CORDIC Background (2A)

Young Won Lim
03/09/2011

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)

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

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

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

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

$$d_i = \pm 1$$

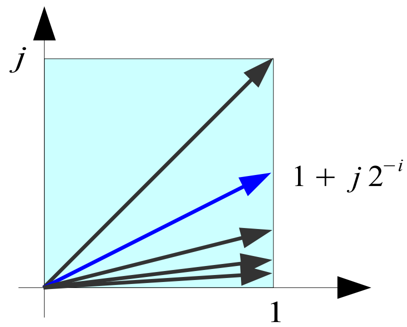
Restrict rotation angle ➡ $\tan\phi = \pm 2^{-i}$

Multiplication ➡ simple shift

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

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

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



$\tan\phi$ ➡ 2^{-i}

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

Vector Rotation (2)

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

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

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

$$d_i = \pm 1$$

Removing Scale Constants K_i

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

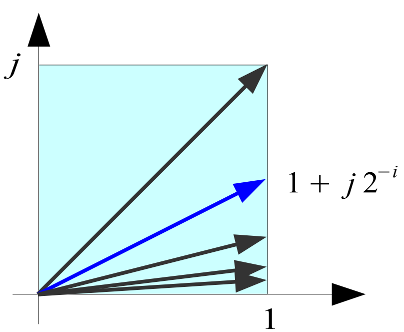
$$y_{i+1} = \left[y_i + x_i \cdot d_i \cdot 2^{-i}\right]$$

$$d_i = \pm 1$$

CORDIC Gain : *growing in magnitude*

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

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



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

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

For correction

Multiplying *K_i's as a processing gain*

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

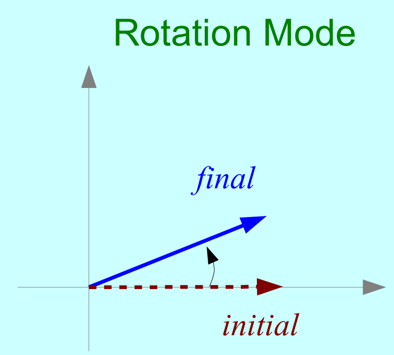
Angle Accumulator

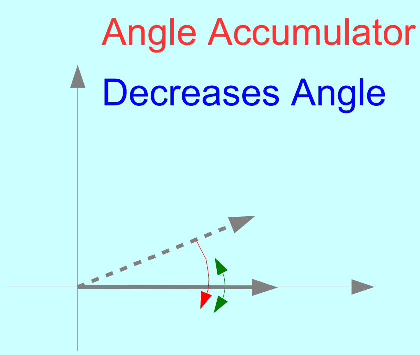
Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$
$$z_n \rightarrow 0$$

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

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



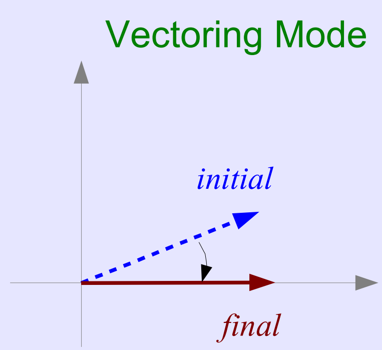


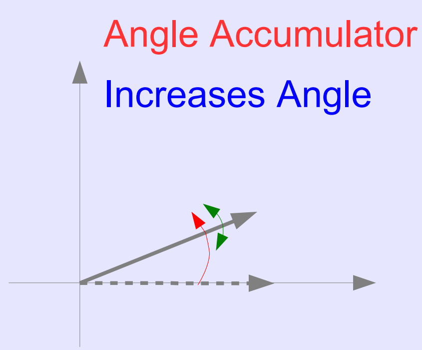
Vectoring Mode

$$z_0 \leftarrow 0$$
$$z_n \rightarrow z_0 + \tan^{-1}(y_0 / x_0)$$

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

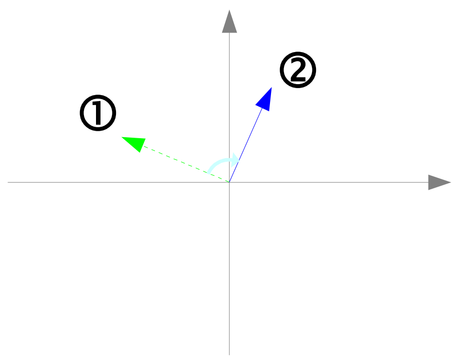
$$d_i = +1 \quad if \quad y_i < 0$$
$$d_i = -1 \quad otherwise$$

