

## Actividad 7.2

1) La generación de PRNG sobre LFSRs implica la generación de un bit-siguiente en función de un par de bit-actuales. Stephen Wolfram propuso distintas clasificaciones según el resultado de los números generados, y dentro de cada clasificación una serie de reglas que indican la función lógica que deberá cumplir cada celda según sus entradas. Algunas de ellas pueden ser las reglas Regla90 y Regla150, pertenecientes al grupo de Reglas de clase 3 (tipo caótico o pseudo-aleatorio), las cuales se describen de la siguiente manera

TABLA I  
REGLA 90 PARA LAS 8 POSIBLES CONFIGURACIONES BINARIAS DE 3 BITS

111	110	101	100	011	010	001	000
0	1	0	1	1	0	1	0

TABLA II  
REGLA 150 PARA LAS 8 POSIBLES CONFIGURACIONES BINARIAS DE 3 BITS

111	110	101	100	011	010	001	000
1	0	0	1	0	1	1	0

donde  $01011010_2 = 90_{10}$  y  $10010110_2 = 150_{10}$ . Y en función de la celda se pueden expresar de la siguiente manera

- *regla 90*  $\rightarrow x_{t+1}^i = x_t^{i-1} \oplus x_t^{i+1}$
- *regla 150*  $\rightarrow x_{t+1}^i = x_t^{i-1} \oplus x_t^i \oplus x_t^{i+1}$

Respecto a PRNG de 4 entradas encontré el ejemplo citado en la presentación

### 4.1 CA50745 RNG

Several one-dimensional Cellular Automata have been found to pass the DIEHARD suite. We considered, for instance, CA38490 with connections  $\{-3, 0, 4, 9\}$  and CA50745 with connections  $\{-7, 0, 11, 17\}$ , according to the terminology used in [8]. Both of them pass the DIEHARD suite, ensuring that the entropy of the generated sequence is enough to really constitute a RNG. In the practice, CA38490 is considerably slower in advancing to a high-entropy state, so we selected CA50745 as the RNG for the system and compared it with the classic LFSR.

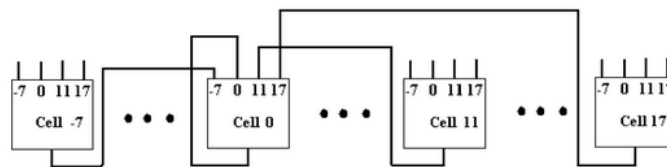


Fig. 9. Cell connectivity for the CA50745 RNG

We implemented the RNG as a one-dimensional automata with 4-input cells, connected to the lattice following the  $\{-7, 0, 11, 17\}$  schema. This is shown in Fig. 9.

2-bit RNG can be implemented by selecting two random bits from a greater RNG. Therefore, a  $N$ -bit RNG gives support to a number  $n$  of 2-bit RNG that is given by the formula:

$$n = \binom{N}{2} = \frac{N(N-1)}{2} .$$

En lo que es la inicialización, se debe procurar no caer en los estados prohibidos según la regla o metodología utilizada para el cálculo del estado próximo, ya que si, en las Reglas previamente mencionadas, el sistema arranca con todas las salidas en '0', el PRNG no logrará avanzar.

Para simular un PRNG se tomó el de la Regla150, como se muestra en el siguiente código

```
entity En_PRNG_150 is
  Generic( N : NATURAL := 10);
  Port (
    CLK : in STD_LOGIC;
    RST : in STD_LOGIC;
    RN : out STD_LOGIC_VECTOR(N-1 downto 0)
  );
end En_PRNG_150;

architecture Arq_PRNG_150 of En_PRNG_150 is
  signal sRN : STD_LOGIC_VECTOR(N-1 downto 0);
  signal sRN_Reg : STD_LOGIC_VECTOR(N-1 downto 0);
begin

  sRN(0) <= '0' XOR sRN_Reg(0) XOR sRN_Reg(1) when RST = '0' else '0';
  bucle: for i in 1 to N-2 generate
    sRN(i) <= sRN_Reg(i-1) XOR sRN_Reg(i) XOR sRN_Reg(i+1) when RST = '0' else '0';
  end generate;
  sRN(N-1) <= sRN_Reg(N-2) XOR sRN_Reg(N-1) XOR '0' when RST = '0' else '0';

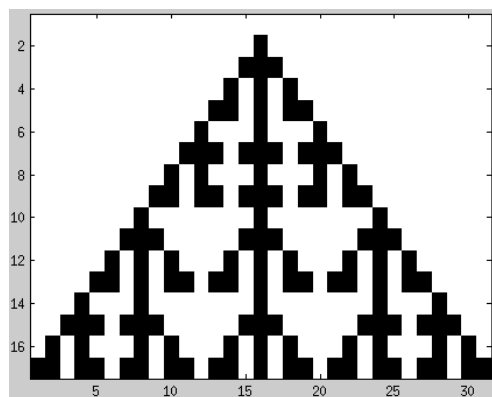
  process(CLK,RST)
  begin
    if RST = '1' then
      sRN_Reg <= (others=>'0');
      sRN_Reg((N-1)/2) <= '1';
    elsif rising_edge(CLK) then
      sRN_Reg <= sRN;
    end if;
  end process;

  RN <= sRN_Reg;
end Arq_PRNG_150;
```

