CORDIC

Coordinate Rotation Digital Computer

Sistemas Digitales (66.17)

FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Introducción

- Fue descripto por primera vez en 1959 por Jack E. Volder
- Desarrollado en el departamento de aeroelectrónica de Convair
- Más tarde John Stephen Walther, en Hewlett-Packard, generalizó el algoritmo
- Originalmente fue implementado usando el sistema binario. En los 70's su variante decimal se empezó a usar fuertemente en calculadoras de mano

Introducción

- Es utilizado sobre todo en dispositivos en los que no hay disponibilidad de multiplicadores, por ejemplo en microcontroladores y FPGA's pequeñas

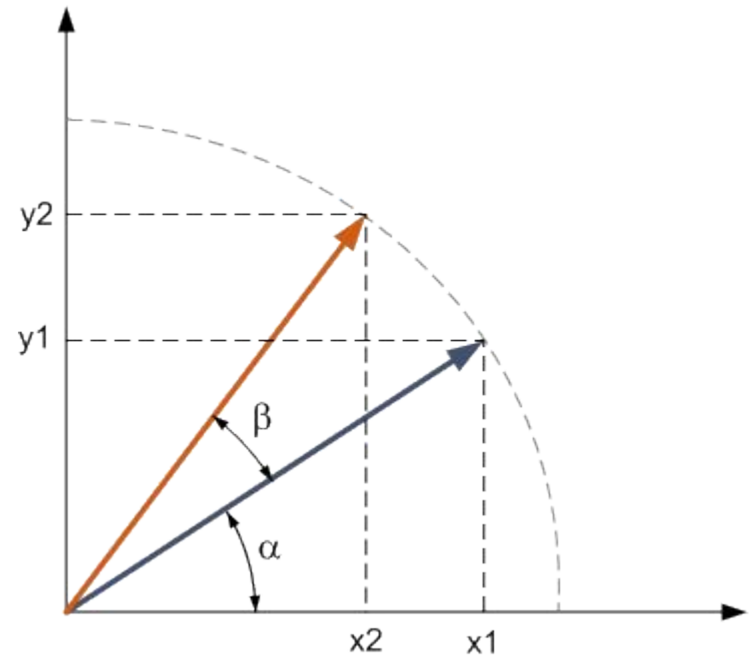
- Sólo utiliza las operaciones de suma, resta y desplazamiento

- Se lo utiliza para la rotación de vectores, el cálculo de funciones trigonométricas (sen, cos, tan, etc), la transformación de coordenadas, etc

Ecuaciones

1
$$x1 = A \,.\, \cos(\alpha)$$
$$y1 = A \,.\, sen(\alpha)$$

2
$$x2 = A \,.\, \cos(\alpha + \beta)$$
$$y2 = A \,.\, sen(\alpha + \beta)$$



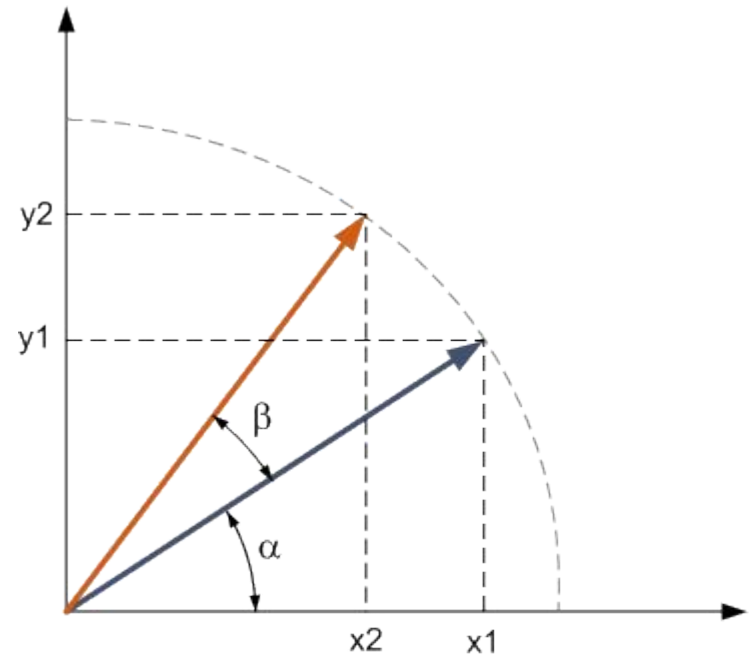
Ecuaciones

Desarrollando el coseno y el seno de la ecuación 2:

(3)

$$x2 = [A \,.\, \cos(\alpha).\cos(\beta) - A \,.\, \mathrm{sen}(\alpha) \,.\, \mathrm{sen}(\beta)]$$

$$y2 = [A \,.\, \mathrm{sen}(\alpha).\cos(\beta) + A \,.\, \cos(\alpha) \,.\, \mathrm{sen}(\beta)]$$



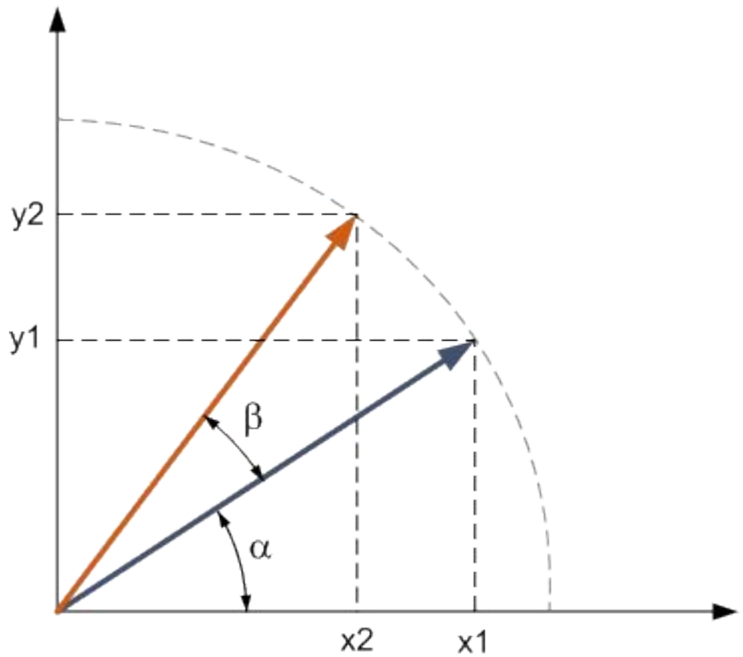
Ecuaciones

Sacando cos(β) como factor común en la ecuación 3:

