*PONG*

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 el presente documento se presenta la concepción, diseño e implementación de una versión propia del juego “*pong”*, basado en el deporte de tenis de mesa o *ping pong*, el cual es para 2 jugadores y consiste en que uno de los jugadores controla en el juego una paleta moviéndola verticalmente en el extremo izquierdo de la pantalla, mientras el otro jugador controla una segunda paleta en el extremo derecho. Los jugadores usan las paletas para pegarle a la pelota de forma que objetivo consiste en que uno de los jugadores consiga más puntos que el oponente al finalizar el juego. Estos puntos se obtienen cuando el jugador adversario falla al devolver la pelota. El proyecto fue implementado en *VHDL* por medio del programa *Quartus II 13.0*, utilizando la tarjeta de desarrollo de *ALTERA DE0 Cyclone III EP3C16F484C6*, el juego se diseñó de tal forma que puede visualizarse desde cualquier pantalla con entrada *VGA,* por medio de un cable *VGA-VGA* el cual se conecta directamente desde la tarjeta a la pantalla. Para los controles de los jugadores se utilizaron *Joysticks* y el puntaje se muestra en 2 de los *display LED* 7 segmentos de la tarjeta.**

***Keywords—VHDL, VGA, joystick, jerarquía, máquina de estados, diagrama de bloques, simulación, pixel, frame.***

# Introducción

En el presente documento se presenta la metodología seguida para el desarrollo del juego “*pong*”, para el cual a continuación se tratan los referentes históricos del juego en el marco teórico, se exponen las especificaciones del mismo junto con los requerimientos establecidos para el desarrollo del presente proyecto, sobre la forma de mostrar imágenes en una pantalla por medio de un conector *VGA-VGA* utilizando la *FPGA* de *Altera*.

En la siguiente sección se presenta la concepción del diseño en diagrama de bloques y posteriormente en máquina de estados finitos de la arquitectura del juego, detallando cada uno de los mismos. Seguido aesto se presentan las pruebas de funcionamiento de los resultados obtenidos del proyecto, con sus respectivas simulaciones. También se presentan pantallazos de una implementación en una pantalla y de adjunta el link del video de la misma presentación del juego resultante. A continuación se exponen las conclusiones del proyecto en una sección y las referencias utilizadas para el desarrollo del proyecto en la última sección.

# Marco Teórico

Pong es reconocido por ser el primer videojuego de la historia, el cual fue creado por un físico llamado William Higinbotham en 1958, e inició como el sistema doméstico Magnavox Odyssey de Ralph Baer. El juego nació como una versión bidimensional virtual del tenis de mesa o *ping pong*, como el primer videojuego de la historia [1][2]. Originalmente fue construido con fines recreativos por la compañía Atari copiado del sistema anteriormente mencionado en 1970, compañía que posteriormente lanzó la versión doméstica del juego en consola[1][2]. El juego se desarrollaba sobre una pantalla negra dividida a la mitad por una línea punteada blanca, y dos paletas cortas gruesas a cada extremo lateral de la pantalla, con las cuales se golpeaba un pequeño cuadrado que se desplazaba de un lado a otro al golpear alguna de las mismas paletas o chocar con los límites superiores o inferiores del espacio[2]. Los jugadores controlaban las paletas por medio de una pequeña rueda y se anotaba un punto cuando el contrincante no podía responder provocando que el cuadrado saliera fuera de los límites[2]. El primer jugador que juntara 11 puntos ganaba[2].

La primera versión doméstica de Pong de Atari fue rechazada por los jugueteros al no vender lo suficiente, sin embargo el fundador de la compañía, Nolan Bushnell, convenció a Tom Quinn de la empresa Sears para que lo distribuyese bajo la marca Sears Tele-Games, vendiendo más de 150.000 unidades en poco tiempo[2]. Posteriormente Atari retomó el control de su producto lanzando versiones cada año, con diferentes variantes como versiones para cuatro jugadores[2]. Finalmente la compañía sucumbió con la entrada de la competencia que plagió el sistema del juego y al tener que pagar 700.000 dólares de la patente sobre 'juegos de pelota y pala' al creador legítimo del mismo[2].