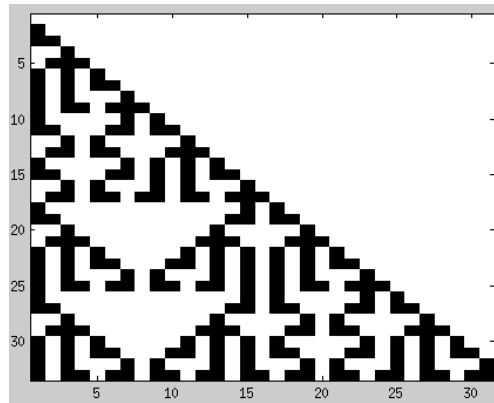
donde se ve que al momento del comienzo (RST = '1') se coloca una semilla en el bit central, ya que si todo comienza en cero el PRNG no avanza...

Se trabajó con una longitud de 31 bits en todos los siguientes ensayos.

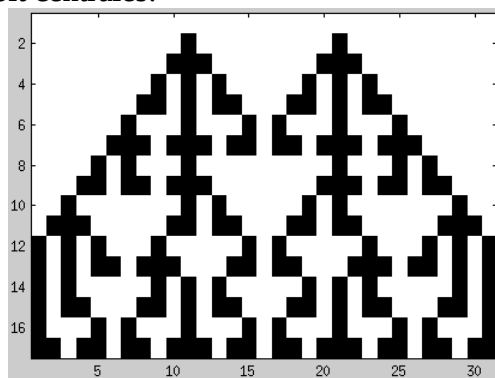
Con lo siguiente se simuló, se exportó a un archivo y se manipuló en MATLAB de manera de poder graficarlo como se encuentra en la mayoría de las fuentes



Con semilla en en el primer bit:



Con la semilla en dos bit centrales:



2) A continuación se recopilan algunas de los ensayos encontrados

### PRUEBA DE CHI-CUADRADO

Esta es la prueba más comúnmente usada. En general, puede ser usada para cualquier distribución. A partir de un histograma, se comparan las frecuencias observadas con las frecuencias obtenidas de la distribución específica (frecuencias esperadas). Si el histograma tiene  $k$  celdas o intervalos, y  $O_i$  y  $E_i$  son las frecuencias observadas y esperadas respectivamente para la  $i$ -ésima celda, la prueba consiste en calcular

$$\chi_0^2 = \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i}$$

Si el ajuste es exacto  $\chi_0^2$  es cero, pero por aleatoriedad no lo será. Se puede demostrar que  $\chi_0^2$  tiene distribución chi-cuadrado con  $k-1$  grados de libertad. Se aceptará que los datos tienen la distribución en prueba con un nivel de significancia de  $\alpha$  si  $\chi_0^2 < \chi_{[\alpha; k-1]}^2$ , donde  $\alpha$  es la probabilidad de cometer Error de Tipo I: rechazar  $H_0$  cuando es verdadera.

Estrictamente hablando, la prueba de chi-cuadrado está diseñada para distribuciones discretas y para muestras grandes. Para distribuciones continuas la chi-cuadrado es solo una aproximación y el nivel de significación se aplica solo si  $n \rightarrow \infty$ .