4

$$x2 = \cos(\beta)\,.\,[A\,.\,\cos(\alpha) - A\,.\,\mathrm{sen}(\alpha)\,.\,\mathrm{tg}(\beta)]$$

$$y2 = \cos(\beta)\,.\,[A\,.\,\mathrm{sen}(\alpha) + A\,.\,\cos(\alpha)\,.\,\mathrm{tg}(\beta)]$$

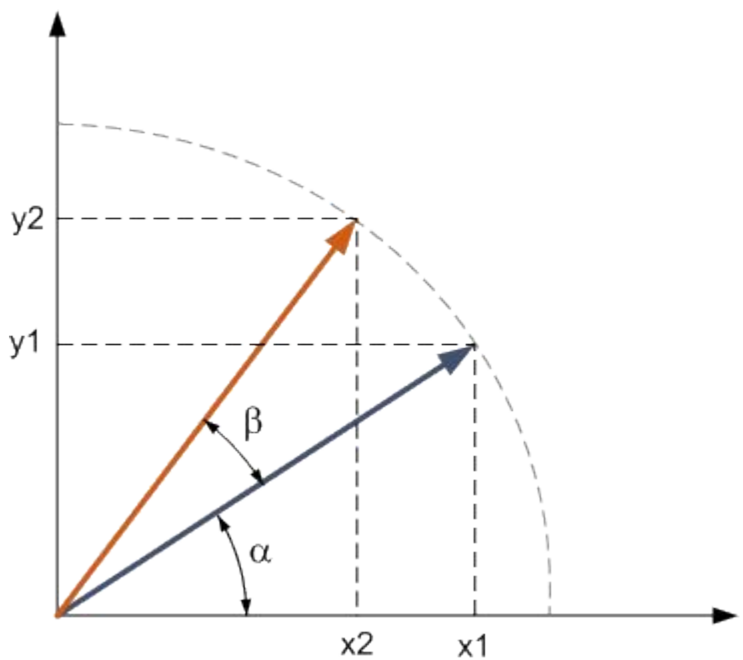


Ecuaciones

Sustituyendo la ecuación 1 en 4:

$$x2 = \cos(\beta)\,.\,[\underbrace{A\,.\cos(\alpha)}_{x1} - A\,.\,\mathrm{sen}(\alpha)\,.\,\mathrm{tg}(\beta)]$$

$$y2 = \cos(\beta)\,.\,[A\,.\,\mathrm{sen}(\alpha) + \underbrace{A\,.\cos(\alpha)}_{x1}\,.\,\mathrm{tg}(\beta)] \tag{4}$$

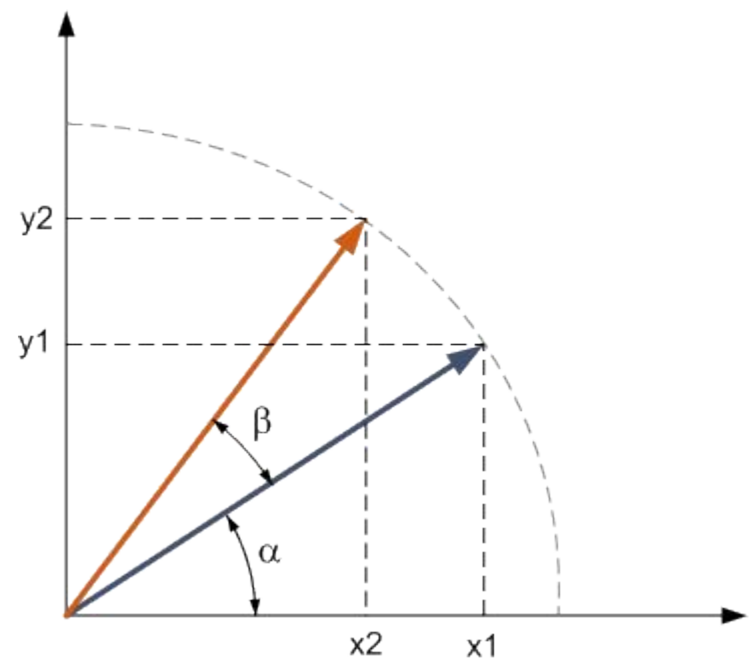


Ecuaciones

Sustituyendo la ecuación 1 en 4:

$$x2 = \cos(\beta) . [A . \cos(\alpha) - A . \mathrm{sen}(\alpha) . \mathrm{tg}(\beta)] \quad (4)$$
$$y2 = \cos(\beta) . [A . \mathrm{sen}(\alpha) + A . \cos(\alpha) . \mathrm{tg}(\beta)]$$

