*Fractales de Mandelbrot*

Adriana Andrea Aguirre   
*Facultad de Ingeniería*   
*Pontificia Universidad Javeriana*Bogotá, Colombias  
aguirreadriana@javeriana.edu.co

Maria Paola Fonseca   
*Facultad de Ingeniería*   
*Pontificia Universidad Javeriana*Bogotá, Colombias  
fonseca\_maria@javeriana.edu.co

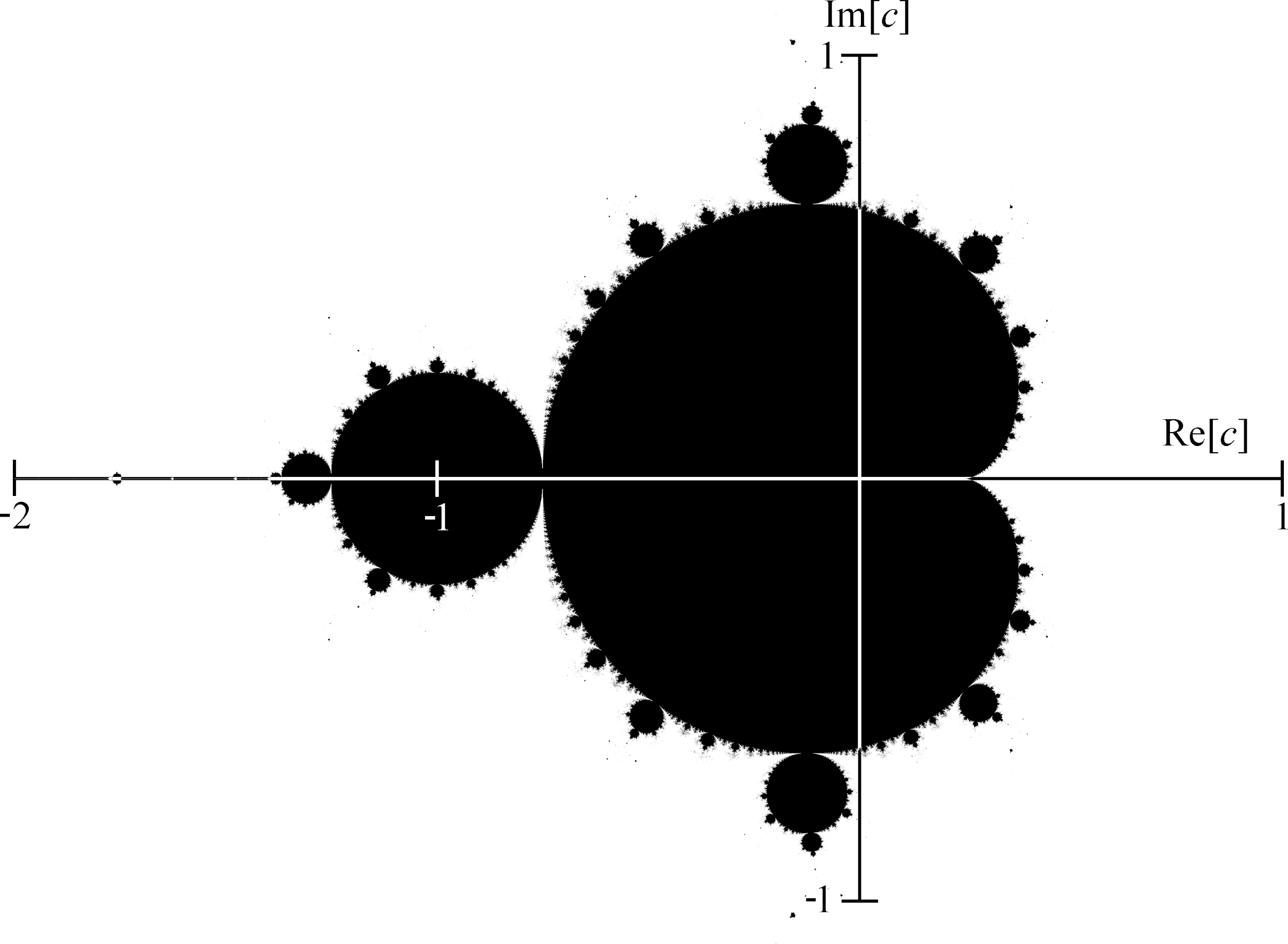
Silvana López   
*Facultad de Ingeniería*   
*Pontificia Universidad Javeriana*Bogotá, Colombias  
silvana-lopez@javeriana.edu.co

***Abstract*—En este documento se presenta la concepción, diseño e implementación de un sistema que dibuja un fractal de Mandelbrot en blanco y negro en un monitor; este se construye a partir de una sucesión por recursión, la cual si queda acotada se dice que el punto evaluado pertenece al conjunto de Mandelbrot, y si no, queda excluido del mismo[1]. El proyecto fue inicialmente implementado para su desarrollo en *software* en *Matlab*, y para su desarrollo en *hardware* en *VHDL* por medio del programa *Quartus II 13.0*, utilizando la tarjeta de desarrollo de *ALTERA DE0 Cyclone III EP3C16F484C6*. El sistema se diseñó de tal forma que inicialmente se visualiza el dibujo del fractal desde las posiciones cartesianas [-2,-1] a [1,1], junto con el tiempo que le tomó al programa calcular el conjunto para 480x480 pixeles dependiendo del número de iteraciones para cada sucesión. Adicionalmente en el sistema diseñado es posible realizar ‘*zoom*’ o acercamientos sobre el dibujo, el cual, puede ser visualizado desde cualquier pantalla con entrada *VGA,* por medio de un cable *VGA-VGA* que se conecta directamente desde la tarjeta al monitor. Para los controles del zoom se utilizaron los pulsadores de la tarjeta y para la visualización del tiempo se realizó comunicación serial entre la tarjeta y un computador, de forma que el resultado se representa en el programa *RealTerm* con la línea “TIEMPO DE EJECUCIÓN: XX.XX s”.**

***Keywords—VHDL, VGA, Fractal de Mandelbrot, iteración, posición, distancia, máquina de estados, diagrama de bloques, simulación, pixel.***

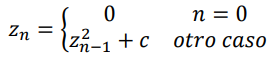
# Introducción

Un fractal es una forma geométrica segmentada que puede ser divida en partes más pequeñas, de apariencia idéntica o similar a la figura original. Uno de los generadores de fractales más estudiados es el conjunto de Mandelbrot, el cual es resultado de las investigaciones del matemático Benoît Mandelbrot (1924-2010) [2]. Para el caso del fractal de Mandelbrot, el cálculo del mismo se rige por ecuaciones matemáticas simples y los límites de un conjunto conforman en fractal. A pesar de usar operaciones simples, requieren de varias iteraciones para determinar la pertenencia o no pertenencia al conjunto, por lo que resulta ser computacionalmente intensivo lograr dibujar una pantalla completa. A continuación se muestra el dibujo del fractal de Mandelbrot.

 Figura 1. Conjunto de Mandelbrot

# Marco Teórico

Un conjunto de Mandelbrot está compuesto por una serie de números complejos, los cuales generan a su vez un límite tipo fractal cuando son dibujados. Se especifica de forma tal que la secuencia definida en el plano complejo mostrada en la ecuación (1) bajo las que se realizan las iteraciones, está compuesta de una serie de puntos para los cuales se tiene que para cualquier número complejo dado se encuentra dentro del conjunto si el valor absoluto de zn está acotado cuando n tiende a infinito.

 (1)

Para el cálculo del conjunto de Mandelbrot sobre un plano cartesiano se debe tener en cuenta que se calcula un punto en el espacio, con una coordenada x y una coordenada y, por lo que las ecuaciones que definen el conjunto de Mandelbrot se muestran en la ecuación (2).

 (2)

De forma que para cualquier punto del plano (cx,cy) dado, este hace parte del conjunto de Mandelbrot si la suma de sus cuadrados es acotada cuando el número de iteraciones tiende a infinito. Por ejemplo, si (cx,cy) es igual a (1,0), la secuencia se generada corresponde a (0,0)(1,0)(2,0)(5,0),(26,0) y aumenta tendiendo a infinito, sin embargo para (cx,cy) igual a (-1,0), la secuencia se generada corresponde a (0,0)(—1,0) (0,0)(—1,0)…, la cual está acotada, de forma que el punto (1,0) no pertenece al conjunto de Mandelbrot mientras que el punto (-1,0) sí lo hace. La imagen de Mandelbrot es generada en el plano graficando cada punto del conjunto en el plano y son coloreados de negro si pertenecen al conjunto, y de blanco los que no pertenecen.

Para el desarrollo del proyecto se deben tener en cuenta las especificaciones de las señales del *VGA,* las cuales deben cumplir con el patrón de funcionamiento ilustrado en la figura 2, así como la asignación de los 12 bits que definen el color de cada pixel, y que corresponden a los valores mostrados en la figura 3, el funcionamiento de cada entrada del conector *VGA* ilustrado en la figura 4, y el funcionamiento del controlador VGA.

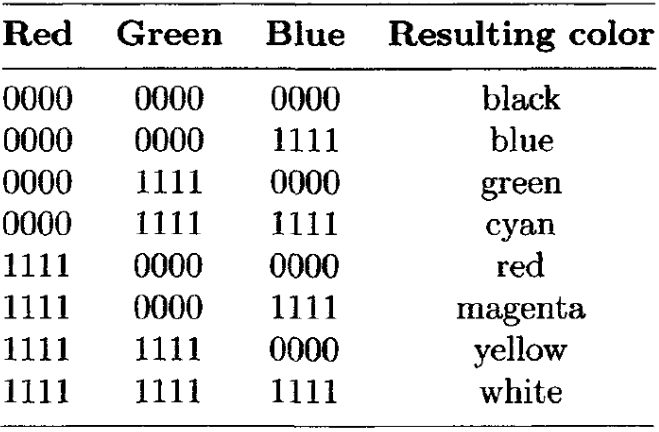
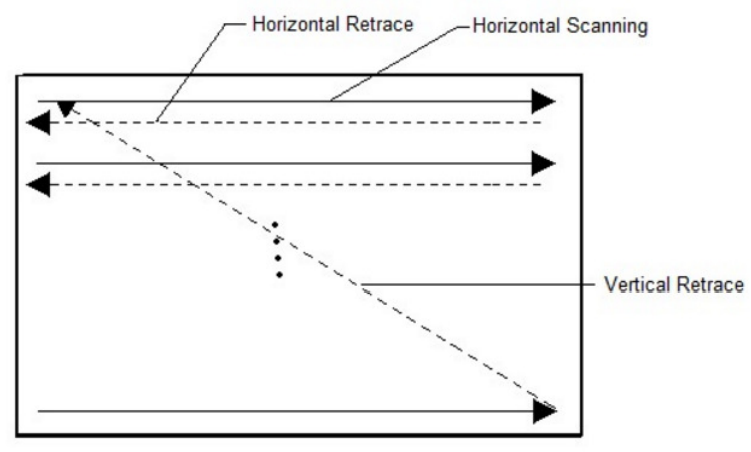


Figura 2. señales del VGA[1]. Figura 3. Color CRT [1].

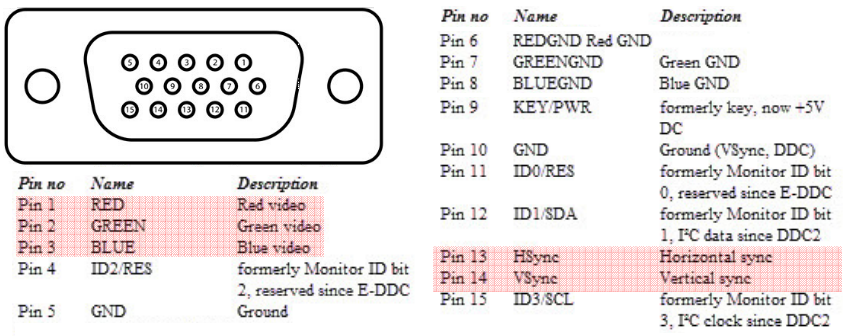


Fig. 4. Conector VGA [1].

El conector *VGA* maneja principalmente 5 señales; *hsync, vsync, red, blue* y *green*[3]. Para el funcionamiento del controlador *VGA* se debe tener en cuenta el arreglo de video del *VGA*, el protocolo serial-serial para generar la sincronización de las señales y la información para cada pixel. Estos fueron inicialmente diseñados para una pantalla CRT, sin embargo las pantallas LCD heredaron el standard.

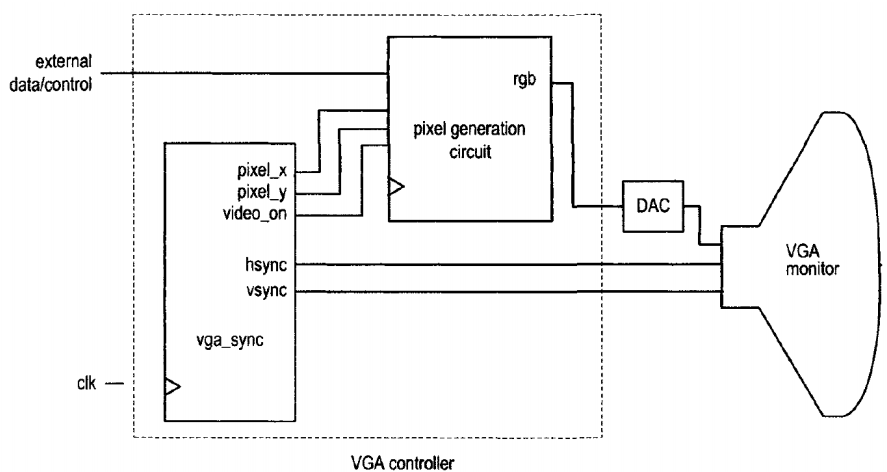


Fig.5. Controlador VGA [1].

Para el controlador del monitor *VGA* se utiliza uno para el cual se tiene un bloque que genera por un lado las señales de hsync y vsync que corresponden a los tiempos de impresión verticales y horizontales de los píxeles de la pantalla, la cual se recorre horizontalmente de izquierda a derecha para imprimir cada línea y verticalmente de arriba a abajo del monitor. De forma que al imprimir toda la pantalla una vez, se tiene un frame. Por el otro lado este bloque genera las señales de video \_on, junto con los pixeles en x y y,los cuales generan pixel a pixel el color que debe tomar el mismo para la posición que se encuentra.