### PRUEBA DE KOLMOGOROV-SMIRNOV

La prueba K-S nos permite decidir si una muestra de  $n$  observaciones es de una distribución continua particular. Se basa en que la diferencia entre la FDA (Función de Distribución Acumulada) observada  $S_n(x)$  y la FDA esperada  $F_X(x)$  debe ser pequeña.  $S_n(x)$  viene dada por:

$$S_n(x) = \frac{\text{número de observaciones} \leq x}{n}$$

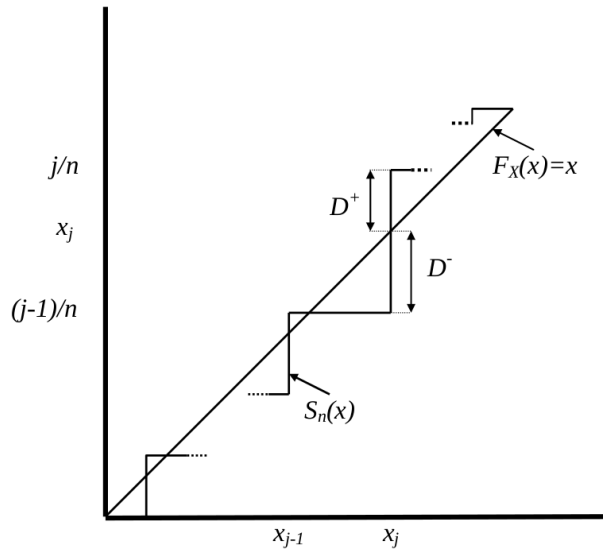
Los símbolos  $D^+$  y  $D^-$  son usados para denotar las desviaciones observadas máximas sobre y bajo la FDA esperada en una muestra de tamaño  $n$ :

$$D^+ = \max_x [S_n(x) - F_X(x)] \quad D^- = \max_x [F_X(x) - S_n(x)]$$

$D^+$  mide la desviación máxima cuando la FDA observada esta sobre la FDA esperada, y  $D^-$  mide la desviación máxima cuando la FDA observada esta bajo la FDA esperada. Si los valores de  $D^+$  y  $D^-$  son menores que  $D_{[\alpha; n]}$  entonces se dice que las observaciones provienen de la distribución respectiva con un nivel de significación de  $\alpha$ .

Un error común calculando  $D^-$  consiste en encontrar el máximo de  $F_X(x_j) - S_n(x_j)$ . Esto es incorrecto ya que  $S_n$  consiste de un segmento horizontal a  $S_n(x_j)$  en el intervalo  $[x_j, x_{j+1})$  y el máximo ocurre justo antes de  $x_{j+1}$  y la diferencia correcta es  $F_X(x_{j+1}) - S_n(x_j)$ .

$$D^+ = \max_j \left( \frac{j}{n} - x_j \right) \quad D^- = \max_j \left( x_j - \frac{j-1}{n} \right)$$



### PRUEBA DE CORRELACIÓN SERIAL

Un método para probar si hay dependencia entre dos variables es calculando su covarianza. Si esta es distinta de cero, entonces son dependientes. En inverso no es cierto; si la covarianza es cero no implica que sean independientes.

Dada una secuencia de números aleatorios se puede calcular la covarianza entre números  $k$ -distantes, es decir, entre  $x_i$  y  $x_{i+k}$ . Esto es llamado autocovarianza con desplazamiento  $k$  denotada como  $R_k$  y dada por:

$$R_k = \frac{1}{n-k} \sum_{i=1}^{n-k} \left( x_i - \frac{1}{2} \right) \left( x_{i+k} - \frac{1}{2} \right)$$

donde  $x_i$  es el  $i$ -ésimo número aleatorio uniforme de la secuencia.

Para  $n$  grande,  $R_k$  se distribuye normalmente con media 0 y varianza  $1/[144(n-k)]$ . El intervalo de confianza para la autocovarianza al  $100(1-\alpha)\%$  es

$$R_k \pm \frac{z_{1-\alpha/2}}{12\sqrt{n-k}}$$

Si este intervalo no incluye el cero, podemos decir que la secuencia tiene correlación significativa. Esto se aplica solo para  $k \geq 1$ . Si  $k = 0$ ,  $R_0$  es la varianza de la muestra que se espera sea  $1/12$  para una secuencia independiente idénticamente distribuida (IID) de uniformes en 0-1 ( $U(0,1)$ ).