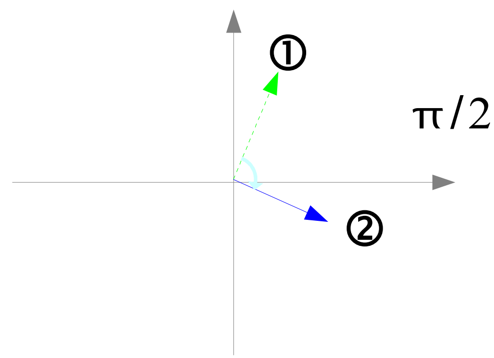




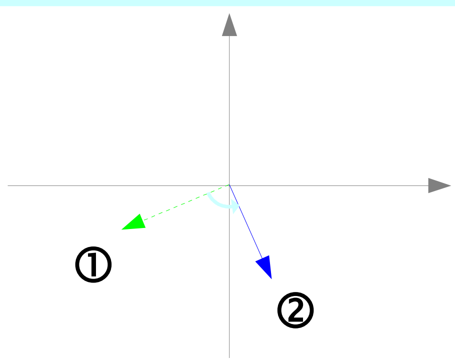
Initial Rotation $\pm\pi/2$

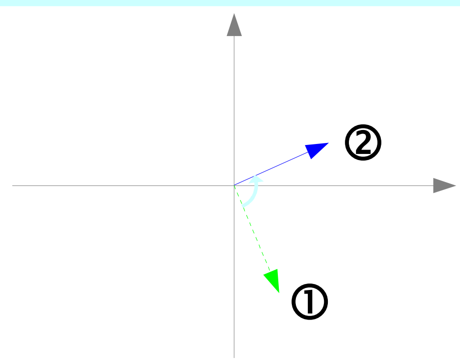
Positive Phase (y > 0) ➡ *Rotate by –90 degrees*





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





Resulting Phase ➡ [-90, +90]

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

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

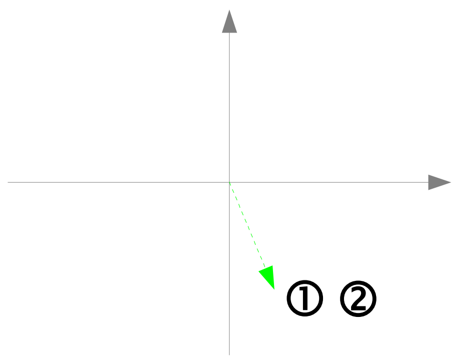
No magnitude change

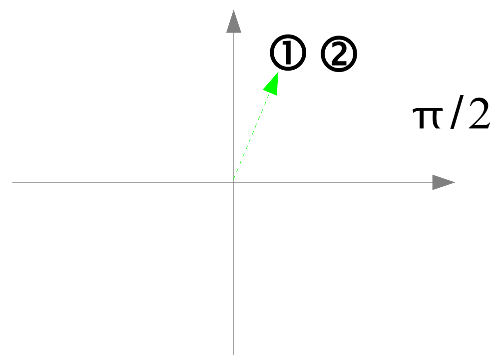
$$x' \leftarrow y$$
$$y' \leftarrow x$$

Consistent

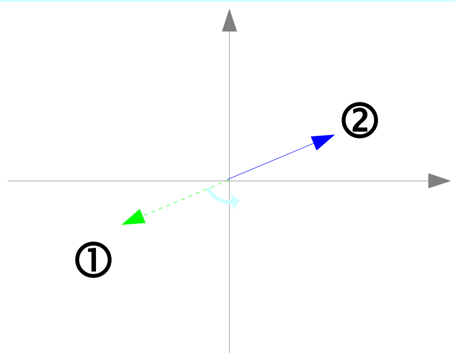
Initial Rotation $0, \ +\pi$

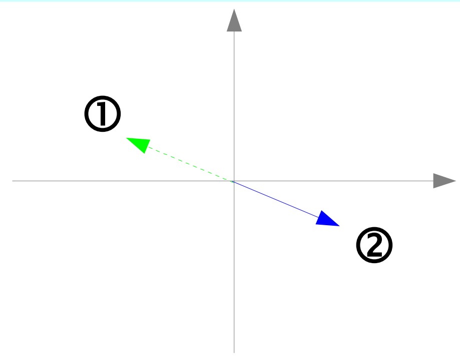
Positive x (x > 0) ➡ *Rotate by –90 degrees*





Negative x (x > 0) ➡ *Rotate by +90 degrees*





Resulting Phase ➡ [-90, +90]

