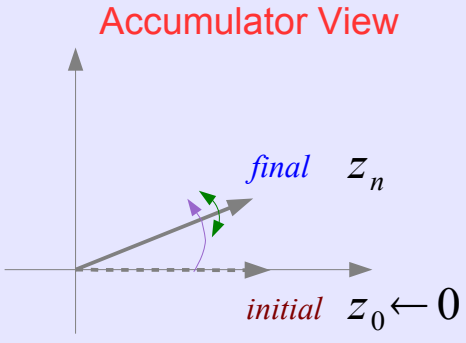
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

$$d_i = -1 \quad otherwise$$



Minimize the residual y component



Add *angles at each step*

The magnitude:

- *byproduct of computing arctangent*
- *the result vector is aligned with x-axis*
- *the x component of the result vector*
- *increased by CORDIC gain*
- *can be scaled by the processor gain*
- *one MULT hardware cost*

$$x_n = A_n \sqrt{x_0^2 + y_0^2}$$

The accuracy of the magnitude result

- *Improves by 2 bits for each iteration performed*

F. Cartesian to Polar Transformation

Vectoring Mode

$$z_0 \leftarrow 0$$

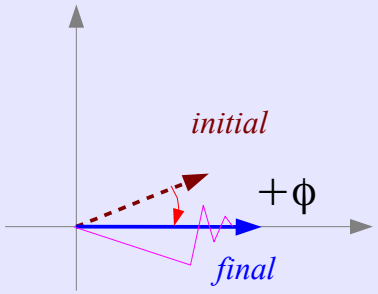
$$z_n \rightarrow z_0 + \tan^{-1}(y_0 / x_0)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < 0$$

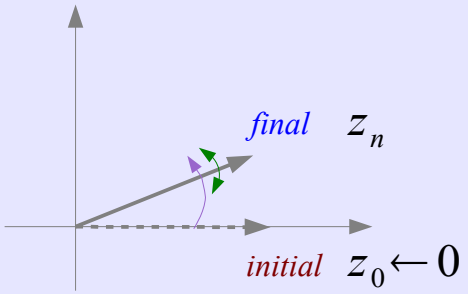
$$d_i = -1 \quad otherwise$$

Vector View



Minimize the residual y component

Accumulator View



Add *angles at each step*

input vector (x, y)

magnitude $r = \sqrt{x^2 + y^2}$ ➡ $x_n = A_n \sqrt{x_0^2 + y_0^2}$

phase angle $\phi = \tan^{-1}(y / x)$ ➡ $z_n = z_0 + \tan^{-1}(y_0 / x_0)$

G. Arcsine and Arccosine

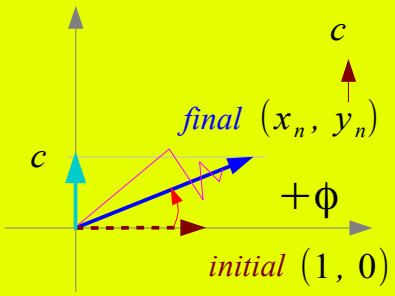
Rotation Mode

$$z_n \rightarrow z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

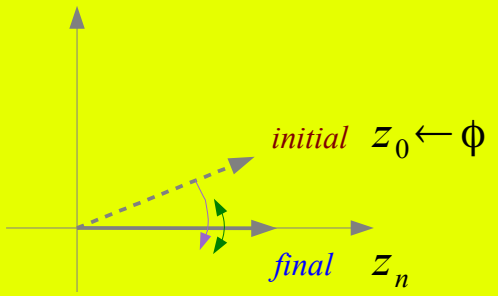
$$d_i = +1 \quad if \quad y_i < c$$
$$d_i = -1 \quad otherwise$$

Vector View



*Until the **y** comp is equal to the input **c***

Accumulator View



***Subtract** angles at each step*

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$
$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < c$$
$$d_i = -1 \quad otherwise$$

$$x_n = \sqrt{(A_n \cdot x_0)^2 - c^2}$$
$$y_n = c$$
$$z_n = z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$
$$A_n = \prod_{i=1}^{n} \sqrt{1 + 2^{-2i}}$$

G. Arccosine

Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$

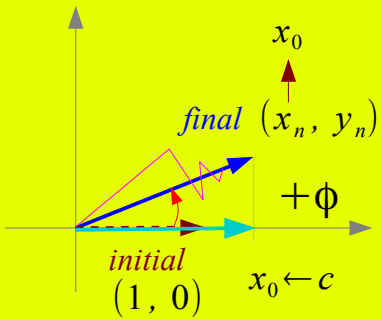
$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad if \quad y_i < c$$

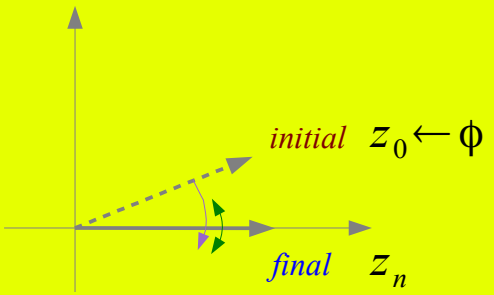
$$d_i = -1 \quad otherwise$$

Vector View



*Until the **y** comp is equal to the input **c***

Accumulator View



***Subtract** angles at each step*

$$input\ vector \quad (x, y)$$

$$magnitude \quad r = \sqrt{x^2 + y^2} \quad \Rightarrow \quad x_n = A_n \sqrt{x_0^2 + y_0^2}$$

$$phase\ \ angle \quad \phi = \tan^{-1}(y / x) \quad \Rightarrow \quad z_n = z_0 + \tan^{-1}(y_0 / x_0)$$

References

[1] http://en.wikipedia.org/
[2] CORDIC FAQ, www.dspguru.com