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 1, así como la asignación de los 12 bits que definen el color de cada pixel, los cuales corresponden a los valores mostrados en la figura 2, el funcionamiento de cada entrada del conector *VGA* ilustrado en la figura 3, y el funcionamiento del controlador VGA.

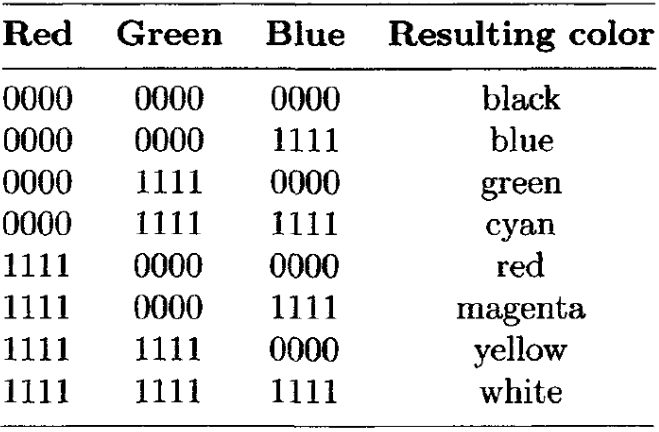
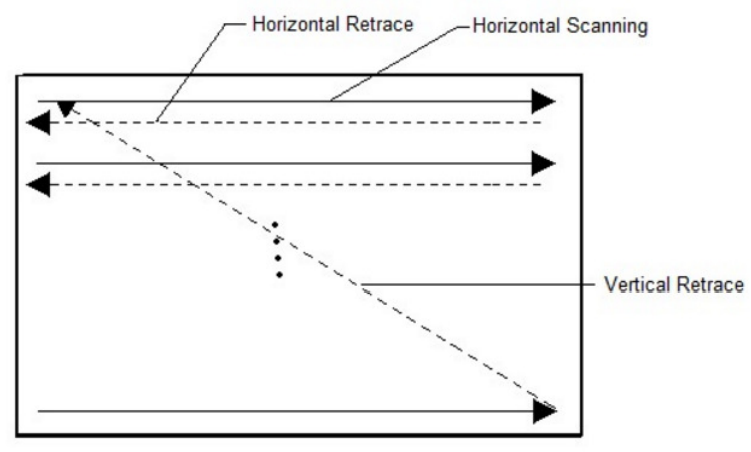


Figura 1. señales del VGA[1]. Figura 2. Color CRT [1].

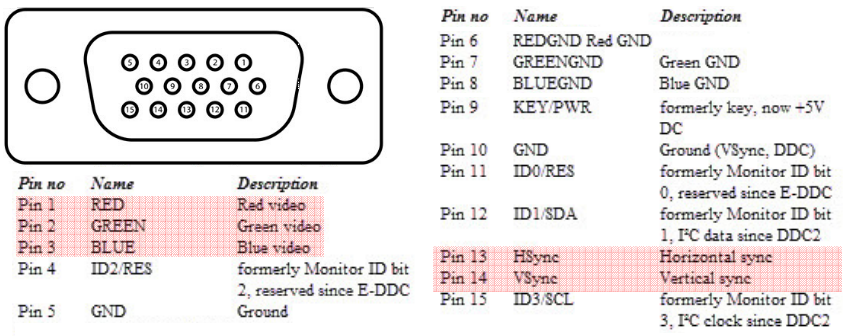


Fig. 3. Conector VGA [1].

El conector *VGA* maneja principalmente 5 señales; *hsync, vsync, red, blue* y *green*[1]. 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.