Para el desarrollo del proyecto se requiere una resolución de 640 x 480 pixeles a 38.4 kB (640\*480\*1/8) mono o 461 kB (640\*480\*12/8) 12-bits. Esto significa que la pantalla debe refrescarse a una velocidad de 25 M pixeles por segundo, a una frecuencia de 60 Hz como velocidad de frame, de forma que cada 16.66 ms debe pasar un frame. Para que un frame sea impreso, se deben imprimir los 640 píxeles de 12 bits existentes en una línea horizontal de la pantalla, para las 480 líneas de píxeles existentes de la pantalla.

La cantidad de pixeles y formas de las señales horizontales se muestran en la figura 5, mientras que la cantidad de pixeles y formas de las señales verticales se muestran en la figura 6. Realizando los cálculos respectivos, cada 40 ns se realiza la lectura e impresión de cada píxel en cada línea.

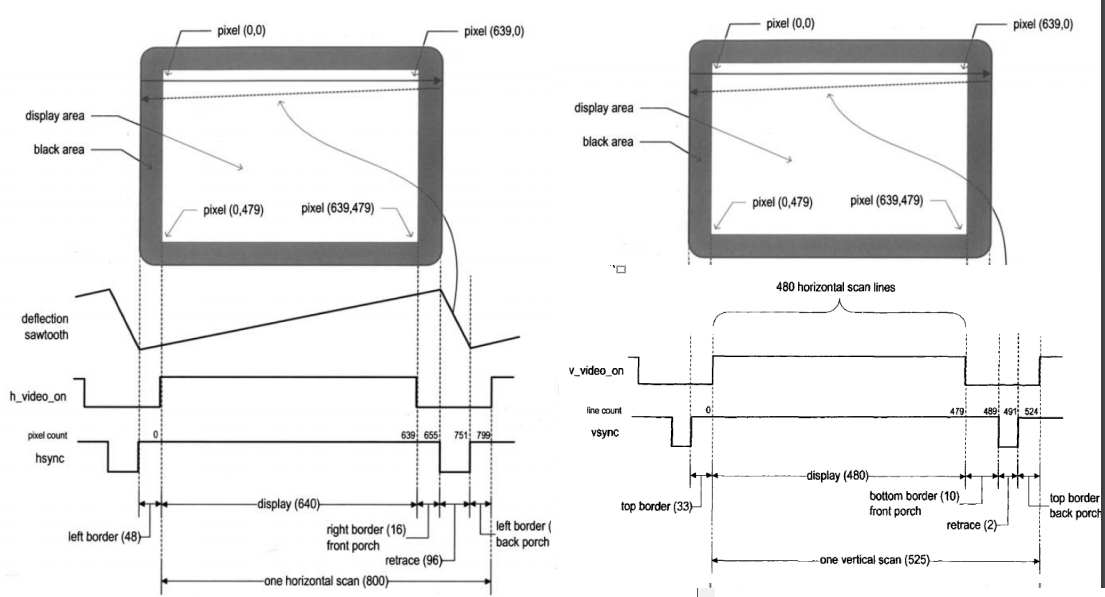


Fig.6. Señales controlador horizontal VGA [1]. Fig.7.Señales controlador vertical VGA [1]

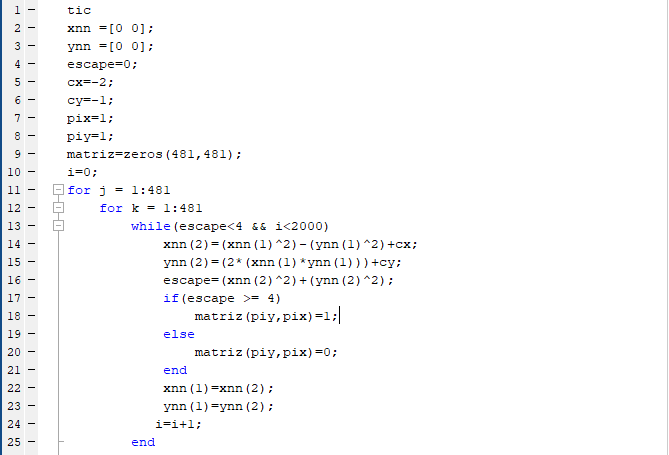
Como se puede apreciar en las figuras 6 y 7, la pantalla completa ocupa un espacio de tiempo entre cada píxel y cada línea, de forma que no se podrá ver la imagen en estos intervalos de tiempo. Estos intervalos de tiempo corresponden a los tiempos en los que se leen los píxeles existentes detrás de los marcos de la pantalla y el tiempo que se demora el barredor devolviendose desde el final de una línea al comienzo de una nueva. Teniendo esto en cuenta en realidad existen 800x525 píxeles de los cuales se resta cada tiempo muerto de forma que a la pantalla llegan 480 líneas de 640 píxeles, existentes en los límites de la pantalla de video on, imprimiendo de negro los demás espacios.

# Diseño del Sistema

*Implementación en Software:*

Para la concepción en *software* se implementó el algoritmo que busca la generación del fractal el programa Matlab 2018b, para el cual se implementó un código que genera el cálculo de una matriz de 480x480 posiciones en las cuales, se almacena un ‘0’ cuando el punto calculado pertenece al conjunto de Mandelbrot, y un ‘1’ cuando el punto calculado no pertenece. La determinación de estos puntos se realiza mediante un barrido de forma que desde la primera columna se calculan las 480 posiciones pertenecientes a la misma, y luego se deducen las de la siguiente columna, hasta que se calculan las 480 posiciones de las 480 columnas a evaluar. El resultado obtenido es una matriz la cual se convierte en dibujo mediante el comando *imshow* el cual toma una matriz de ‘1’ y ‘0’ e interpreta los ‘1’ como blanco y ‘0’ como negro, para realizar una imagen.

A continuación se muestra el código utilizado para la implementación en software de la generación del fractal de Mandelbrot en la figura 8.



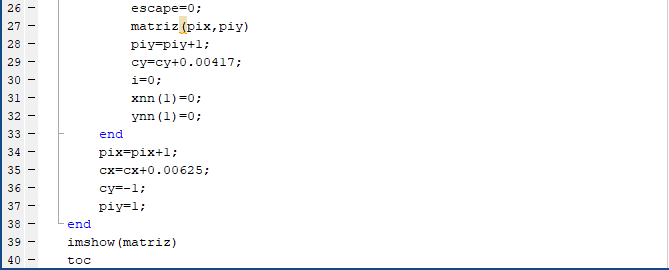


Figura 8. código de Matlab.

En la figura 9 se ilustra el resultado obtenido para el dibujo del fractal de Mandelbrot realizado a partir del conjunto de Mandelbrot generado en el programa anteriormente enunciado.

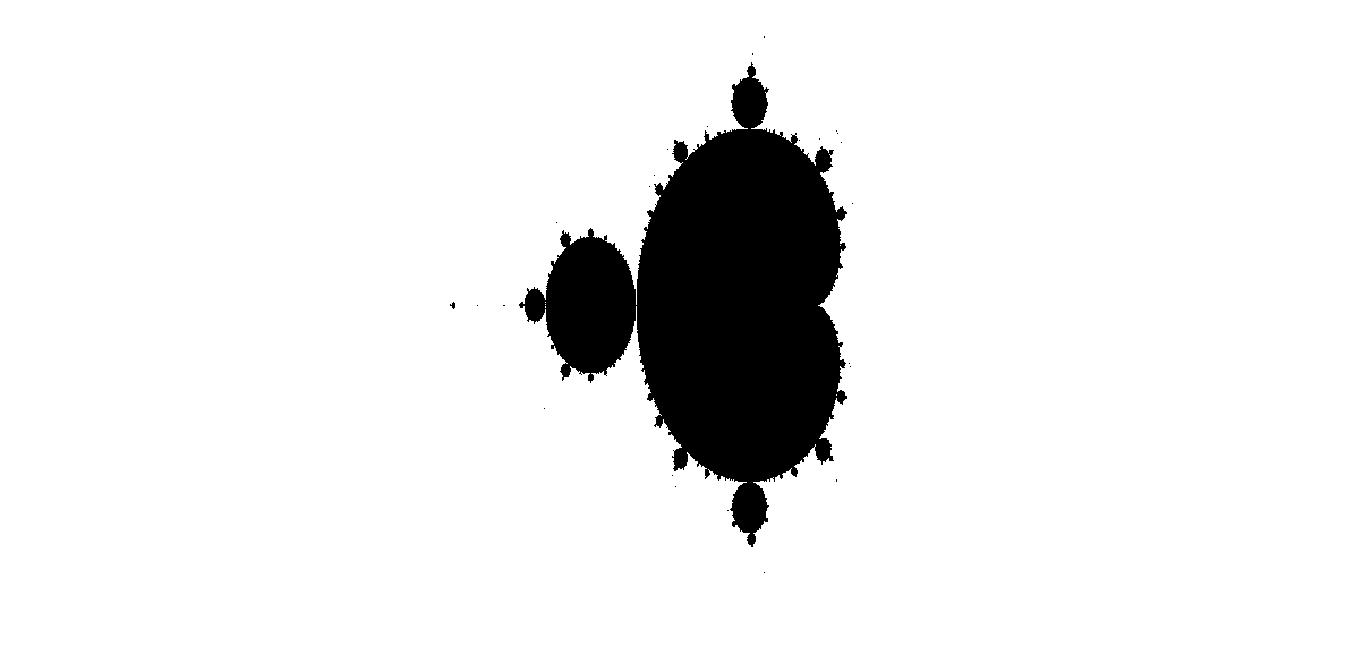


Figura 9. Fractal de Mandelbrot [-2,-1] a [1,1].

La imagen obtenida se muestra en la figura 9, donde se tiene la figura del fractal definido desde los límites [-2,-1] a [1,1], de manera que el cálculo de cada posición de la matriz se realiza sumando un delta de distancia desde los límites inferiores ([-2,-1]), de forma que solamente estos dos límites y los deltas de distancia determinan los límites superiores del dibujo. Las ecuaciones que definen los límites de distancia están definidos en las ecuaciones (3) y (4).

 (3)

 (4)

De esta forma, la definición de diferentes límites de inicio, y diferentes deltas de distancia, generan un marco de dibujo diferente, obteniendo así un ‘*zoom*’ para el dibujo que se logra a partir de la matriz calculada. Realizando estos cambios se muestran los resultados conseguidos al realizar ‘*zoom*’ sobre diferentes secciones del dibujo de la figura (9).

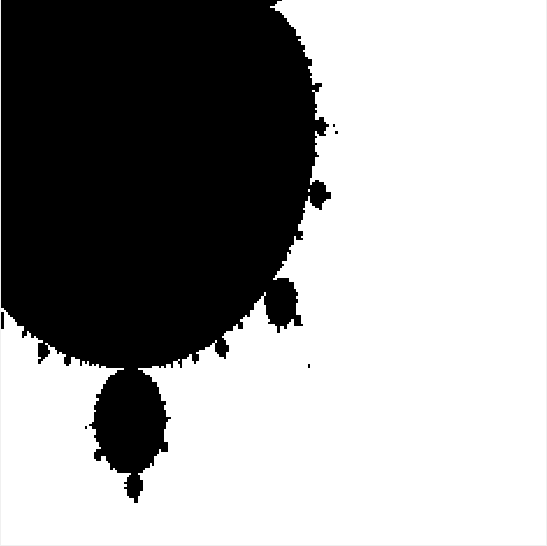
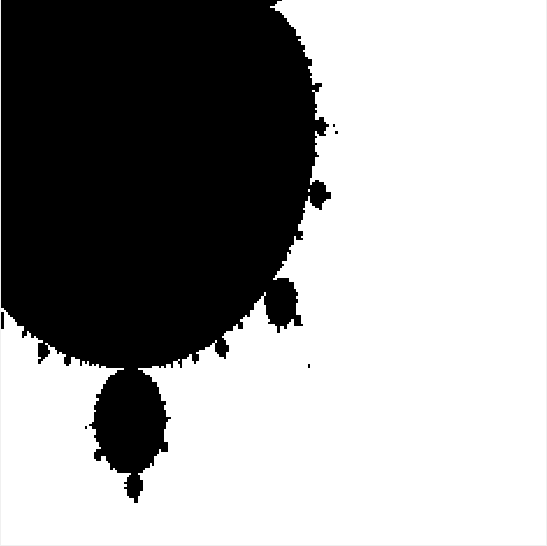
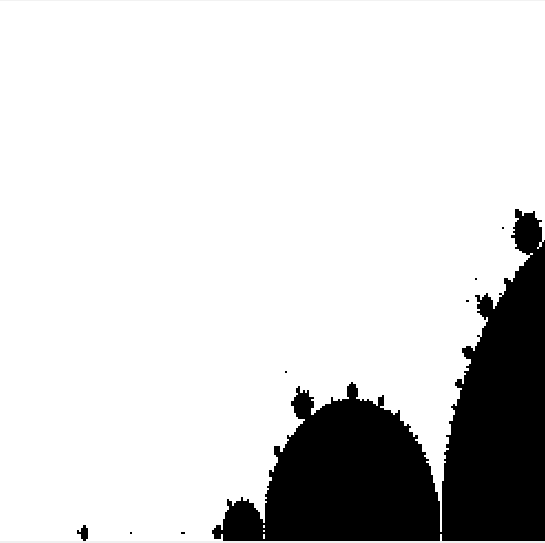
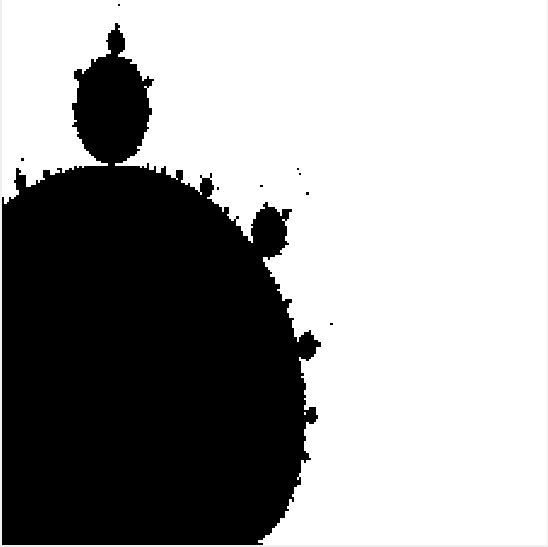
****

Figura 10. ‘*zoom*’ sobre el fractal de Mandelbrot.

Por medio del comando ‘*tic*’ ‘*toc*’ en Matlab, se puede realizar la medición del tiempo de ejecución del código para generar la matriz con diferentes límites de iteraciones para el cálculo de cada punto, los resultados obtenidos se muestran en la figura 11, que corresponde al tiempo de ejecución para 2000 iteraciones.



Figura 11. Medición del tiempo elapsado en la ejecución del código.