y1



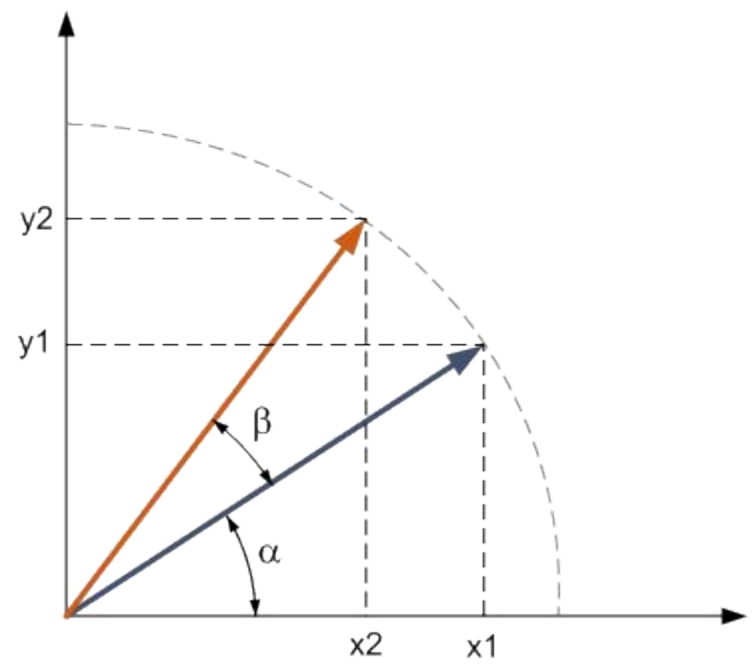
Ecuaciones

Sustituyendo la ecuación 1 en 4:

$$x2 = \cos(\beta).[x1 - tg(\beta).y1]$$

$$y2 = \cos(\beta).[y1 + tg(\beta).x1]$$

Givens



Ecuaciones

Hasta este momento no hemos llegado a nada interesante.
Ahora, pensemos por un momento qué pasaría si los ángulos en los que puede rotar el vector se restringen de tal manera que:

$$tg(\beta) = \pm\, 2^{-i}$$

¿Se lograría algún beneficio con esto?

¡La multiplicación en el término que incluye la tangente se reduce a una simple operación de desplazamiento!

Ecuaciones

Por lo dicho:

$$tg(\beta) = \pm 2^{-i} \quad \Rightarrow \quad \beta = tg^{-1}(\pm 2^{-i}) \quad \Rightarrow \quad \cos(\beta) = \cos[tg^{-1}(\pm 2^{-i})] \quad \Rightarrow$$

$$\Rightarrow \quad \cos(\beta) = \cos[tg^{-1}(2^{-i})] = \boxed{K_i = \frac{1}{\sqrt{1+2^{-2i}}}}$$

Luego, reemplazando en las ecuaciones de Givens:

$$x_{i+1} = K_i \, [\, x_i - y_i \, (\, d_i \, . \, 2^{-i} \,)\,]$$

$$y_{i+1} = K_i \, [\, y_i + x_i \, (\, d_i \, . \, 2^{-i} \,)\,]$$

El producto de los Ki's se puede tratar como una ganancia del sistema. La ganancia exacta depende de la cantidad de iteraciones y es aproximadamente igual a 1,647:

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

Ecuaciones

Por lo dicho:

$$tg(\beta) = \pm 2^{-i} \quad \Rightarrow \quad \beta = tg^{-1}(\pm 2^{-i}) \quad \Rightarrow \quad \cos(\beta) = \cos[tg^{-1}(\pm 2^{-i})] \quad \Rightarrow$$

$$\Rightarrow \quad \cos(\beta) = \cos[tg^{-1}(2^{-i})] = \boxed{K_i = \frac{1}{\sqrt{1+2^{-2i}}}}$$

Luego, reemplazando en las ecuaciones de Givens:

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

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

El valor de d_i será ± 1

Ecuaciones

El algoritmo incluye una tercera ecuación que describe el acumulador angular:

$$z_{i+1} = z_i - d_i . tg^{-1}(2^{-i})$$

Modos de operación

El algoritmo puede operar en dos modos diferentes:

- **Modo rotación**
  - Rota un vector en un ángulo especificado
  - El acumulador angular se inicializa con el ángulo a rotar
  - La decisión de rotación en cada iteración se lleva a cabo de tal manera de disminuir el ángulo residual en el acumulador angular (se utiliza su signo)

- **Modo vector**

- Rota un vector hacia el eje de coordenadas x, guardando los ángulos requeridos para lograrlo
- Busca minimizar la componente y del vector residual
- La dirección de rotación se decide por el signo de la componente y residual

Modos de operación

- **Modo rotación**

$$x_{i+1} = x_i - y_i.d_i.2^{-i}$$
$$y_{i+1} = y_i + x_i.d_i.2^{-i}$$
$$z_{i+1} = z_i - d_i.tg^{-1}(2^{-i})$$

Donde: d_i = -1 si z_i < 0, en otro caso +1

Finalmente se tiene:

$$x_n = A_n[x_0.\cos(z_0) - y_0.\sin(z_0)]$$
$$y_n = A_n[y_0.\cos(z_0) + x_0.\sin(z_0)]$$
$$z_n = 0$$