### UNIFORMIDAD K-DIMENSIONAL O K-DISTRIBUTIVA

Las pruebas anteriores aseguran uniformidad en una dimensión. El concepto de uniformidad puede ser extendido a  $k$  dimensiones.

Supongamos que  $u_n$  es el  $n$ -ésimo número en una secuencia distribuida uniformemente entre 0 y 1. Dados dos números reales  $a_1$  y  $b_1$ , también entre 0 y 1 y tal que  $b_1 > a_1$ , la probabilidad de que  $u_n$  este en el intervalo  $[a_1, b_1)$  es  $b_1 - a_1$ :

$$P(a_1 \leq u_n < b_1) = b_1 - a_1 \quad \forall b_1 > a_1$$

Esta es la propiedad 1-distributiva de  $u_n$ .

La 2-distributividad es una generalización de esta propiedad en dos dimensiones y requiere que un par de valores  $u_{n-1}$  y  $u_n$  satisfagan la siguiente condición:

$$P(a_1 \leq u_{n-1} < b_1 \text{ y } a_2 \leq u_n < b_2) = (b_1 - a_1)(b_2 - a_2) \quad \forall b_1 > a_1 \text{ y } \forall b_2 > a_2$$

Una secuencia es  $k$ -distributiva si:

$$P(a_1 \leq u_n < b_1, \dots, a_k \leq u_{n+k-1} < b_k) = (b_1 - a_1) \dots (b_k - a_k) \quad \forall b_i > a_i \quad i = 1, 2, \dots, k$$

Notar que una secuencia  $k$ -distributiva es  $(k-1)$ -distributiva, pero el inverso no es cierto. Una secuencia puede ser uniforme en una dimensión inferior y no en una superior. Dado un grupo de generadores, es preferible aquel que produzca más uniformidad en la mayor dimensión.

Antes de conducir estas pruebas, es conveniente chequear si la secuencia es uniforme en dos dimensiones graficando pares solapados sucesivos:  $(x_{n-1}, x_n)$ ,  $(x_n, x_{n+1})$ .

3) Para la implementación de CORDIC se implementó de manera de ingresar tanto las proyecciones en X y en Y del vector  $(X_0, Y_0)$  como así también el ángulo inicial respecto al eje X ( $Z_0$ ).

Por otro lado también se ingresa el ángulo deseado, que en caso de vectorización, su valor 0 o 90 dependerá si se desea la componente en X o en Y respectivamente.

Para almacenar los valores de  $\arctan()$  se implementó una ROM con un COE File de valores constantes. Se propuso trabajar en ángulos multiplicados por diez para poder trabajar con números enteros con precisión de una décima de grado ya que se trata de una validación del método. En caso de ser funcional, solo se debe aumentar la profundidad y valores de la ROM y todo debería seguir funcionando solo que con diferentes escalabilidades de ángulos.

Se propone enviar como salida de soporte el valor RUN, el cual anuncia la cantidad de iteraciones realizadas. El caso general es que haga todas las previstas, pero puede suceder que se llegue al ángulo deseado en un número menor de iteraciones, caso en el cual la posterior corrección por escala sería incorrecto ya que hay iteraciones que nunca se realizaron y el error sería demasiado grosero. De esta manera, el valor de calibración sería:

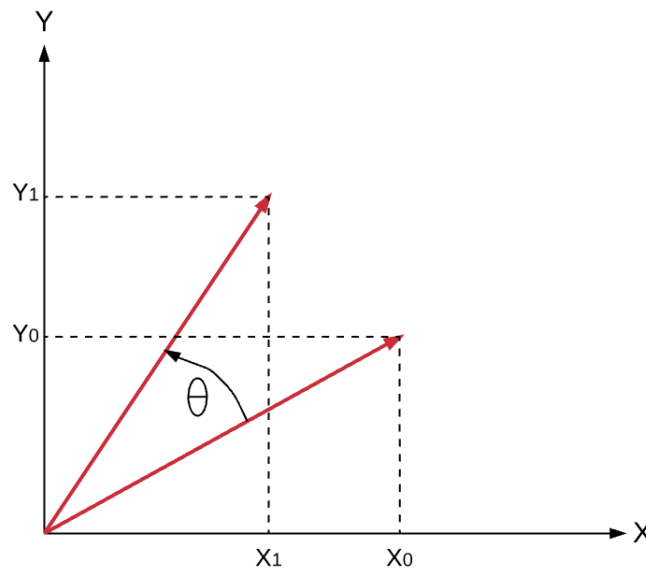
$$K = 1;$$

for  $i = 1: \text{RUN}$

$$K = K * \cos(\text{atan}(2^{-i}));$$

Donde  $K \rightarrow 0.607253$ , siendo el valor de calibración  $K^{-1} = 1.646760$