*Implementación en Hardware:*

Para el diseño se concibió inicialmente el diagrama de bloques del sistema, para el cual las entradas provienen de 4 interruptores y 2 pulsadores, y las salidas son las señales que controlan la pantalla *VGA*, es decir, hsync y vsync, junto con la información contenida cada 12 bits del pixel actual, y la señal de la comunicación serial donde se transmite el tiempo que toma el sistema en calcular el conjunto de Mandelbrot para 480x480 puntos con un número x de iteraciones que puede variarse.

Para la impresión del fractal se exigió una imagen de 480x480 pixeles, por lo que 80 pixeles hacia la derecha y hacia la izquierda de los marcos horizontales y verticales se dejaron en negro. El sistema se compone de un bloque para controlar el tiempo de la pantalla horizontal, y un bloque para controlar el tiempo de la pantalla vertical, junto con un bloque que se encarga de la medición del tiempo de cálculo del conjunto completo, un bloque que se encarga de transmitir la información de forma serial hacia un dispositivo externo TTL a USB, y un bloque de control del dibujo, que se encarga de generar los cálculos del conjunto completo de Mandelbrot y guardarlos en una memoria RAM, para posteriormente leerla e imprimir los datos codificados como pixeles. En este mismo bloque está contenida la acción de ‘zoom’ sobre el dibujo.

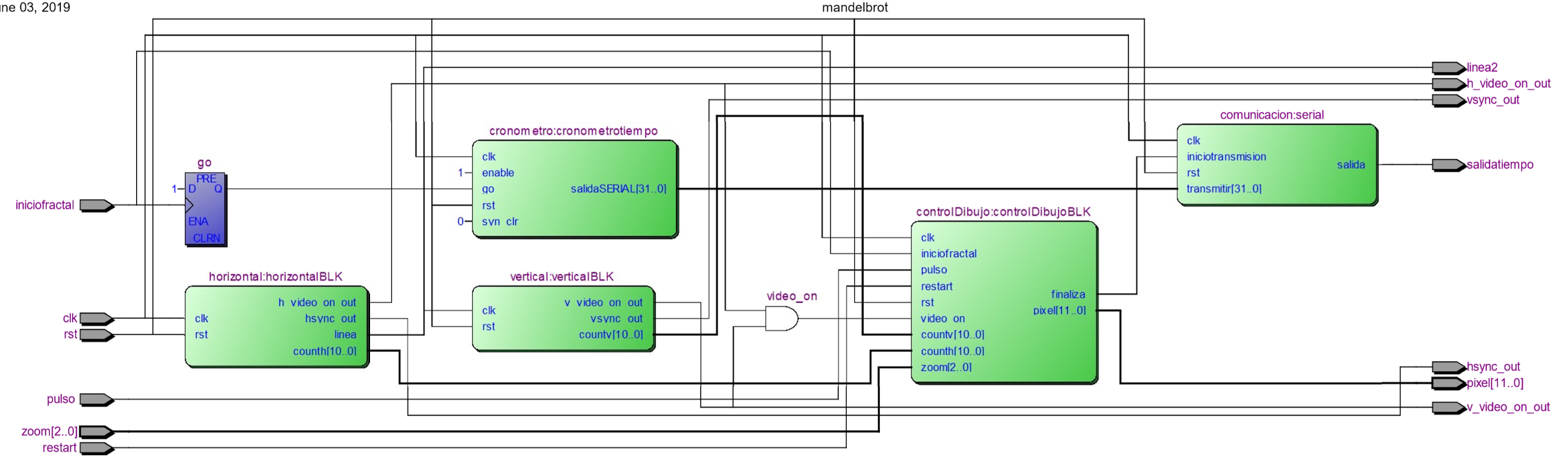


Fig. 12 . Diagrama en bloques general del sistema.

En la figura 12 aparece representado el diagrama de bloques del sistema realizado, para el cual ingresa la señal de reinicio mediante un interruptor, inmediatamente a esto se realiza el cálculo del tiempo y al final de calcular la matriz de la memoria, se envían los datos vía serial a un computador. A partir de tres interruptores se realiza zoom a los cuatro cuadrantes, lo cual se puede realizar dos veces.

Para controlar vsync se cuentan los pixeles de cada línea de forma que al contar los 640 + píxeles muertos, se cuenta la impresión de una línea completa, vsync funciona como un contador de líneas. los bloques Display 1 y 2 son los bloques encargados de realizar la conversión de binario a codificación en los 7 segmentos para cada uno de los dos dígitos correspondientes al puntaje que lleva cada jugador. Finalmente del Dibujo es de donde salen los bits que conciernen a la información del color del píxel que está siendo leído, de forma que se genera un vector con el color deseado en lugar de almacenar cada serie de 12 bits que corresponde a la información de cada píxel en una memoria.

A continuación en la figura 13 se presenta el diagrama de bloques referente al control horizontal de video para la pantalla *VGA*, el cual produce las señales que controlan el barrido horizontal de píxeles durante cada línea de las 480 líneas que se deben mostrar en la pantalla.

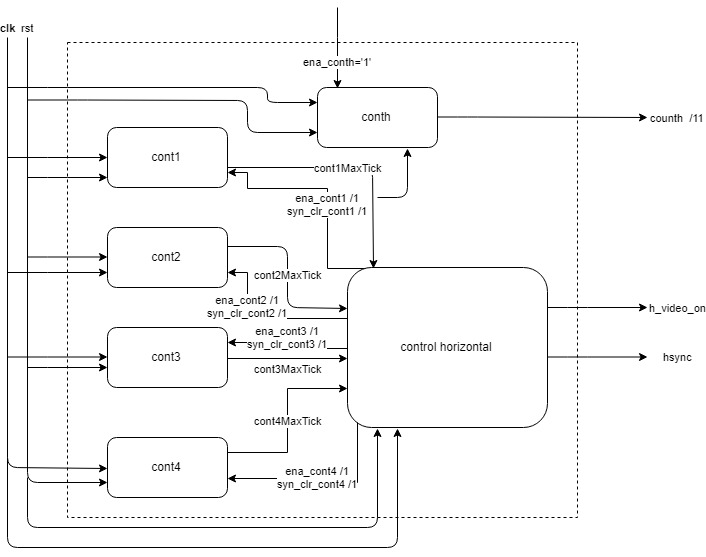


Fig. 13. Diagrama en bloques control horizontal.

En el control horizontal se tiene como entrada únicamente la señal de habilitación de los enable de los contadores horizontales. Teniendo en cuenta que el tiempo de habilitación de cada pixel es de dos ciclos de reloj al tener un reloj interno de la tarjeta de 20 MHz (20 ns), estos contadores cuentan ciclos de reloj, de forma que para contar el primer tiempo que ocurre cuando h\_video\_on está apagado pero h\_sync está prendido, al ser 48 pixeles del borde izquierdo, se cuenta hasta 48\*2=96. Para el siguiente tiempo en el cual se tienen estas dos señales en alto, se cuenta hasta 640\*2=1280. Para el tiempo que le sigue se cuenta hasta 16\*2=32 y finalmente hasta 96\*2=192. Al tener las mediciones exactas de cada uno de estos tiempos se pueden generar fácilmente las formas de onda necesarias para la correcta impresión del video. A continuación se presentan los tiempos sugeridos por el manual de la tarjeta para la generación de la señal hsync.

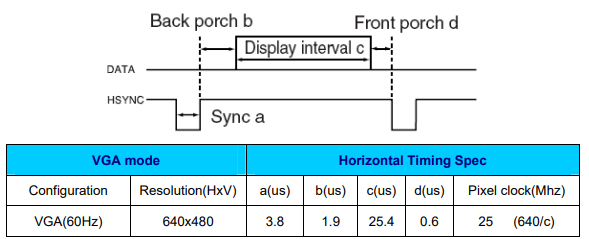


Fig. 9. Especificaciones de tiempo de la tarjeta para el control horizontal [3].

El manual de la tarjeta sugiere estos tiempos para las señales hsync y data o h\_video\_on, los cuales son bastante aproximados a los usados para estas señales. A continuación en la figura 10 se presenta el diagrama de bloques referente al control vertical de video para la pantalla *VGA*, que produce las señales que controlan el barrido vertical de líneas de pixeles durante las 480 líneas que se deben mostrar en la pantalla.

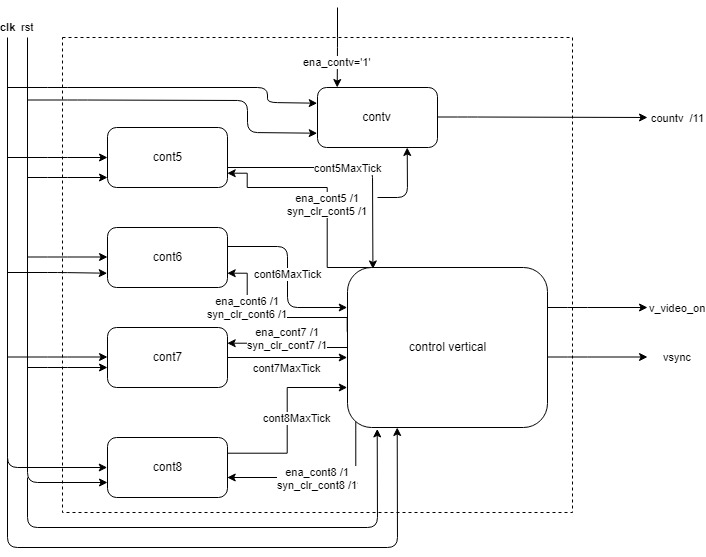


Fig.14 . Diagrama en bloques control vertical.

En el control vertical del mismo modo que en horizontal se tiene como entrada únicamente la señal de habilitación de los enable de los contadores verticales. Para este caso se tomaron las banderas como indicadores de conteos de líneas completas, de forma que se cuenta de igual manera que en el control horizontal, pero se usan las salidas de sus contadores para contar los tiempos de las líneas. Para los contadores presentes en estos bloques, al contar las líneas cuentan desde que ocurre cuando h\_video\_on está apagado pero h\_sync está prendido, las 33 líneas del borde superior, 480 líneas para cuando se tienen las dos señales en alto, 10 para el siguiente tiempo y 2 para el tiempo del retardo. De igual manera, a continuación se presentan los tiempos sugeridos por el manual de la tarjeta para la generación de la señal vsync.

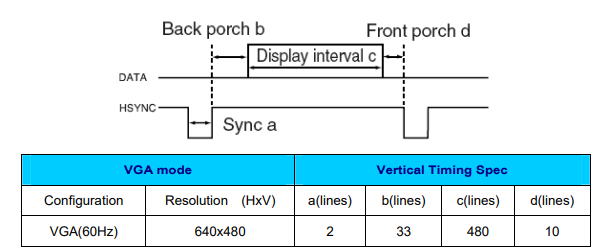
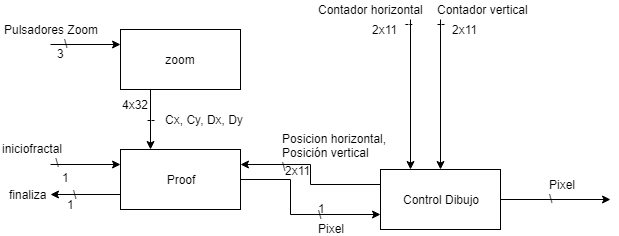


Fig. 11. Especificaciones de tiempo de la tarjeta para el control vertical [3].

El manual de la tarjeta sugiere estos tiempos para las señales vsync y data o v\_video\_on, los cuales son bastante aproximados a los usados para este tipo de señales. (se cuenta en términos de líneas horizontales de la pantalla *VGA*).



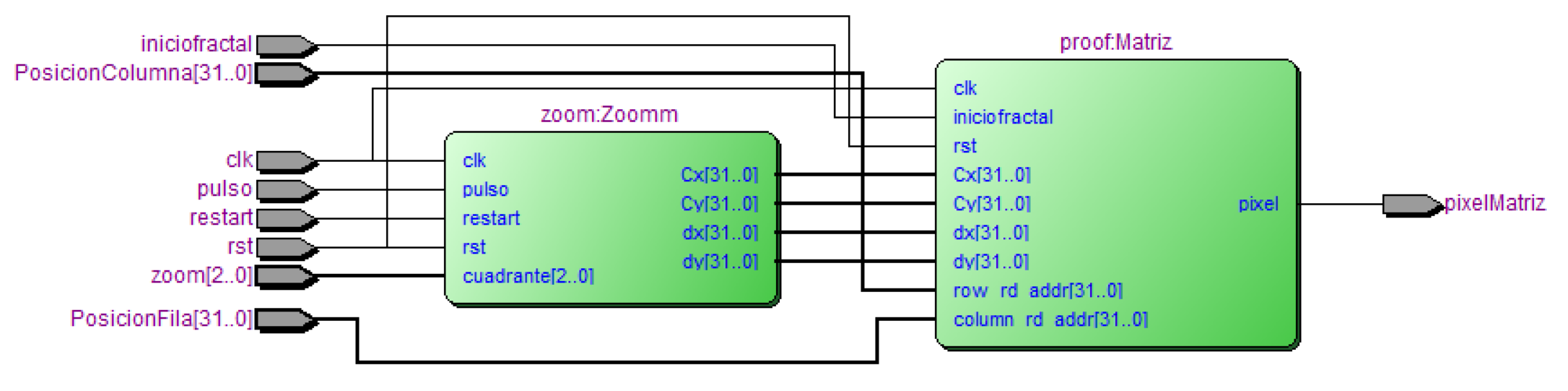


Fig.12 . Diagrama en bloques control dibujo.

En la figura 12 se presenta el diagrama de bloques de control dibujo, donde se controla todo el dibujo de la pantalla. Dentro de control dibujo se encuentra el bloque que genera la matriz, en el cual se encuentran los bloques que realizan el cálculo para cada píxel, la memoria RAM donde se almacenan los resultados obtenidos y el control que maneja este subsistema. De igual forma, se encuentra el bloque de zoom, donde se almacenan diferentes valores para un punto c y una distancia d determinada para realizar un acercamiento del fractal.