Modos de operación

- **Modo vector**

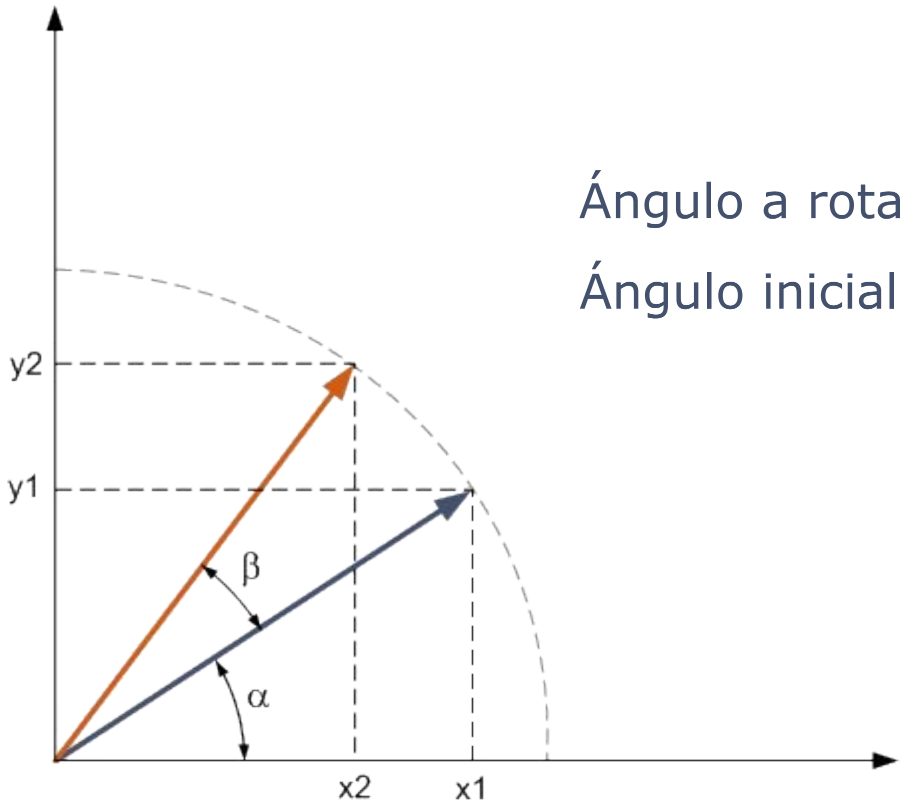
$$x_{i+1} = x_i - y_i.d_i.2^{-i}$$
$$y_{i+1} = y_i + x_i.d_i.2^{-i}$$
$$z_{i+1} = z_i - d_i.tg^{-1}(2^{-i})$$

Donde: $d_i = +1$ *si* $y_i < 0$*, en otro caso -1*

Finalmente se tiene:

$$x_n = A_n \sqrt{x_0^2 + y_0^2}$$
$$y_n = 0$$
$$z_n = z_0 + tg^{-1}\left(y_0 / x_0 \right)$$