Presetaré un pequeño ejemplo para poder tener un punto de referencia



Suponiendo un caso en que

$$X_0 = 10$$

$$Y_0 = 10$$

$$Z_0 = 45^\circ$$

$$A_0 = 14.14$$

Y se lo quiere rotar hacia

$$Z_1 = 75^\circ$$

Los valores esperados teóricamente serían

$$X1 = X0 \cdot \cos(Z1 - Z0) - Y0 \cdot \sin(Z1 - Z0) = 3.66 = K \cdot 6.027$$

$$Y1 = X0 \cdot \sin(Z1 - Z0) + Y0 \cdot \cos(Z1 - Z0) = 13.66 = K \cdot 22.495$$

$$A1 = 14.14$$

Para poder lidiar con estas operaciones en VHDL lo que se hizo fue trabajar con ángulos multiplicados por 10 (se podría hacer con más precisión pero solo sería cambiar la profundidad de la ROM y no agrega nada al objetivo real) y con coordenadas multiplicadas por 1000 para tener acceso hasta a 3 decimales.

Ingresando los siguientes valores

$$X0 = 10000 \text{ (10)}$$

$$Y0 = 10000 \text{ (10)}$$

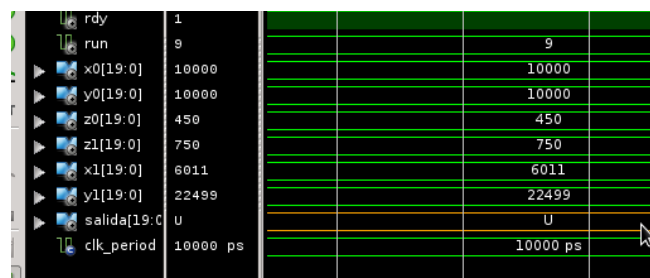
$$Z0 = 450 \text{ (45°)}$$

$$Z1 = 750 \text{ (75°)}$$

Se obtuvo

$$X1 = 6011 \text{ (6.011)} = 3650.2/K \text{ (3.6502/K)}$$

$$Y1 = 22499 \text{ (22.499)} = 13662.6/K \text{ (13.6626/K)}$$



Signal	Value
rdy	1
run	9
x0[19:0]	10000
y0[19:0]	10000
z0[19:0]	450
z1[19:0]	750
x1[19:0]	6011
y1[19:0]	22499
salida[19:0]	U
clk_period	10000 ps

Ahora si nos movemos al segundo cuadrante

$$X0 = -10000 \text{ (-10)}$$

$$Y0 = 10000 \text{ (10)}$$

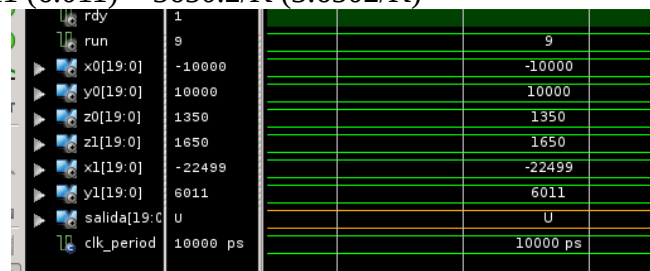
$$Z0 = 1350 \text{ (135°)}$$

$$Z1 = 1650 \text{ (165°)}$$

Se obtuvo

$$X1 = -22499 \text{ (-22.499)} = -13662.6/K \text{ (-13.6626/K)}$$

$$Y1 = 6011 \text{ (6.011)} = 3650.2/K \text{ (3.6502/K)}$$



Signal	Value
rdy	1
run	9
x0[19:0]	-10000
y0[19:0]	10000
z0[19:0]	1350
z1[19:0]	1650
x1[19:0]	-22499
y1[19:0]	6011
salida[19:0]	U
clk_period	10000 ps

Obteniéndose resultados válidos también en el resto de los cuadrantes



4) La aproximación al CORDIC para funciones hiperbólicas nacen de la generalización del mismo algoritmo