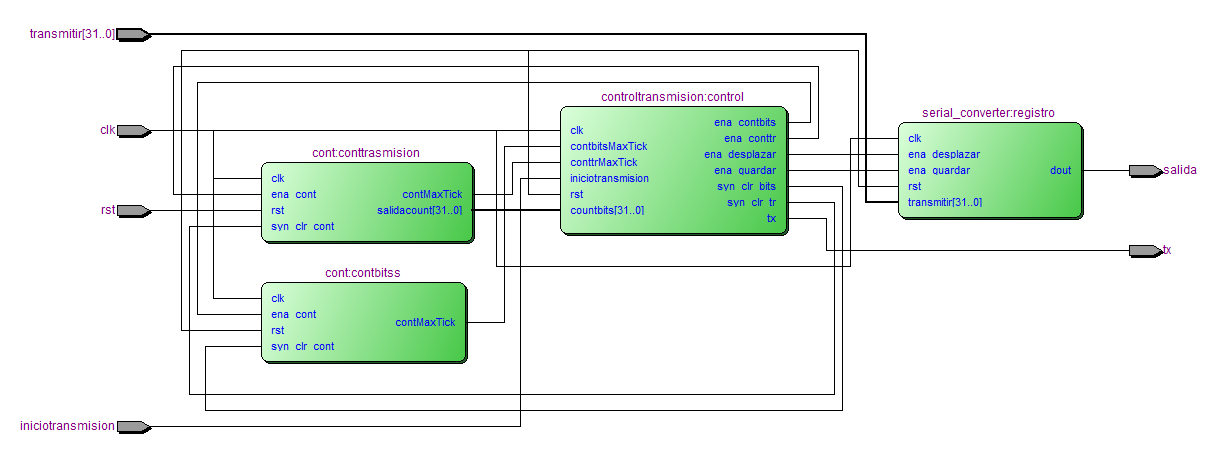


Fig.13 . Diagrama en bloques comunicación serial.

En la figura 13 se presenta el diagrama en bloques de la comunicación serial que consta de 4 bloques: un registro que convierte de paralelo a serie, un control de la transmisión y dos contadores, el primero que cuenta el tiempo de transmisión por bit para una velocidad de transmisión de 57600 Baudios (17.36 μs) y el segundo que cuenta el número de bits que se han transmitido. Esto es importante ya que se debe conocer cuando se termina de recibir un bit.

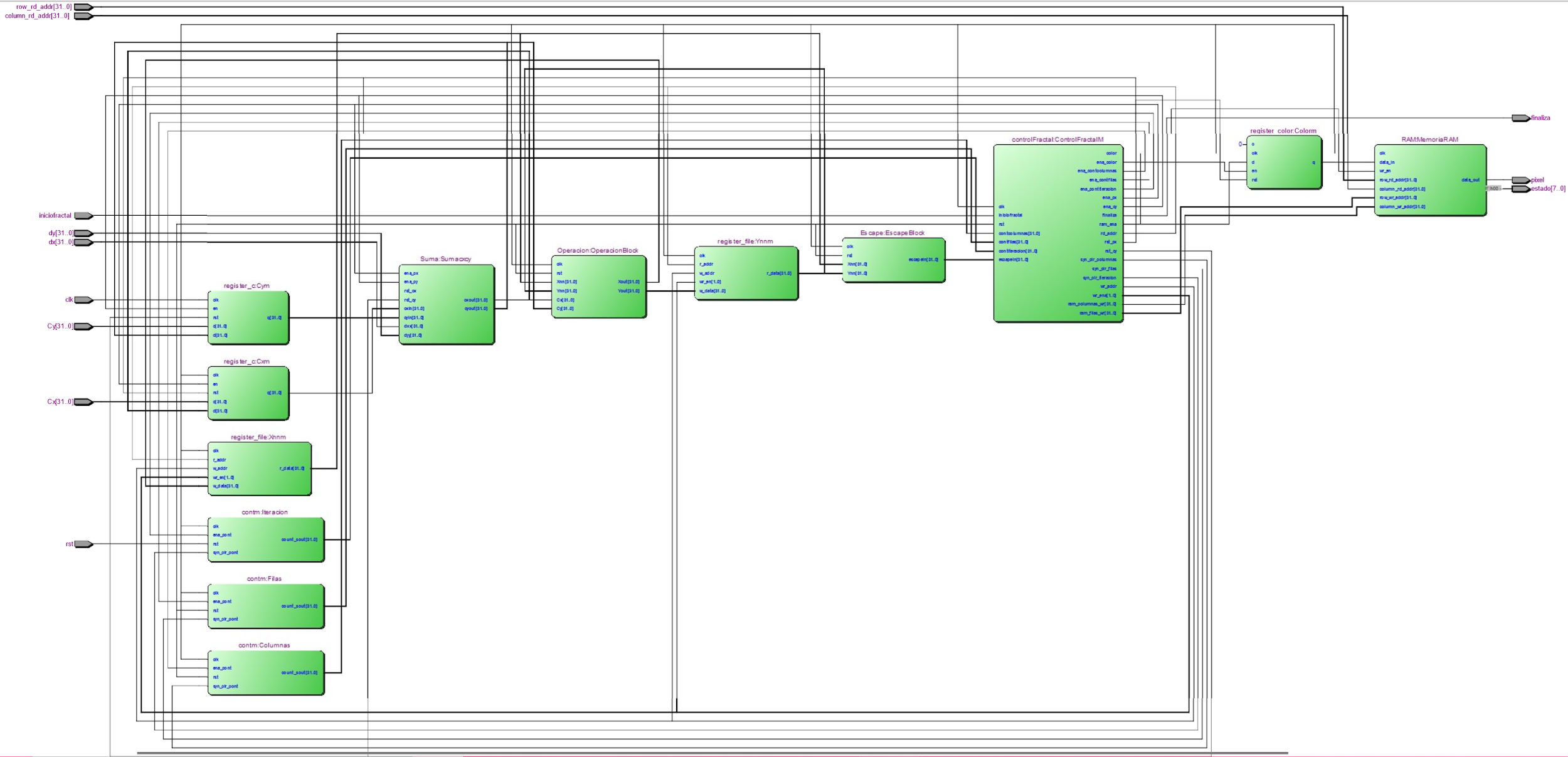


Fig.14 . Diagrama esquemático de bloques proof

En la figura 14 se presenta el diagrama en bloques del bloque proof en el cual se encuentran varios registros donde se van actualizando Cx o Cy, entre otras, además de contar con otros bloques como operación, suma, escape, etc.

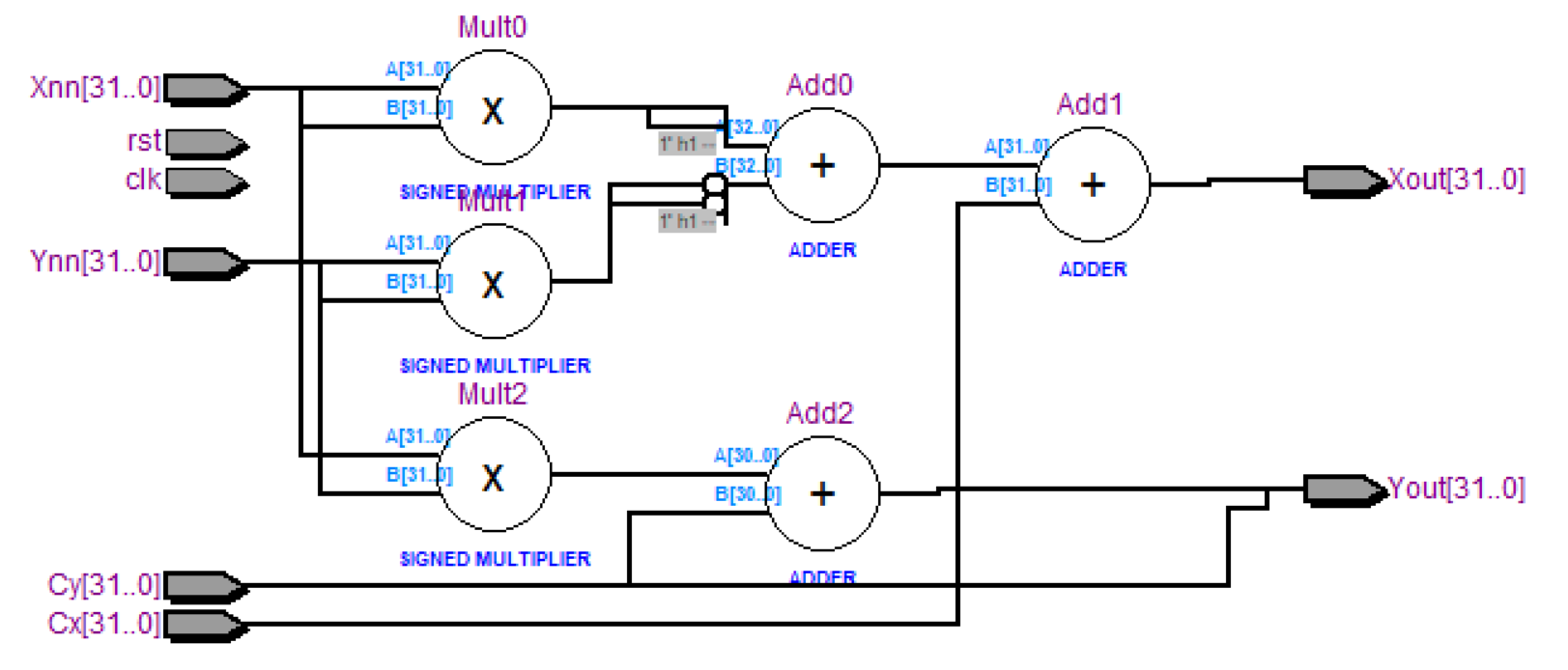


Fig.15 . Diagrama esquemático de operación

En el bloque de operación se realizan las operaciones que se realizan dentro de las iteraciones, de forma que solo contiene multiplicaciones y sumas, y se evidencia como solo se genera circuito combinatorio, de forma que simplemente se realizan multiplicaciones con desplazamiento y sumas.

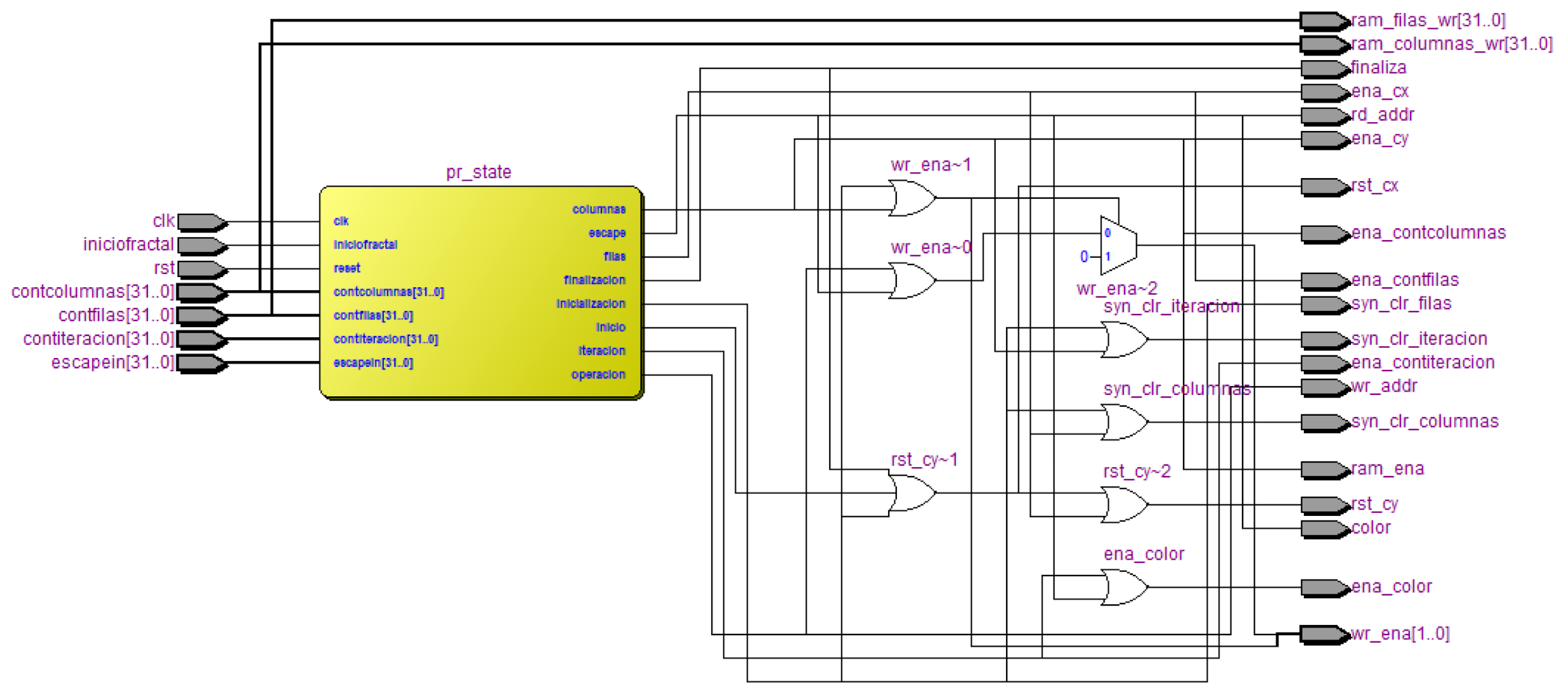


Fig.15 . Diagrama esquemático de control fractal.

En la figura 15 se presenta el diagrama en bloques de el control fractal en donde se pueden evidenciar todas las señales que salen del mismo a los otros bloques.

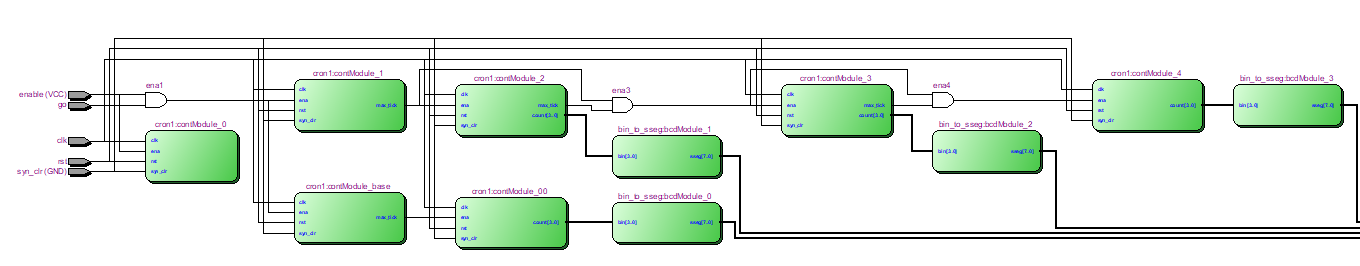


Fig.16 . Diagrama esquemático de bloques del cronómetro.