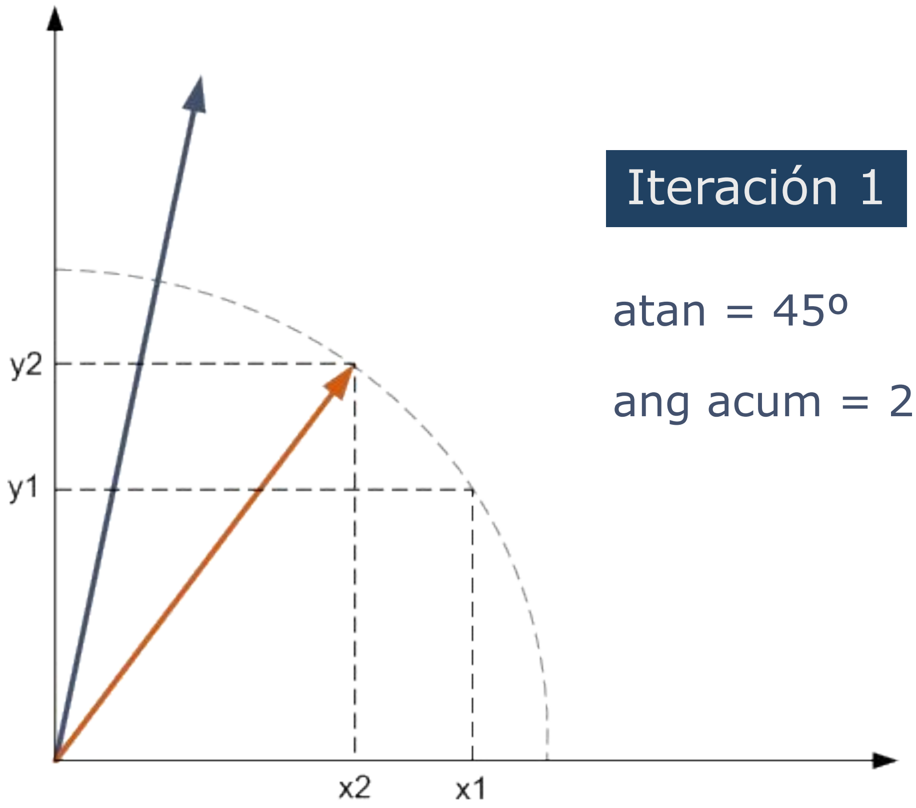
Ejemplo



Ángulo a rotar: 20º

Ángulo inicial: 33º

Ejemplo

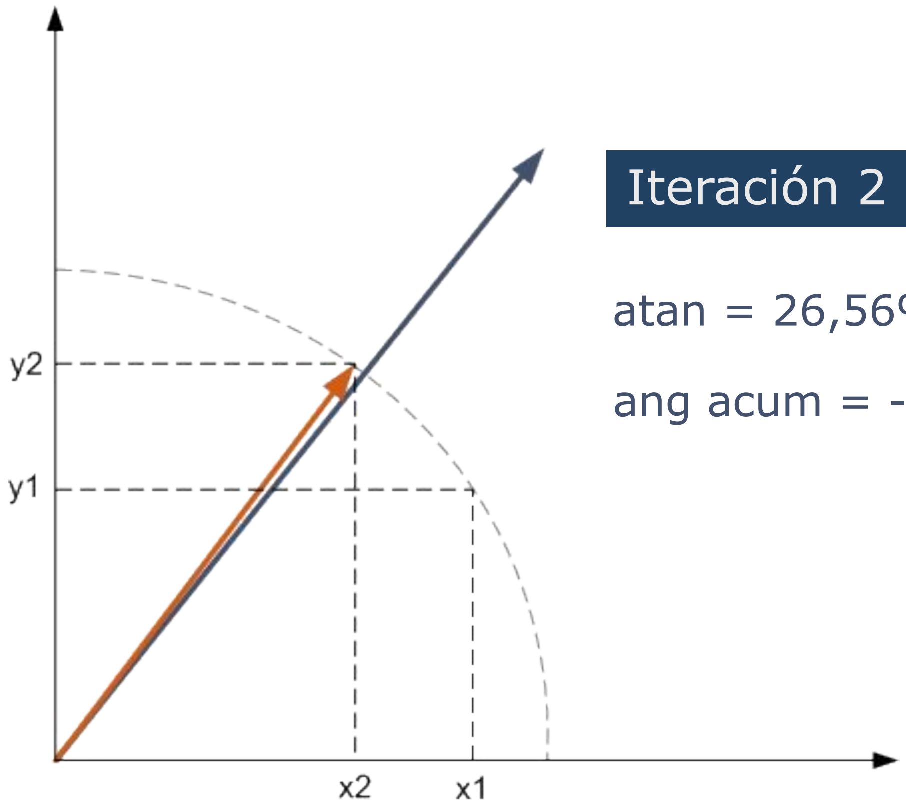


Iteración 1

atan = 45º

ang acum = 20 - 45 = -25

Ejemplo

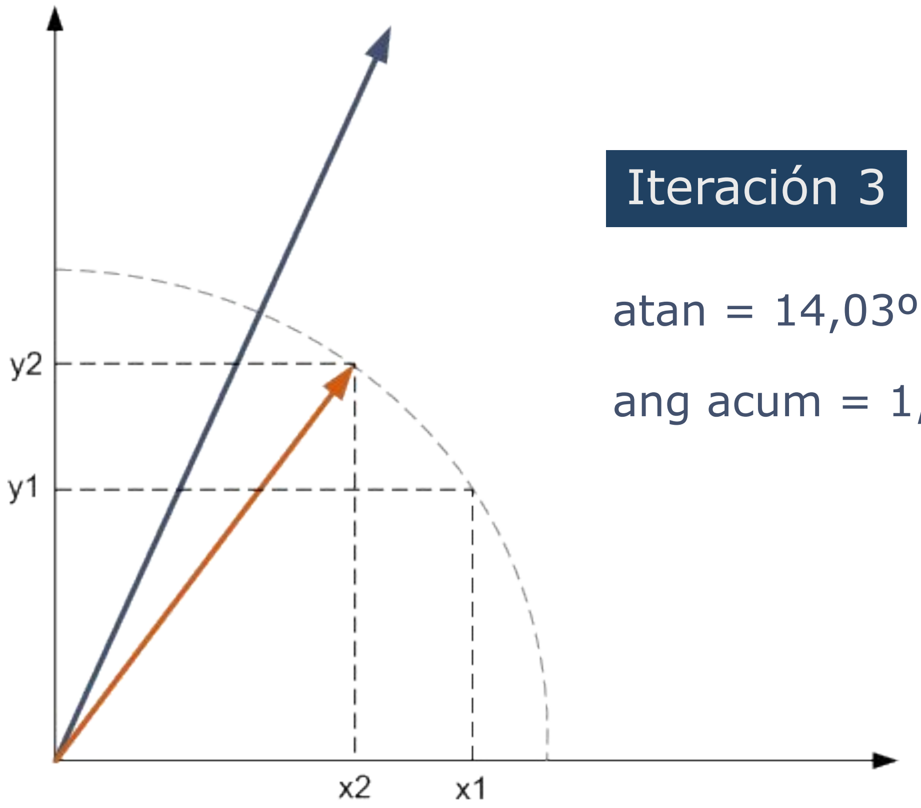


Iteración 2

atan = 26,56º

ang acum = -25 + 26,56 = 1,56