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

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

No magnitude change

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

Convenient wiring in FPGA

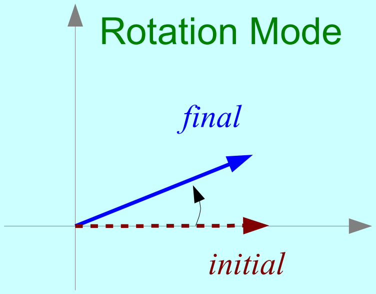
A. Sine and Cosine (1)

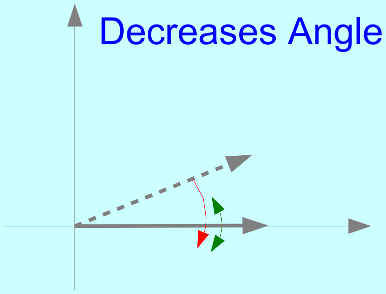
Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$
$$z_n \rightarrow 0$$

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

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

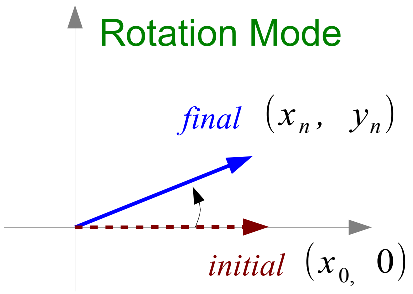


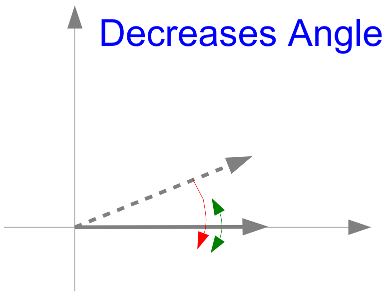


Finding Sine and Cosine

$$(x_{0,} \ 0) \rightarrow (x_n, \ y_n)$$

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





A. Sine and Cosine (2)

Finding Sine and Cosine

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

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

Unscaled Sine and Cosine

$$if \quad x_0 \leftarrow 1/A_n$$

$$x_n = \cos z_0$$
$$y_n = \sin z_0$$

Modulated Sine and Cosine

$$K_n \cdot x_n = K_n \cdot A_n \cdot x_0 \cos z_0$$
$$K_n \cdot y_n = K_n \cdot A_n \cdot x_0 \sin z_0$$

$$K_n \cdot x_n = x_0 \cos z_0$$
$$K_n \cdot y_n = x_0 \sin z_0$$

Look Up Table
→ a pair of MULT

CORDIC method
→ MULT as a part of rotation

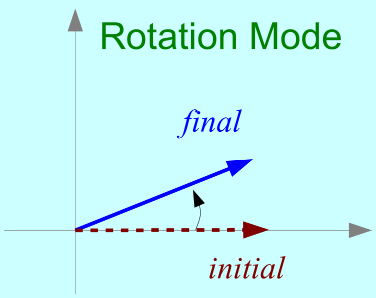
B. Polar to Rectangular

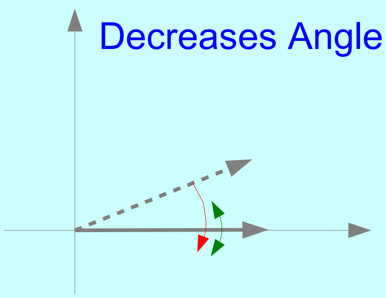
Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$
$$z_n \rightarrow 0$$