En la figura 16 se muestra el diagrama en bloques para el cronómetro en el cual se diseñó un sistema contador binario, el cual, cuenta inicialmente cada 100 ms, luego cuenta 10 veces del 0 al 9 esos 100 ms para corresponder al mili-segundero del cronómetro. Cada vez que pasan los 10 pulsos de 100 ms cuenta un segundo 10 veces, de 0 a 9 correspondiendo con el dígito del segundero, y finalmente cada vez que pasan los 10 pulsos de 1 s cuenta una décima de segundo 10 veces, de 0 a 9, correspondiendo con el dígito que lleva las décadas del cronómetro. Para esto se implementó el bloque ‘cont1’ el cual es un contador genérico que cuenta hasta un límite ‘con’ con un número de bits ‘n’, y se llamó dicho bloque varias veces con los valores correspondientes para cada una de estas 4 secciones. Posteriormente se creó el bloque ‘bin\_to\_sseg’ para convertir los cuatro dígitos codificados en código ASCII, cada uno de 8 bits.

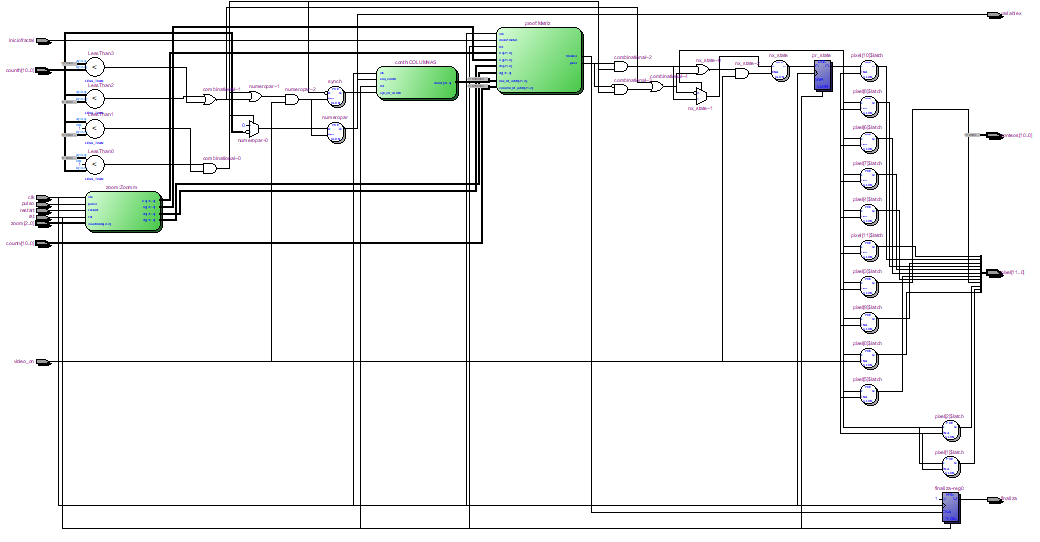


Fig.17 . Diagrama esquemático de bloque de control dibujo.

En el diagrama de la figura 17 se muestra el esquemático de control dibujo y si interacción con los bloques de proof y de zoom.

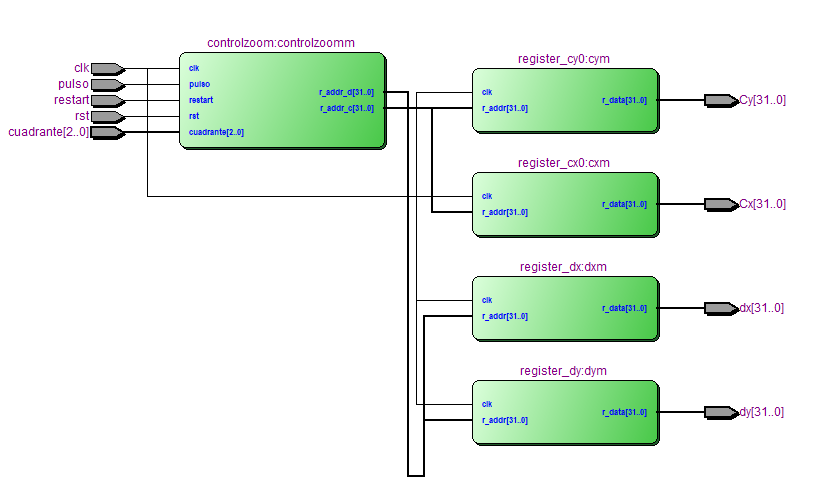


Fig.18 . Diagrama esquemático de bloque de zoom.

En el diagrama de la figura 18 se muestra el esquemático de bloques de zoom el cual está compuesto de controlzoom, donde se encuentra una ,equina de estados que me indica que dirección de memoria debo sacar dependiendo del valor de los interruptores de zoom, y varios registros de registros donde se almacenan los números y límites necesarios para que se muestre el zoom (C y d).

A continuación en las figura 19, 20 y 21 se muestran los esquemáticos de suma, escape y la memoria RAM los cuales se encuentran en el bloque proof, los registros que se encuentran dentro de este bloque varían respectivamente de lo que almacenan.

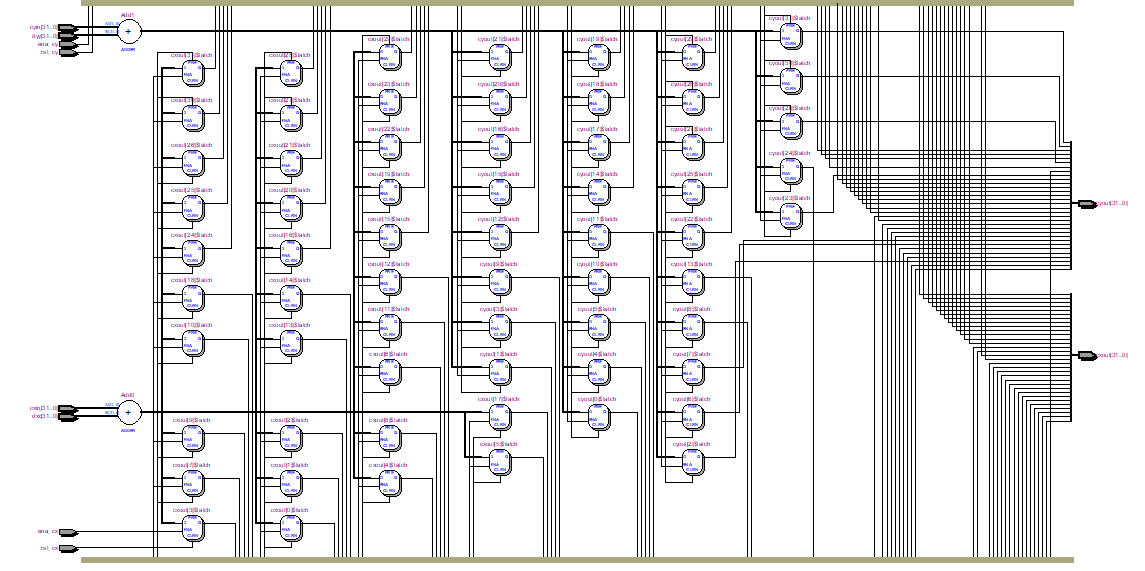


Fig.19 . Diagrama esquemático de suma.

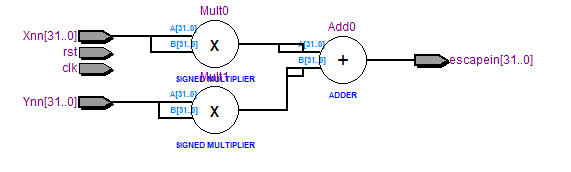


Fig.20 . Diagrama esquemático de escape.

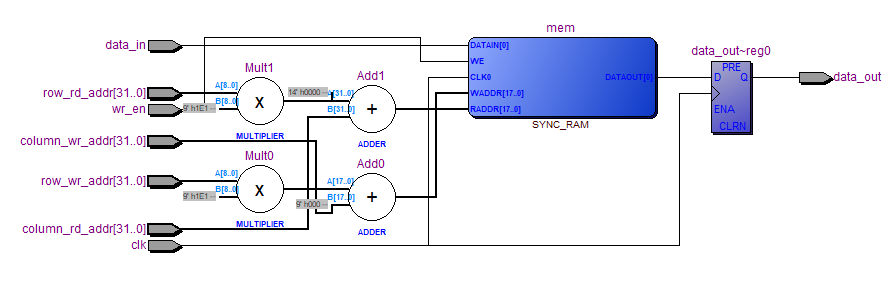


Fig.21 . Diagrama esquemático de memoria RAM.

A continuación en las figura 22, 23 y 24 se muestran los esquemáticos de registro paralelo a serie, contador y decodificador ASCII los cuales se encuentran en el bloque de comunicación serial, en todos los bloques y en el bloque de cronómetro respectivamente.

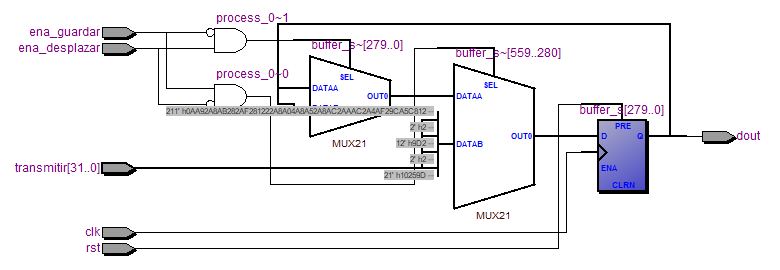


Fig.22 . Diagrama esquemático de registro paralelo a serie.

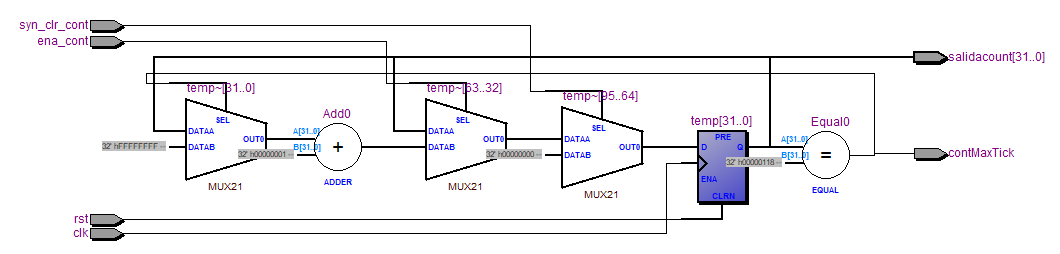


Fig.23 . Diagrama esquemático de contador.

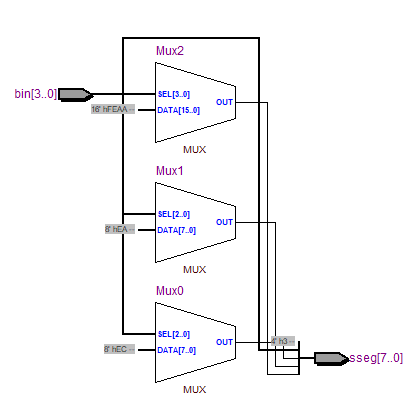


Fig.24 . Diagrama esquemático de codificador en código ASCII.

## *Máquinas de estados implementadas en el sistema*

Para el diseño del proyecto se planteó una máquina de estados principal llamada control dibujo que se encarga de seleccionar si el color del pixel a mandar es negro o blanco, dependiendo del valor de un contador de píxeles y uno de líneas. Esta máquina de estados dibuja un marco en el lado derecho e izquierdo de forma que la ventana de dibujo sea cuadrada, y mientras se encuentre leyendo los pixeles de esta ventana lee constantemente los valores de la matriz de la memoria *RAM* para la impresión de cada pixel existente dentro de la misma, de forma que se imprime un pixel en negro si el valor almacenado en la matriz en ‘0’ y un pixel blanco si el valor almacenado en la matriz es ‘1’.

Dentro de la máquina de estados principal, el manejo del cálculo de las posiciones de la matriz que define si se saca un pixel blanco o un píxel negro se realiza dentro del bloque proof, al cual, le llegan los valores iniciales de los límites inferiores y la distancia mediante el bloque zoom, donde cada uno de los mismos es manejado por una máquina de estados. Adicional a estos, el bloque de comunicación a su vez posee su propia máquina de estados la cual se encarga de la producción de una comunicación serial entre la tarjeta y el computador.

El bloque de cronómetro no maneja ninguna máquina de estados, simplemente inicia con la señal de ‘iniciofractal’ y está activa durante los conteos hasta que llegue la señal ‘finaliza’.

Máquina de estados de los controles vertical y horizontal

Los controles horizontal y vertical manejan los tiempo de inicio de los contadores utilizados para generar las señales v\_video\_on, h\_video\_on, v\_sync y h\_sync.

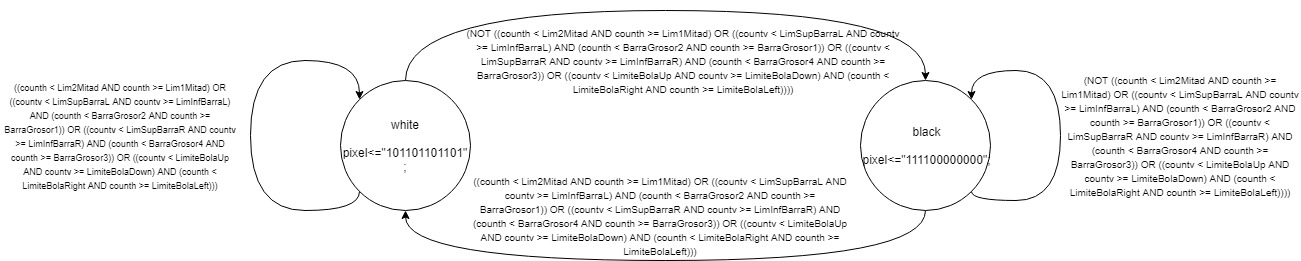


Fig. 25 . Diagrama de máquina de estados control dibujo.