Ejemplo

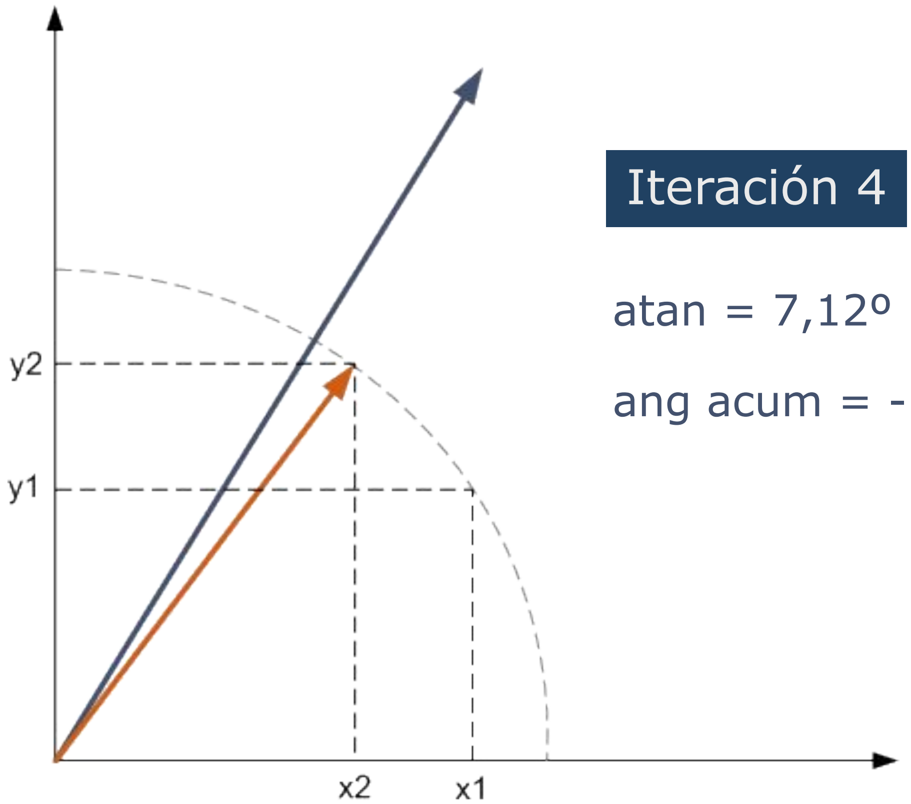


Iteración 3

atan = 14,03º

ang acum = 1,56 - 14,03 = -12,47

Ejemplo

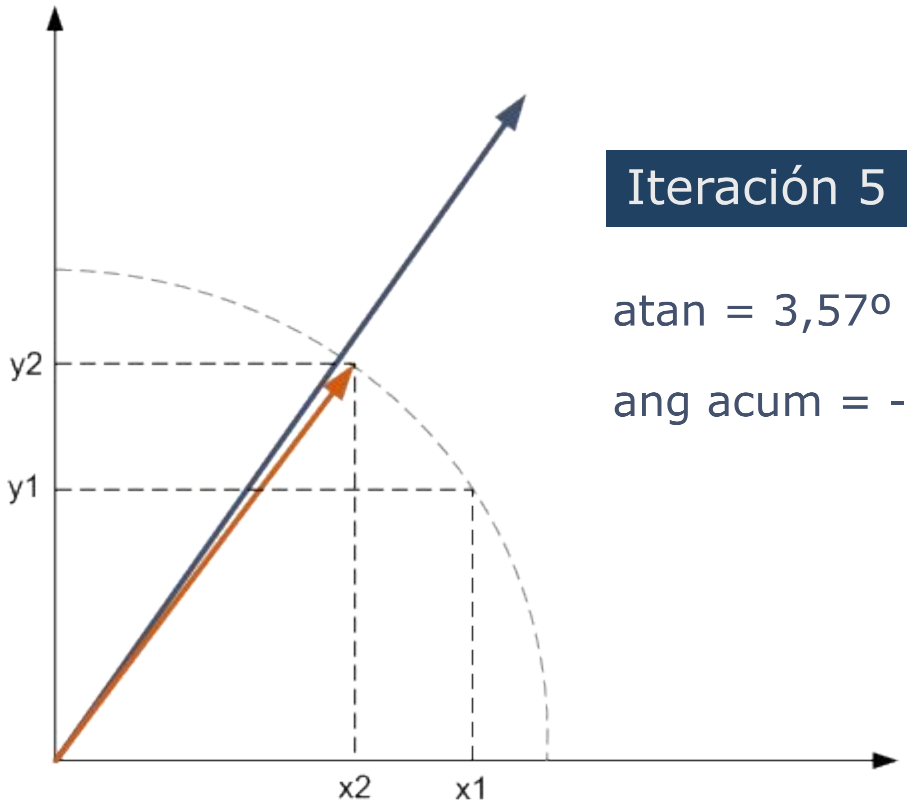


Iteración 4

atan = 7,12º

ang acum = -12,47 + 7,12 = -5,35

Ejemplo

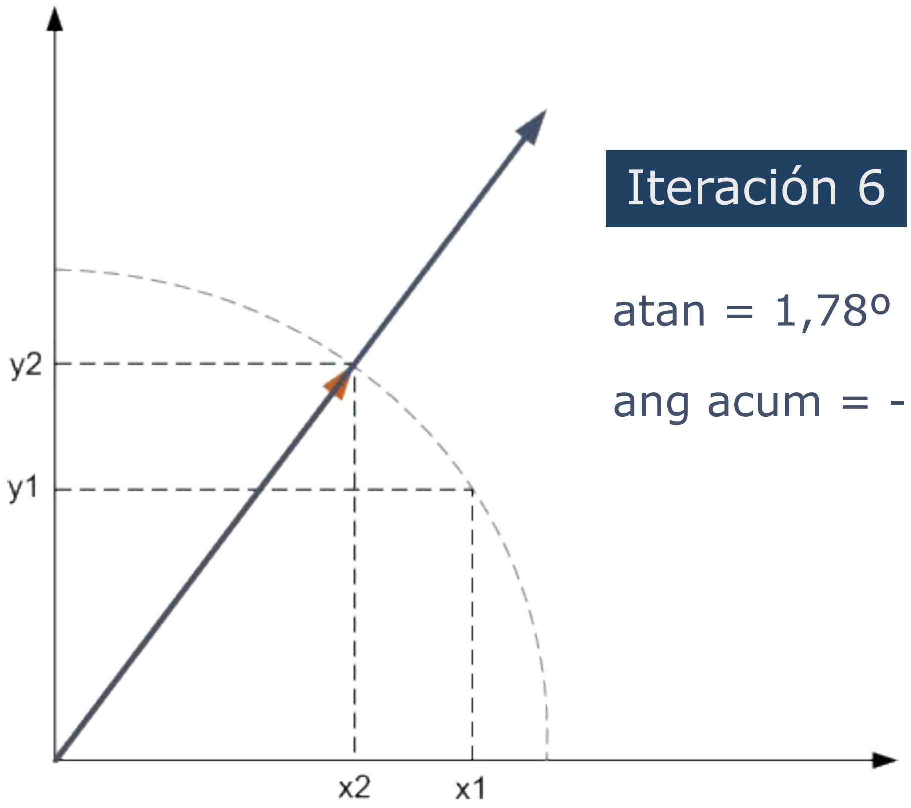


Iteración 5