## 1.5 CORDIC generalizado

El algoritmo CORDIC básico puede ser generalizado para proveer una herramienta aún mas potente para el cálculo de funciones. Esta generalización fue propuesta por Walther [5]. Se define

$$\begin{aligned}x_{i+1} &= x_i - \mu y_i d_i 2^{-i} \\y_{i+1} &= y_i + x_i d_i 2^{-i} \\z_{i+1} &= z_i - d_i f(2^{-i})\end{aligned}\tag{1.21}$$

La única diferencia que presenta este algoritmo con el básico, es la introducción del parámetro  $\mu$  en la ecuación correspondiente al componente  $x$  y la función  $f(\cdot)$  que adquiere significado según el sistema que se utilice. El parámetro  $\mu$  y la función  $f(\cdot)$  pueden asumir uno de los tres valores que se muestran a continuación

$\mu = 1$	y	$f(x) = \arctg(x)$	Rotación Circular (CORDIC básico)
$\mu = 0$	y	$f(x) = x$	Rotación Lineal
$\mu = -1$	y	$f(x) = \tgh^{-1}(x)$	Rotación Hiperbólica

### 1.5.2 Caso hiperbólico

Si se parte de (1.21) con  $\mu = -1$  y  $f(x) = \tgh^{-1}(x)$  se obtiene

$$\begin{aligned}x_{i+1} &= x_i + y_i d_i 2^{-i} \\y_{i+1} &= y_i + x_i d_i 2^{-i} \\z_{i+1} &= z_i - d_i \tgh^{-1}(2^{-i})\end{aligned}$$

Para el modo rotación, donde  $d_i = \begin{cases} -1 & ,si\ z_i < 0 \\ 1 & ,si\ z_i \geq 0 \end{cases}$

la rotación hiperbólica produce

$$\begin{aligned} x_n &= A_n (x_0 \cosh z_0 + y_0 \sinh z_0) \\ y_n &= A_n (y_0 \cosh z_0 + x_0 \sinh z_0) \\ z_n &= 0 \\ A_n &= \prod_{i=0}^{n-1} \sqrt{1 - 2^{-2i}} \cong 0,80 \end{aligned} \quad (1.26)$$

Para el modo vectorización, donde  $d_i = \begin{cases} -1 & ,si\ y_i \geq 0 \\ 1 & ,si\ y_i < 0 \end{cases}$

la rotación hiperbólica produce

$$\begin{aligned} x_n &= A_n \sqrt{x_0^2 - y_0^2} \\ y_n &= 0 \\ z_n &= z_0 + \operatorname{tgh}^{-1} \left( \frac{y_0}{x_0} \right) \\ A_n &= \prod_{i=0}^{n-1} \sqrt{1 - 2^{-2i}} \end{aligned} \quad (1.27)$$

Las rotaciones elementales en el sistema hiperbólico no convergen. Sin embargo Walther [5] demostró que se pueden conseguir los resultados esperados si se repiten algunas de las iteraciones efectuadas [6].

Las funciones trigonométricas hiperbólicas equivalentes a las funciones trigonométricas circulares pueden obtenerse de manera similar a partir de la ecuación 1.26.

Referencias:

- [https://www.researchgate.net/publication/39664001\\_Modelizacion\\_del\\_generador\\_auto-shrinking\\_mediante\\_automatas\\_celulares](https://www.researchgate.net/publication/39664001_Modelizacion_del_generador_auto-shrinking_mediante_automatas_celulares)
- <https://books.google.com.ar/books?id=zpU3xIaf9i4C&pg=PA392&lpg=PA392&dq=wolfram+lfsr+4+entradas&source=bl&ots=eVUDfENHac&sig=JKFqDoz7RCAVg6uWM5hD9l5ZcO0&hl=es&sa=X&ved=0ahUKEwi0qN6-4ovVAhVM5CYKHVHdDPAQ6AEIMjAC#v=onepage&q&f=false>
- <http://webdelprofesor.ula.ve/ingenieria/hhoeger/simulacion/PARTE5.pdf>
- [http://sedici.unlp.edu.ar/bitstream/handle/10915/3835/1 - El algoritmo CORDIC.pdf?sequence=6](http://sedici.unlp.edu.ar/bitstream/handle/10915/3835/1_-_El_algoritmo_CORDIC.pdf?sequence=6)