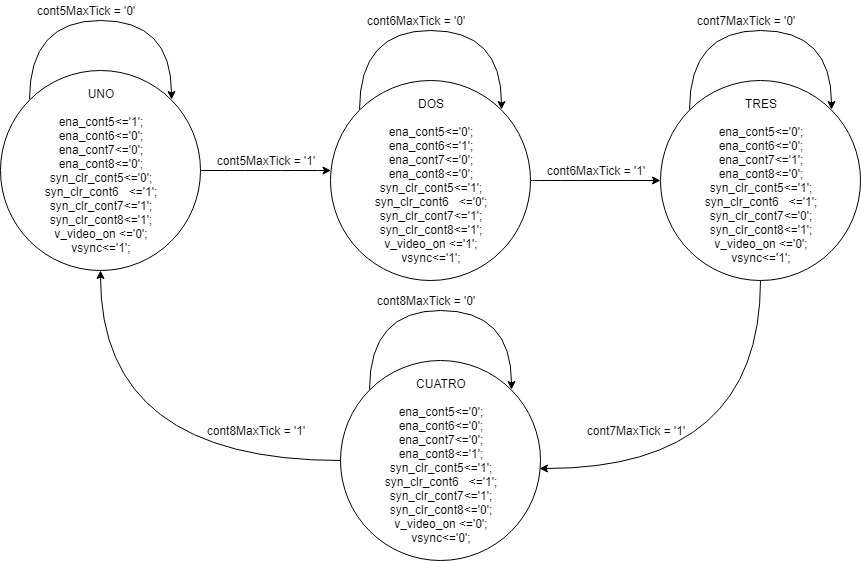


Fig. 26 . Diagrama de máquina de estados control vertical.

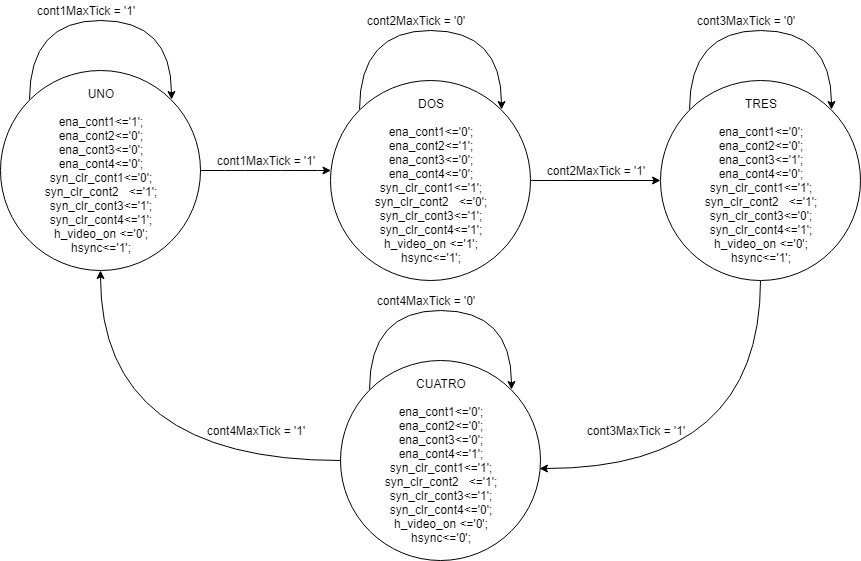


Fig.27 . Diagrama de máquina de estados control horizontal.

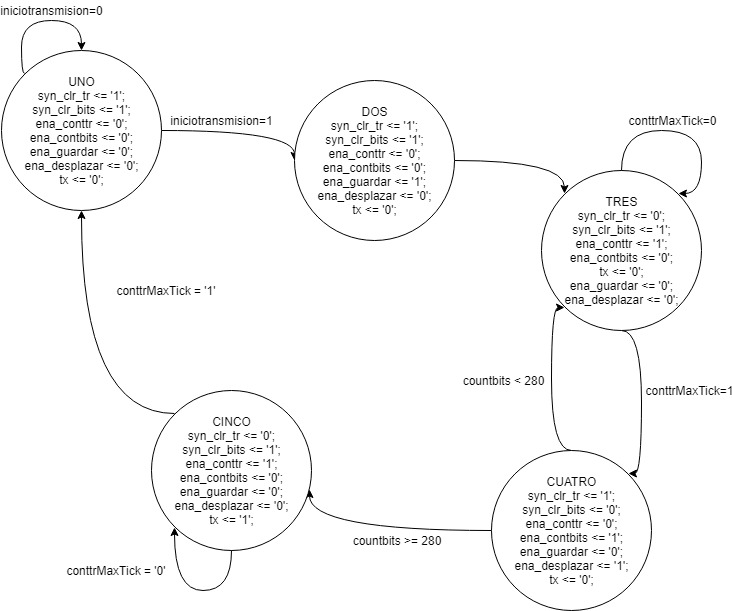


Fig.28 . Diagrama de máquina de estados comunicación serial.

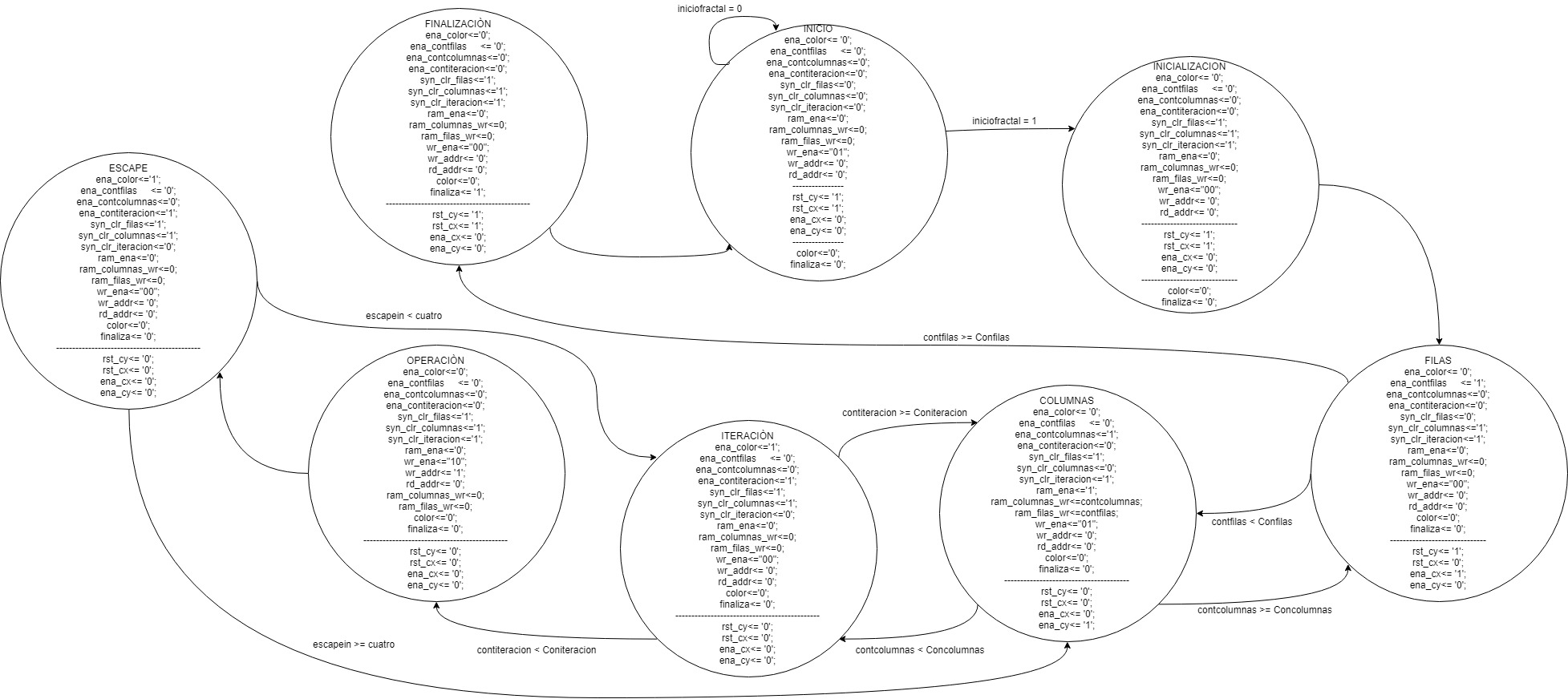


Fig.29 . Diagrama de máquina de estados control fractal.

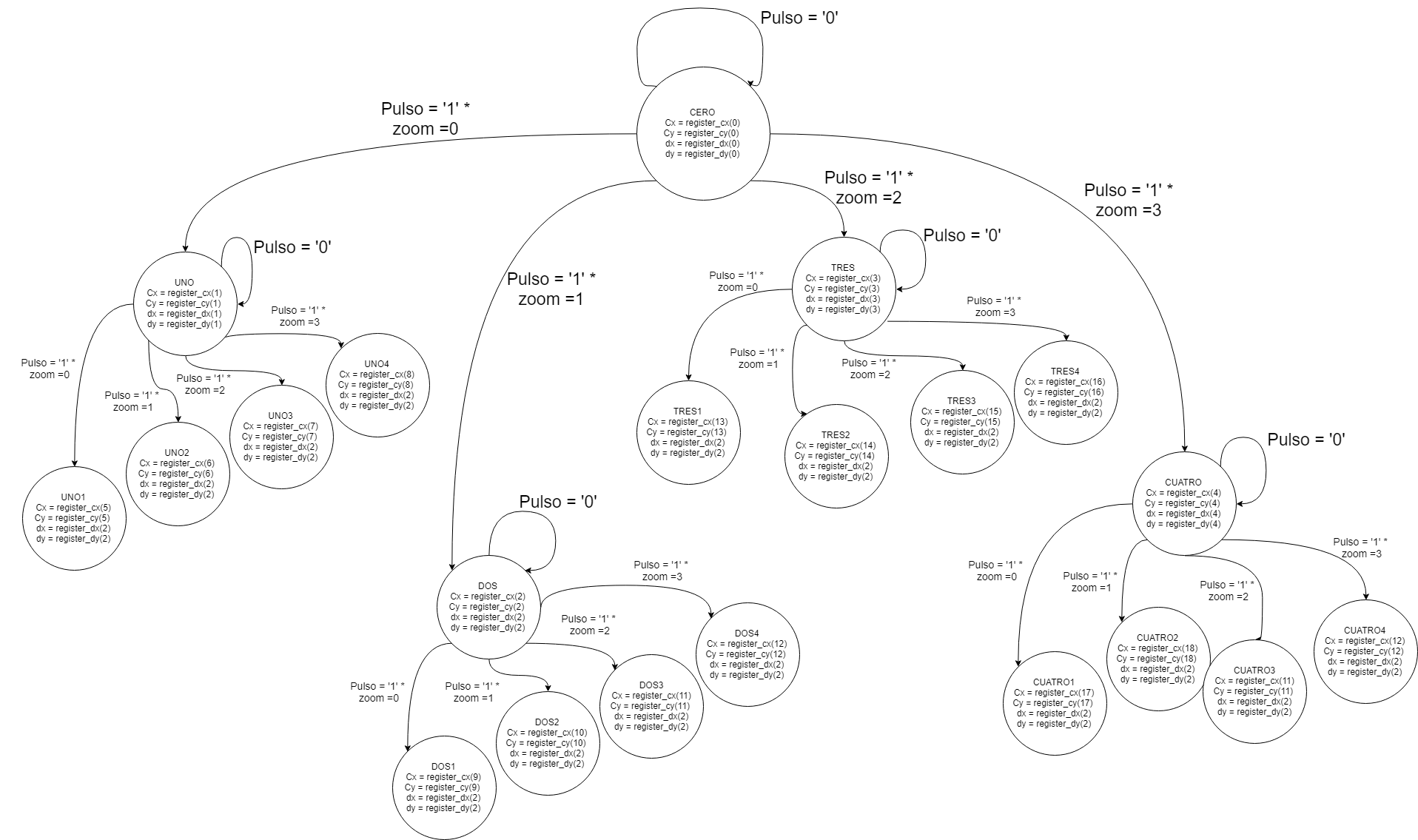


Fig.30 . Diagrama de máquina de estados control zoom.

## *Descripción de Estados*

Diagrama de máquina de estados control dibujo:

* white: El estado white funciona para poner en blanco el píxel deseado, para esto se miran con las condiciones el contador vertical y horizontal y se comparan con los límites del marco, si el contador horizontal se encuentra afuera de los marcos, se evalúa si la matriz donde se contiene la información del color del pixel indica la no pertenencia al conjunto de Mandelbrot con un ‘1’, y en este caso se pinta de blanco este pixel, y se queda en este estado mientras se encuentre dentro del marco de dibujo y el pixel a dibujar sea blanco.
* black: El estado black funciona para poner en negro el píxel deseado, para esto se miran con las condiciones el contador vertical y horizontal y se comparan con los límites del marco; si el contador horizontal se encuentra dentro de los marcos o si al evaluar la matriz dentro del marco, el color del pixel indica la pertenencia al conjunto de Mandelbrot con un ‘0’, se pinta de negro este píxel, y se queda en este estado mientras estas dos condiciones se cumplan.

Diagrama de máquina de estados control vertical:

* UNO: El estado uno activa el contador 5 que cuenta hasta 32 e inhabilita los demás contadores. Pasa al estado dos cuando la bandera cont5MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* DOS: El estado dos activa el contador 6 que cuenta hasta 479 e inhabilita los demás contadores. Pasa al estado tres cuando la bandera cont6MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* TRES: El estado tres activa el contador 7 que cuenta hasta 9 e inhabilita los demás contadores. Pasa al estado cuatro cuando la bandera cont7MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* CUATRO: El estado uno activa el contador 8 que cuenta hasta 1 e inhabilita los demás contadores. Pasa al estado uno cuando la bandera cont8MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.

Diagrama de máquina de estados control horizontal:

* UNO: El estado uno activa el contador 1 que cuenta hasta 96 e inhabilita los demás contadores. Pasa al estado dos cuando la bandera cont1MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* DOS: El estado dos activa el contador 2 que cuenta hasta 1280 e inhabilita los demás contadores. Pasa al estado tres cuando la bandera cont2MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* TRES: El estado tres activa el contador 3 que cuenta hasta 32 e inhabilita los demás contadores. Pasa al estado cuatro cuando la bandera cont3MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.
* CUATRO: El estado uno activa el contador 4 que cuenta hasta 192 e inhabilita los demás contadores. Pasa al estado uno cuando la bandera cont4MaxTick se pone en ‘1’, lo cual indica que el contador terminó de contar.

Diagrama de máquina de estados control comunicación serial:

* UNO: El estado uno espera a que la señal iniciotransmisión (iniciofractal) este en ‘1’ para pasar al estado dos, de lo contrario se queda en el estado uno.
* DOS: En el estado dos se activa el señal de enable que hacer que se guarde en el registro que convierte de paralelo a serie ( se guardan los bits provenientes del cronómetro). Pasa enseguida al estado tres.
* TRES: En el estado tres se activa el contador de tiempo de transmisión por bit de aproximadamente 17.2 μs y se recibe una señal cuando termina de contar este tiempo, apenas termina de contar se pasa al estado cuatro.
* CUATRO: En el estado cuatro se activa el contador de bits y si este es menor a 280 vuelve al estado tres, si ya llegó a los 280 pasa al estado cinco.
* CINCO: En el estado cinco se vuelve a contar el tiempo de transmisión por bit y cuando termina vuelve al estado uno.