atan = 3,57º

ang acum = -5,35 + 3,57 = -1,78

Ejemplo



Iteración 6

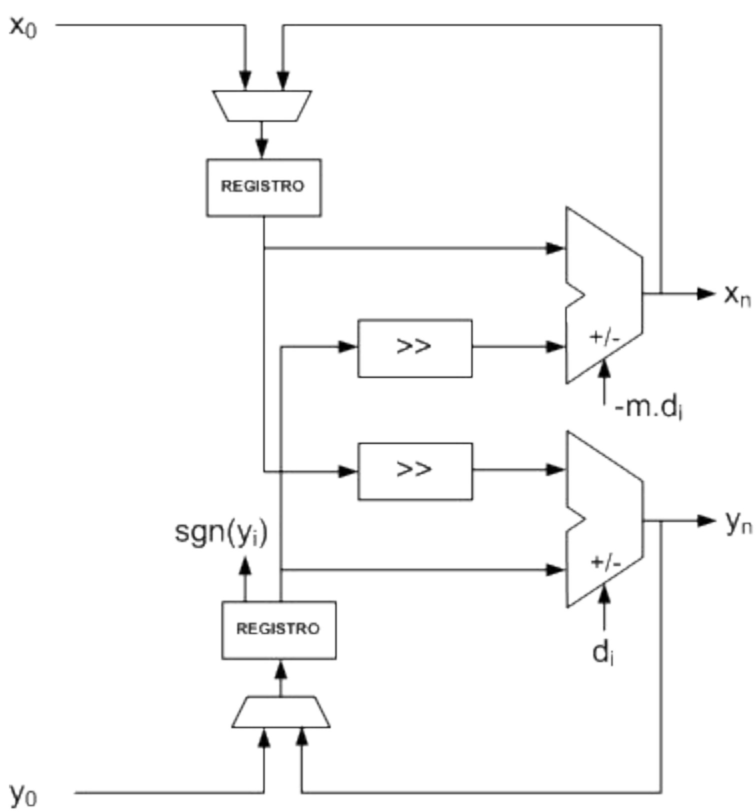
atan = 1,78º

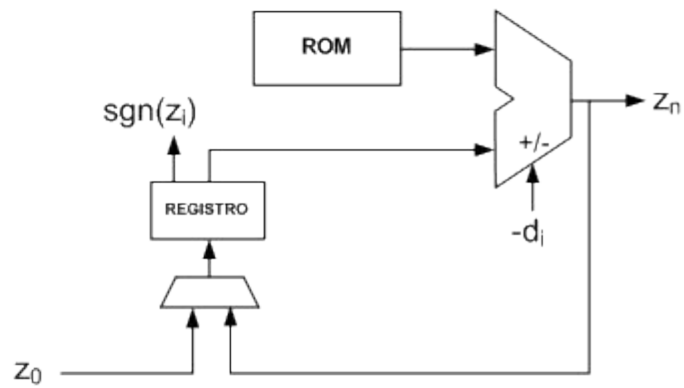
ang acum = -1,78 + 1,78 = 0

Arquitecturas

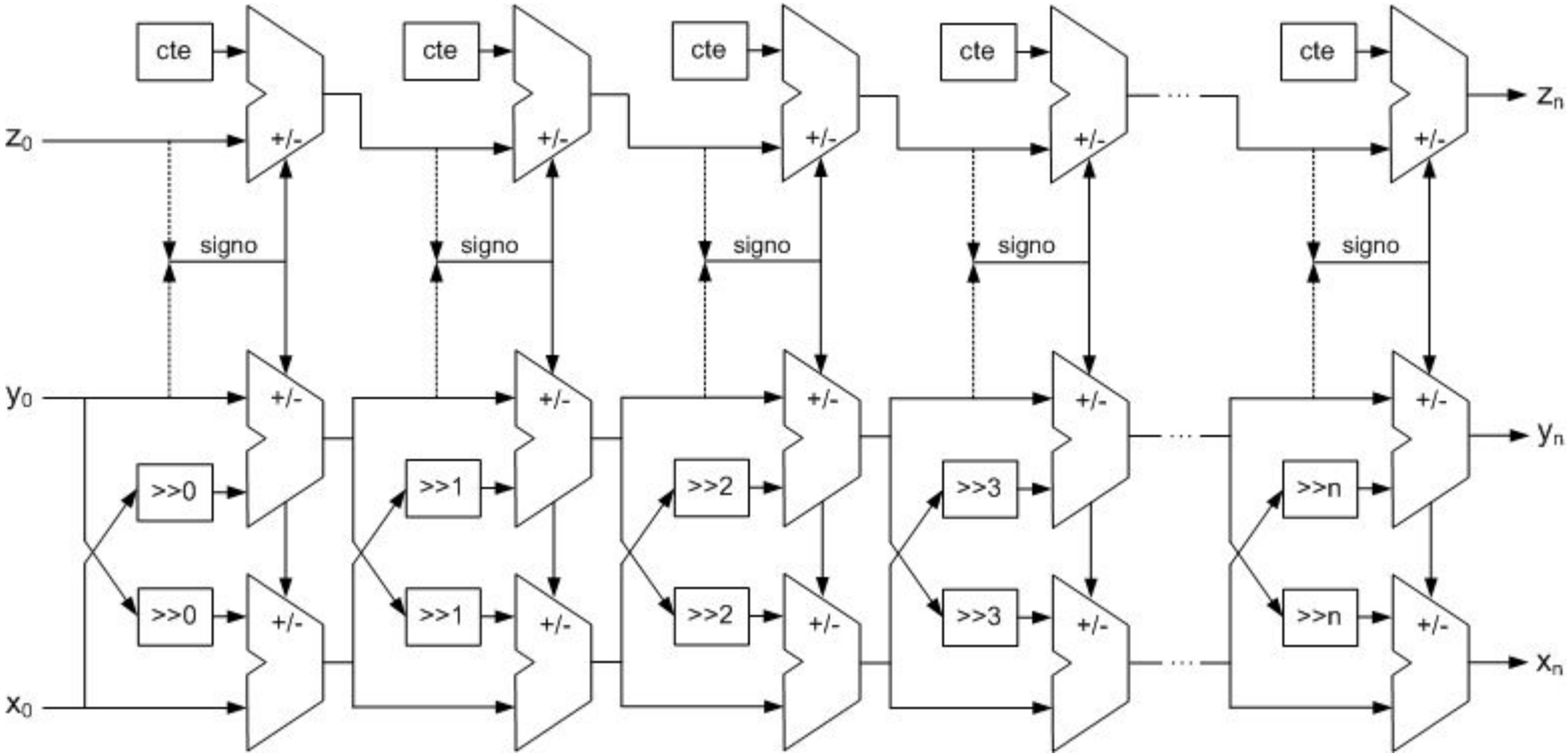
- Iterativa
- Unrolled
- Pipeline unrolled