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

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

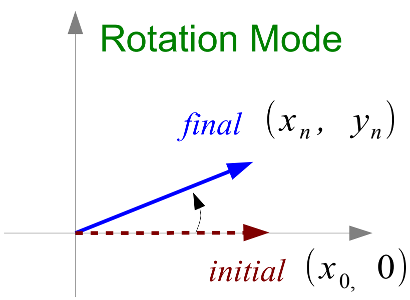


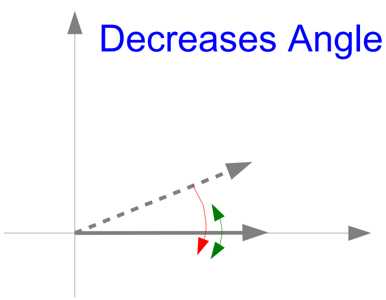


Finding Sine and Cosine

$$(x_{0,}\ 0) \rightarrow (x_n,\ y_n)$$

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





$$x_0 \leftarrow r \qquad z_0 \leftarrow \theta$$

$$x_n \rightarrow r\cos\theta \qquad y_n \rightarrow r\sin\theta$$

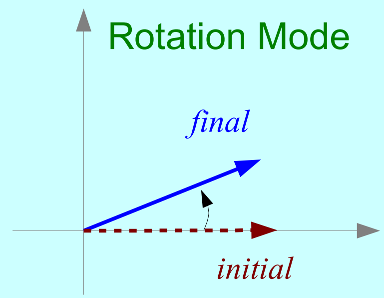
B. Polar to Rectangular

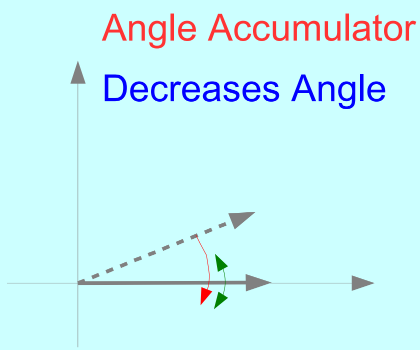
Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$
$$z_n \rightarrow 0$$

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

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

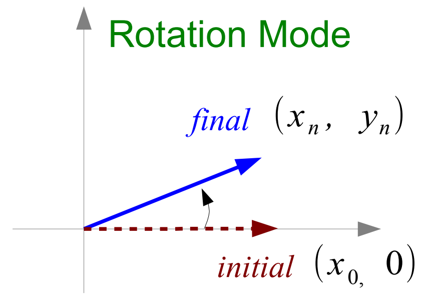


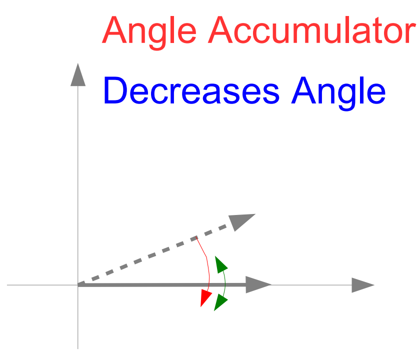


Finding Sine and Cosine

$$(x_{0,} \ \ 0) \rightarrow (x_n, \ \ y_n)$$

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





$$x_0 \leftarrow r$$
$$z_0 \leftarrow \theta$$

$$x_n \rightarrow r \cos\theta$$
$$y_n \rightarrow r \sin\theta$$

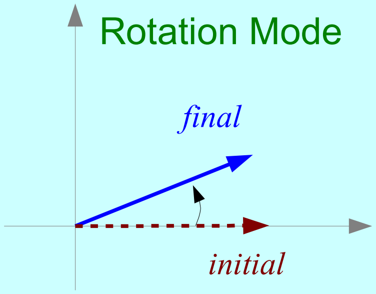
C. General Vector Rotation

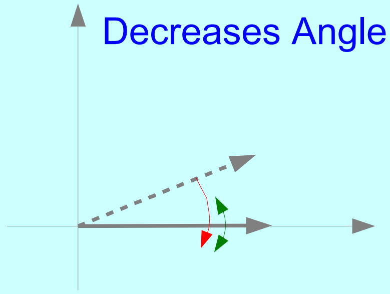
Rotation Mode

$$z_0 \leftarrow \phi \quad (desired\ angle)$$
$$z_n \rightarrow 0$$

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

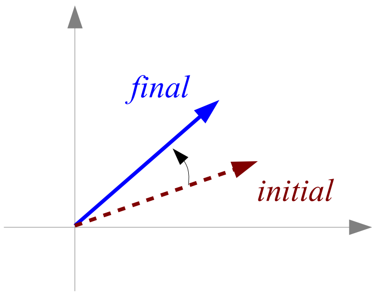
$$d_i = -1 \quad if \quad z_i < 0$$
$$d_i = +1 \quad otherwise$$





Motion Correction and Control System

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



C.

Vectoring Mode

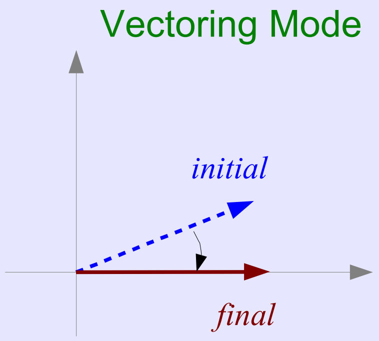
$$z_0 \leftarrow 0$$

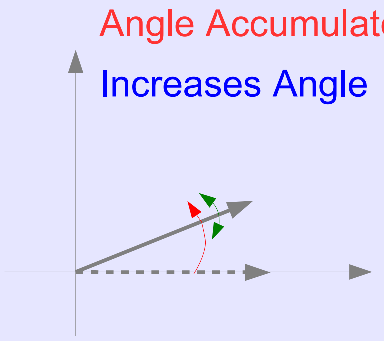
$$z_n \rightarrow z_0 + \tan^{-1}(y_0 / x_0)$$

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

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

$$d_i = -1 \quad otherwise$$





References

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