Diagrama de máquina de estados control fractal:

* INICIO: El estado uno espera a que la señal iniciofractal este en ‘1’ para pasar al estado inicialización, de lo contrario se queda en el estado inicio.
* INICIALIZACIÓN: En el estado inicialización se pasa directamente a filas y se inicializan las variables.
* FILAS: En el estado filas se pasa al estado columnas si no se han contado todas las filas, o al estado finalización si ya se contaron todas las filas.
* COLUMNAS: En el estado columnas se pasa al estado iteración si no se han contado todas las columnas, o al estado filas si ya se contaron todas las columnas.
* ITERACIÓN: En el estado iteración se pasa al estado operación si no se han contado todas las iteraciones, o al estado columnas si ya se contaron todas las iteraciones.
* OPERACIÓN: En el estado operación se pasa directamente al estado escape.
* ESCAPE: En el estado escape se pasa al estado columnas si ya se llegó al número cuatro, de lo contrario pasa a iteración.
* FINALIZACIÓN: En el estado finalización pasa directamente al estado inicio.

Diagrama de máquina de estados control zoom:

* CERO: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 0 donde se encuentran los números para no realizar zoom.
* UNO: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 1 donde se encuentran los números para realizar zoom en el primer cuadrante de la pantalla.
* DOS: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 2 donde se encuentran los números para realizar zoom en el segundo cuadrante de la pantalla.
* TRES: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 3 donde se encuentran los números para realizar zoom en el tercer cuadrante de la pantalla.
* CUATRO: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 4 donde se encuentran los números para realizar zoom en el cuarto cuadrante de la pantalla.
* UNO1: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 5 donde se encuentran los números para realizar zoom en el primer cuadrante de la pantalla.
* UNO2: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 6 donde se encuentran los números para realizar zoom en el segundo cuadrante de la pantalla.
* UNO3: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 7 donde se encuentran los números para realizar zoom en el tercer cuadrante de la pantalla.
* UNO4: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 8 donde se encuentran los números para realizar zoom en el cuarto cuadrante de la pantalla.
* DOS1: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 9 donde se encuentran los números para realizar zoom en el primer cuadrante de la pantalla.
* DOS2: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 10 donde se encuentran los números para realizar zoom en el segundo cuadrante de la pantalla.
* DOS3: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 11 donde se encuentran los números para realizar zoom en el tercer cuadrante de la pantalla.
* DOS4: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 12 donde se encuentran los números para realizar zoom en el cuarto cuadrante de la pantalla.
* TRES1: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 13 donde se encuentran los números para realizar zoom en el primer cuadrante de la pantalla.
* TRES2: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 14 donde se encuentran los números para realizar zoom en el segundo cuadrante de la pantalla.
* TRES3: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 15 donde se encuentran los números para realizar zoom en el tercer cuadrante de la pantalla.
* TRES4: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 16 donde se encuentran los números para realizar zoom en el cuarto cuadrante de la pantalla.
* CUATRO1: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 17 donde se encuentran los números para realizar zoom en el primer cuadrante de la pantalla.
* CUATRO2: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 18 donde se encuentran los números para realizar zoom en el segundo cuadrante de la pantalla.
* CUATRO3: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 19 donde se encuentran los números para realizar zoom en el tercer cuadrante de la pantalla.
* CUATRO4: En este estado las salidas Cx y Cy toman los valores de la memoria en la posición 20 donde se encuentran los números para realizar zoom en el cuarto cuadrante de la pantalla.

# Pruebas de Funcionamiento

Para verificar el funcionamiento se realizaron pruebas de los bloques principales como las máquinas de estado, los contadores no se simularon, pues las señales provenientes de ellos eran utilizadas por otros bloques como cálculo para poder mostrar la imagen.

Para verificar el funcionamiento de los bloques de operación y los registros Xnn y Ynn se revisó en si se realizaba el desplazamiento entre las posiciones (0) que representa el el valor anterior de X es decir Xn-1 y el valor actual almacenado en Xnn (1), de igual forma para el registro de Y. De igual forma, se verificaron los estados en los cuales cambiaba el valor de operación (cálculo realizado con el valor actual), actualizando el valor actual de x en el estado operación y el valor anterior en el estado de escape.

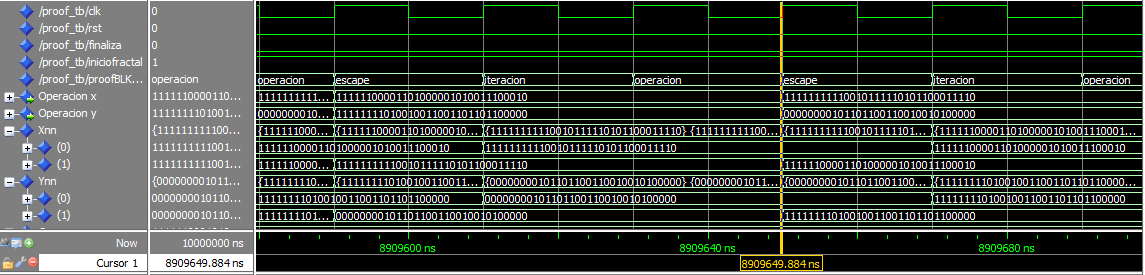


Fig.31 . Simulación

Para corroborar el funcionamiento de los registros que almacenan C y la suma, se verificó en qué estado cambiaba el valor del registro y si este se actualizaba. El valor de registro de Cx debía cambiar en el estado de filas y el registro Cy en el estado de columnas.

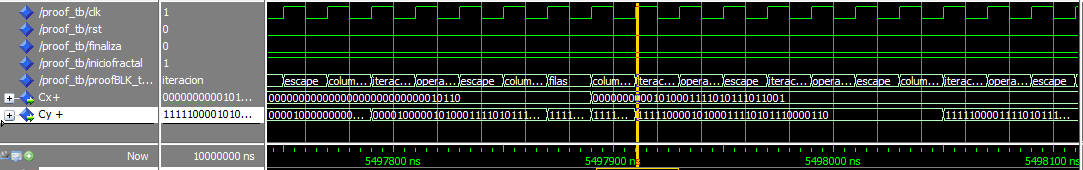


Fig.32 . Simulación

Se verificó que el bloque de cálculo generará la señal de finalización para la implementación del cálculo del tiempo

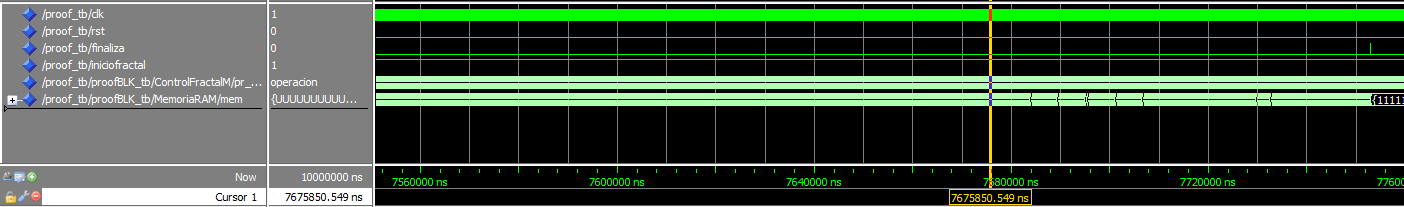


Fig.33 . Simulación

Para verificar el cálculo y las señales enviadas al bloque de VGA, se redujo el tamaño de la matriz a 100x100 posiciones con 100 iteraciones y se verificaron los valores con la matriz en *software* del mismo tamaño e iteraciones.

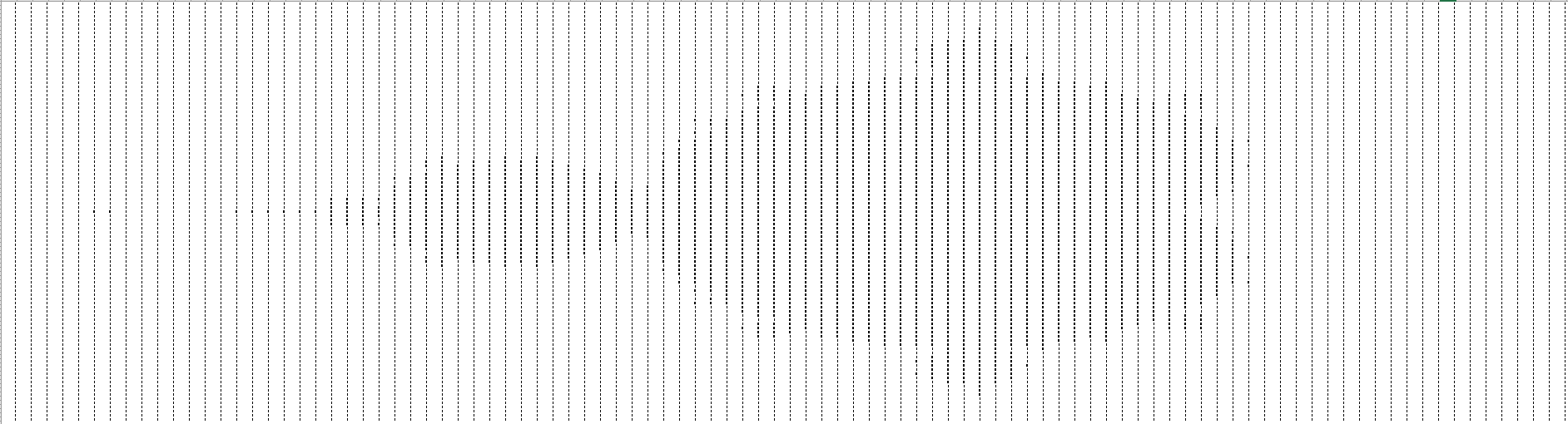


Fig.34 . Simulación

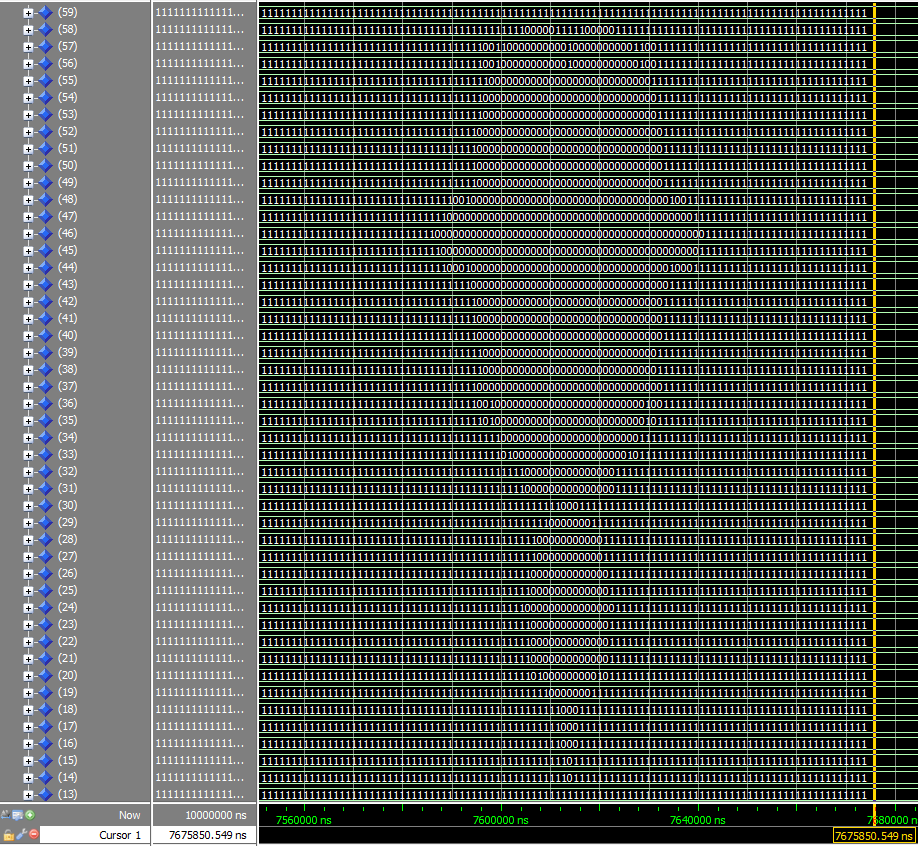


Fig.35 . Simulación

*Resultados hardware:*

Los resultados obtenidos en *hardware* se muestran a continuación en la figura X.

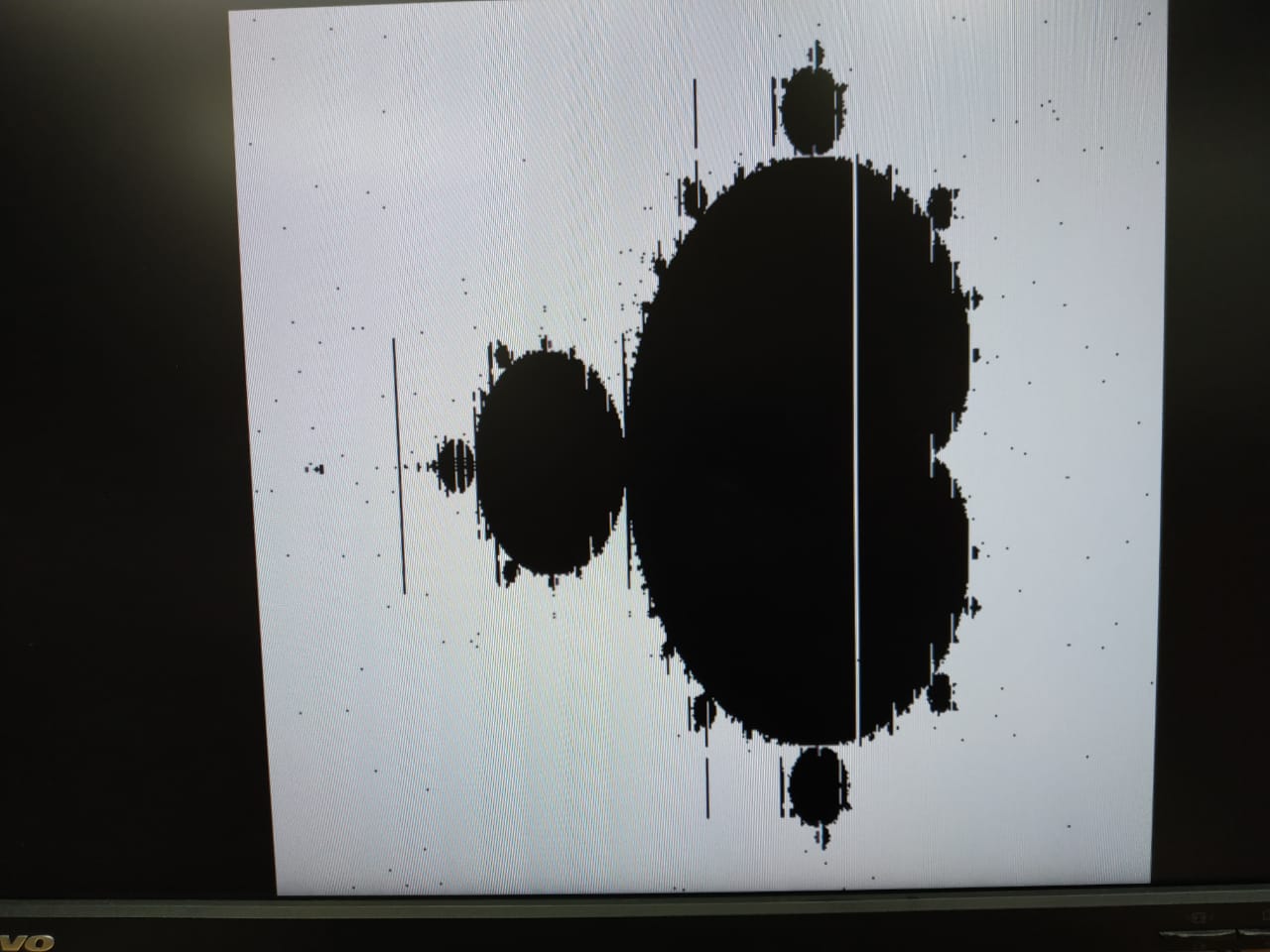
**

Figura 36. Resultados de la implementación del proyecto.