Arquitectura iterativa

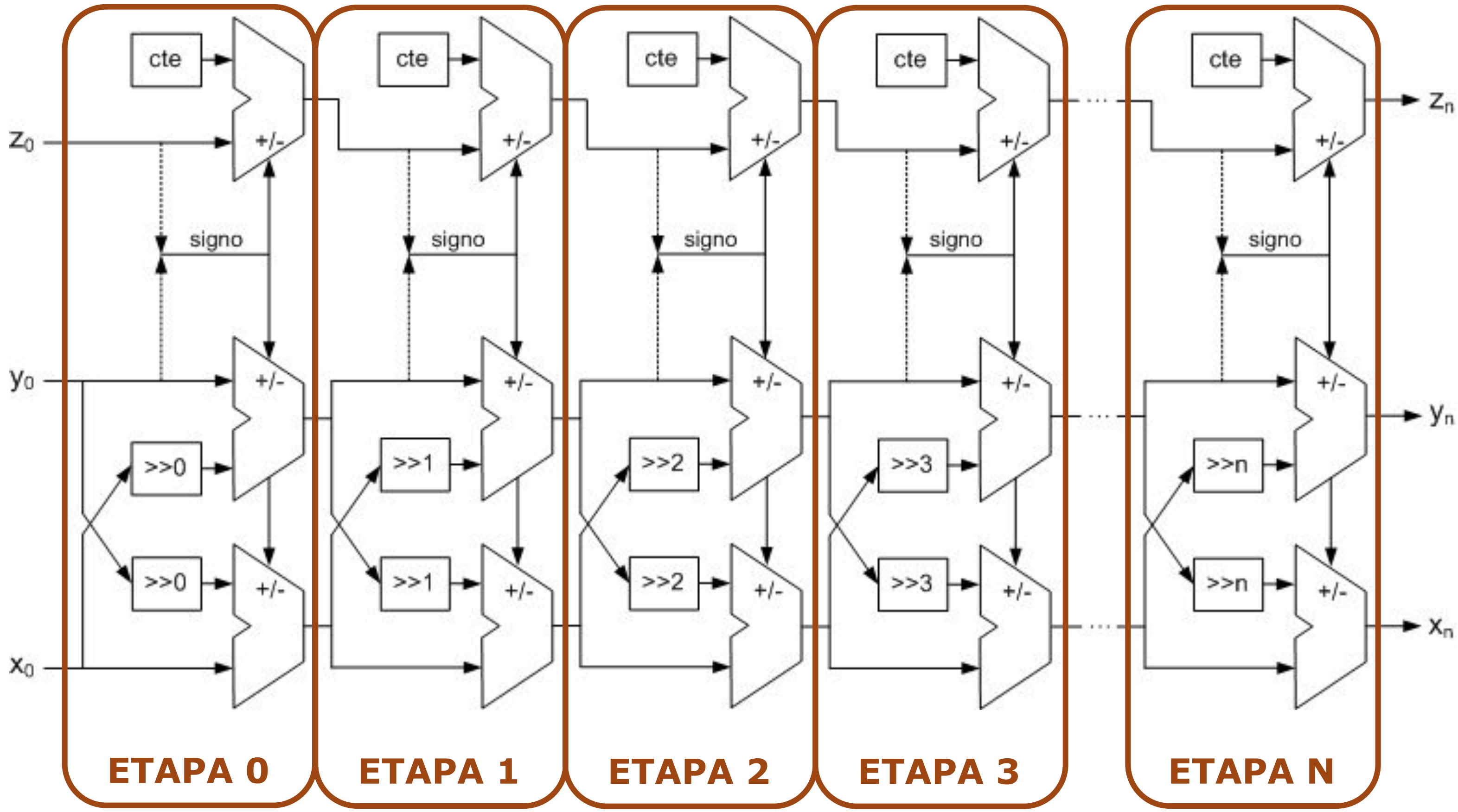




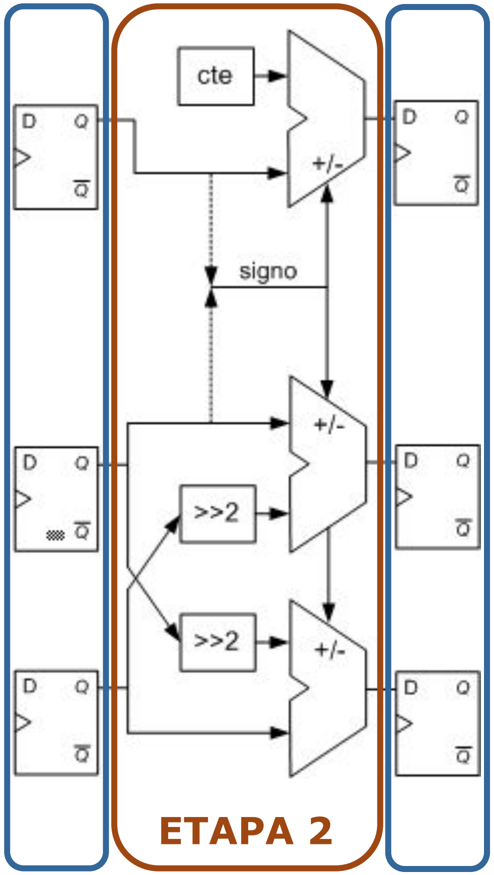
Arquitectura Unrolled



Arquitectura Pipeline Unrolled



Arquitectura Pipeline Unrolled



Ejercicios

1) Determinar cómo calcularía el seno y el coseno de un ángulo dado utilizando el algoritmo CORDIC.

2) Determinar cómo transformaría coordenadas polares en cartesianas utilizando el algoritmo CORDIC.

3) Determinar cómo calcularía la arcotangente de un ángulo dado utilizando el algoritmo CORDIC.

4) Determinar cómo transformaría coordenadas cartesianas en polares utilizando el algoritmo CORDIC.

FIN