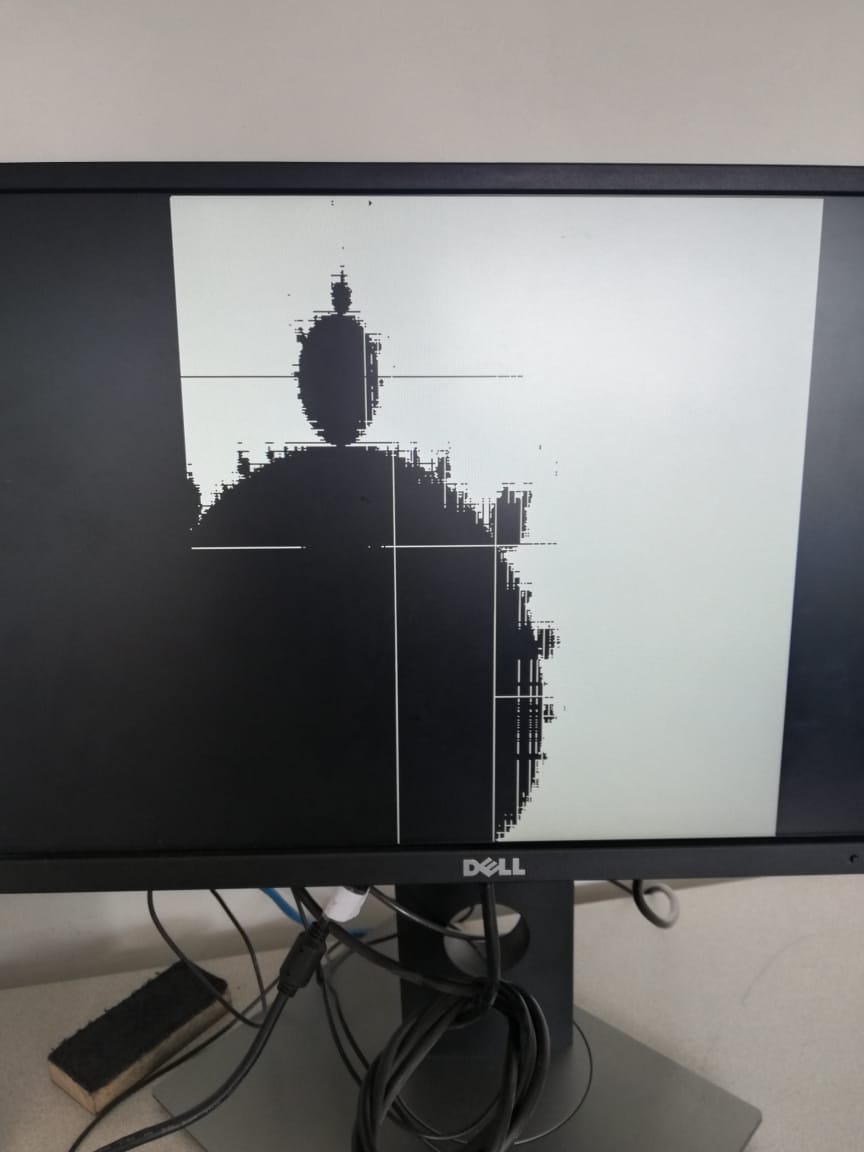
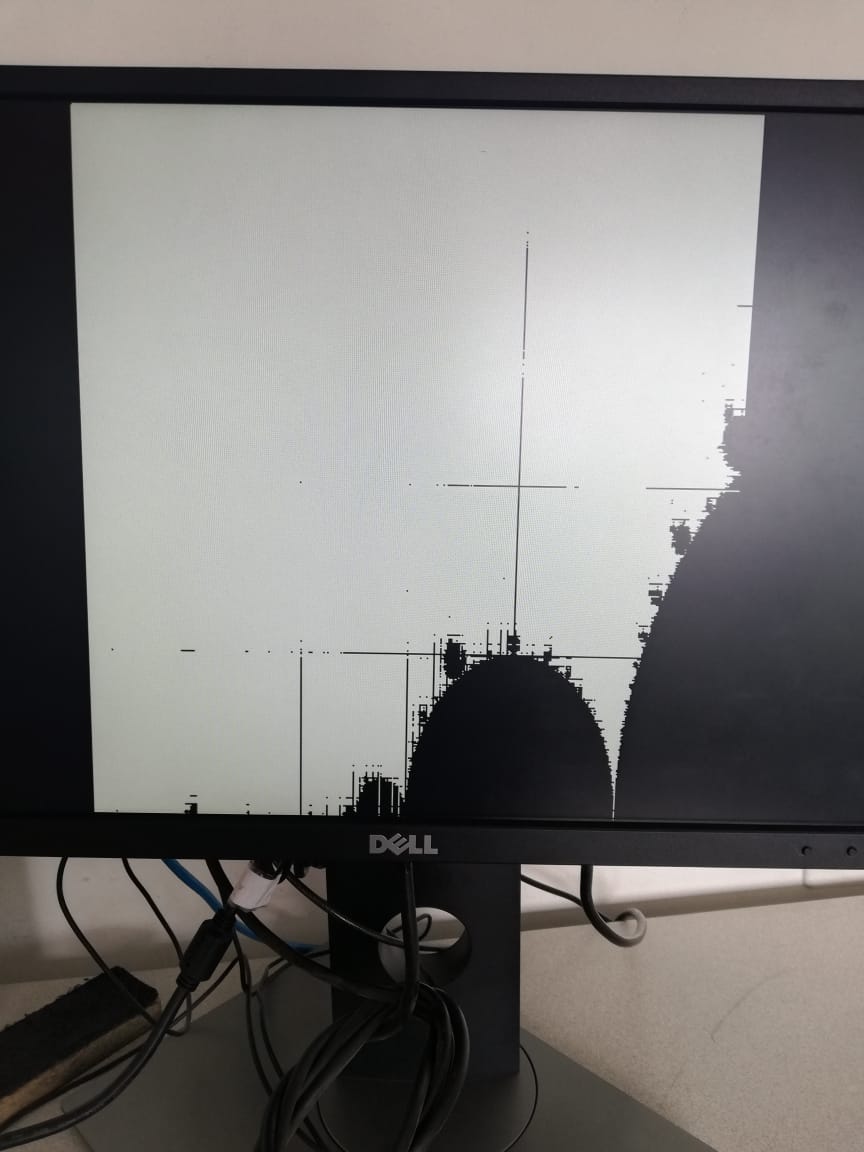
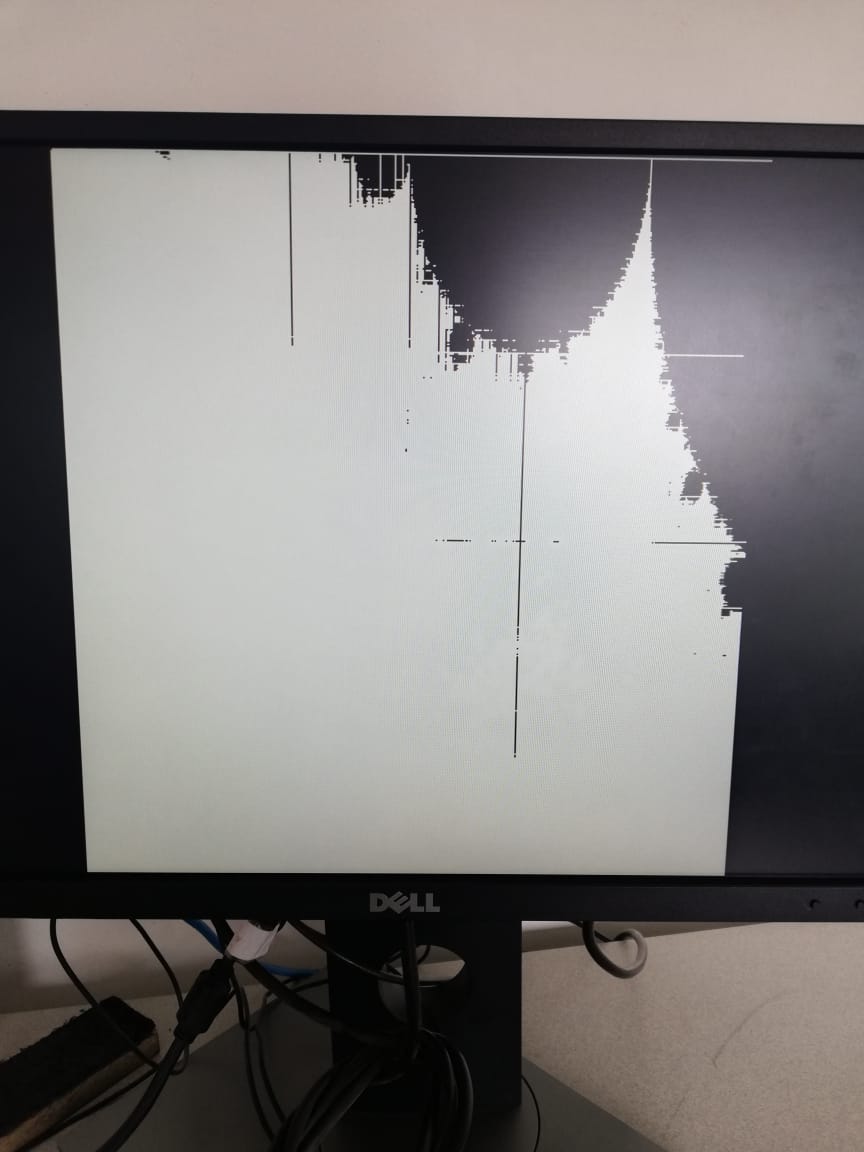
**

Figura 37 . ‘*zoom*’ sobre el fractal de Mandelbrot en Hardware.

En la figura 38 se tienen los resultados del tiempo empleado para calcular la matriz.



Figura 38. Resultados de tiempo obtenidos vía comunicación serial.

# Conclusiones

Para la realización de este proyecto se realizó inicialmente una concepción del problema, con el fin de interiorizar conceptos y entender los objetivos que se deben cumplir en el desarrollo del mismo para su funcionamiento, posterior a esta etapa se empezó a diseñar el sistema con una metodología *up-down* en la cual inicialmente se concibió el sistema como una serie de bloques de funciones específicas, y al plantear el funcionamiento del sistema, fue posible sintetizarlo en una máquina de estados, así como los demás bloques que también requerían de una. A partir de estos cimientos se desarrolló el código en lenguaje *VHDL* en el programa Quartus II 13.0 y posteriormente a una fase de simulaciones se asignaron las entradas a los pines del puerto VGA de salida y finalmente se visualizó el resultado en la pantalla. Al implementar esta metodología de diseño se logró cumplir con los objetivos y lograr los resultados deseados de forma ordenada y óptima.

Para la visualización de una imágen en una pantalla por medio del puerto *VGA* fue necesario tener en cuenta las frecuencias aceptadas por el mismo monitor, y el nivel de sincronismo necesario entre las señales de sincronismo vertical y horizontal, de forma que se acató la recomendación encontrada en el manual de la tarjeta de sincronizar el eje horizontal con el tiempo, pero el eje vertical con cada línea del eje horizontal, y de esta forma se pudo visualizar correctamente la pantalla en el monitor.

Al hacer un comparativo entre el tiempo de cálculo de la matriz de Mandelbrot tanto en software como en hardware, por ejemplo con un número de iteraciones de 2000, en el software utilizando Matlab) el tiempo de ejecución fue de 28.95 segundos aproximadamente, mientras que con las mismas iteraciones el tiempo de ejecución en hardware fue de 7.44 segundos aproximadamente, por lo tanto se puede concluir que el hardware es mucho más eficiente que el software en cuanto a términos de tiempo y velocidad se refiere, pues es evidente que el tiempo varía mucho en la realización de lo mismo. Claramente al usar el hardware (y en este caso una fpga) se pueden realizar varias actividades en paralelo, lo que disminuye el tiempo, mientras que en software se va haciendo una función a la vez.

##### Referencias

1. Mandelbrot, Benoît: «Fractal aspects of the iteration for complex» artículo en la revista Annals of the New York Academy of Sciences, 357, págs. 249-259.
2. E. Gerlein, “Project No. 2 Proyecto Final: Generador de Fractales de Mandelbrot ”. 2019.
3. E. Gerlein, “Project No. 1 Digital design in fpga: PONG”. 2019.
4. P. Sánchez, "Se cumplen treinta años del nacimiento de 'Pong', el primer juego electrónico", El País, 2002. [Online]. Available: https://elpais.com/diario/2002/09/19/ciberpais/1032400944\_850215.html.[Accessed: 02- Apr- 2019].
5. ALTERA, “DE0 USER MANUAL”. terasIC, pp. 34-36. [Online]. Available: http://esca.korea.ac.kr/teaching/FPGA\_boards/DE0/DE0\_User\_Manual.pdf.[Accessed: 02- Apr- 2019].