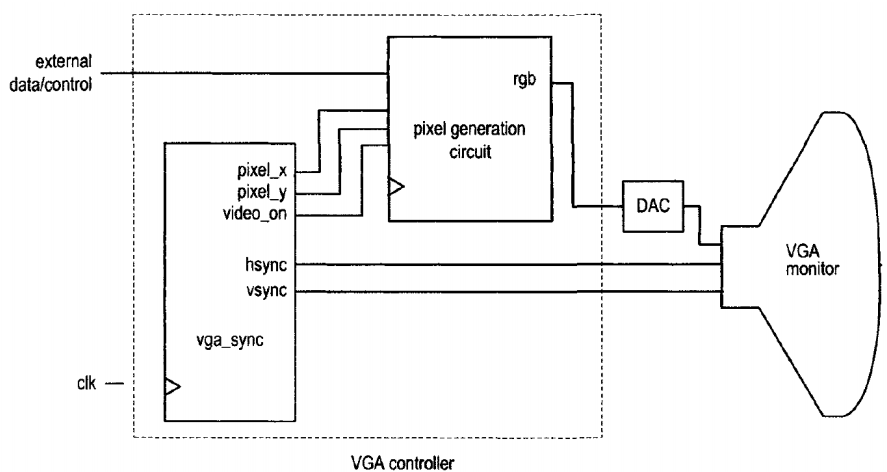


Fig.4. Controlador VGA [1].

Para el controlador del monitor *VGA* se utiliza un controlador 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 respectivo, cada 40 ns se realiza la lectura e impresión de cada píxel en cada línea.

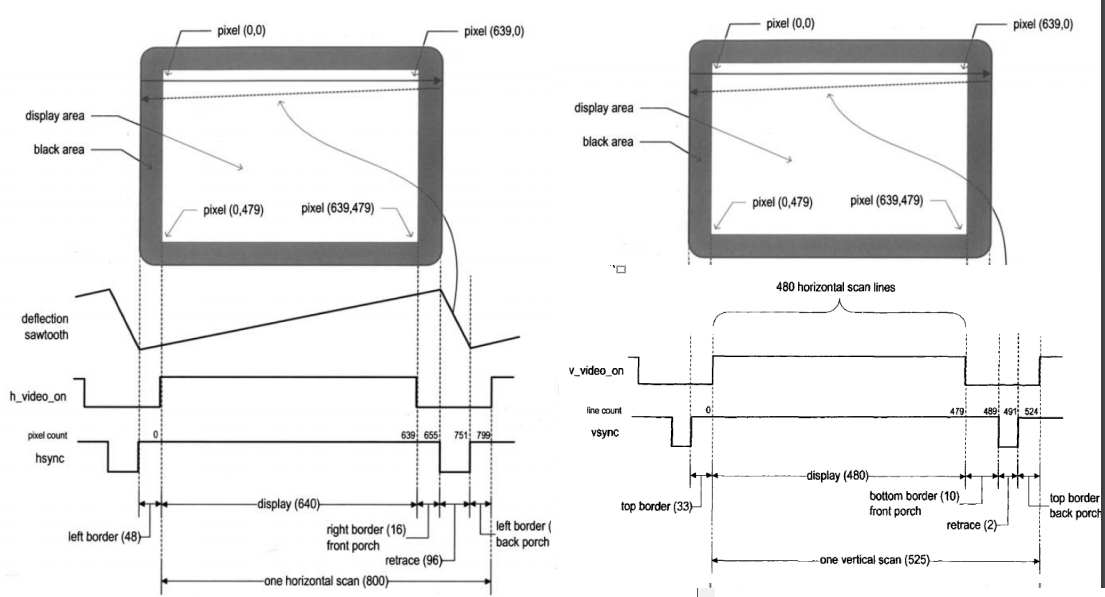


Fig.5. Señales controlador horizontal VGA [1]. Fig.6.Señales controlador vertical VGA [1]

Como se puede apreciar en las figuras 5 y 6, la pantalla completa ocupa un espacio de tiempo entre cada píxel y cada línea, de forma que no se podrá ver la imágen 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 restan 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

Para el diseño se concibió inicialmente el diagrama de bloques del sistema, para el cual las entradas provienen de 2 Joystick, 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 las señales que controlan los dos displays *LED* siete segmentos.

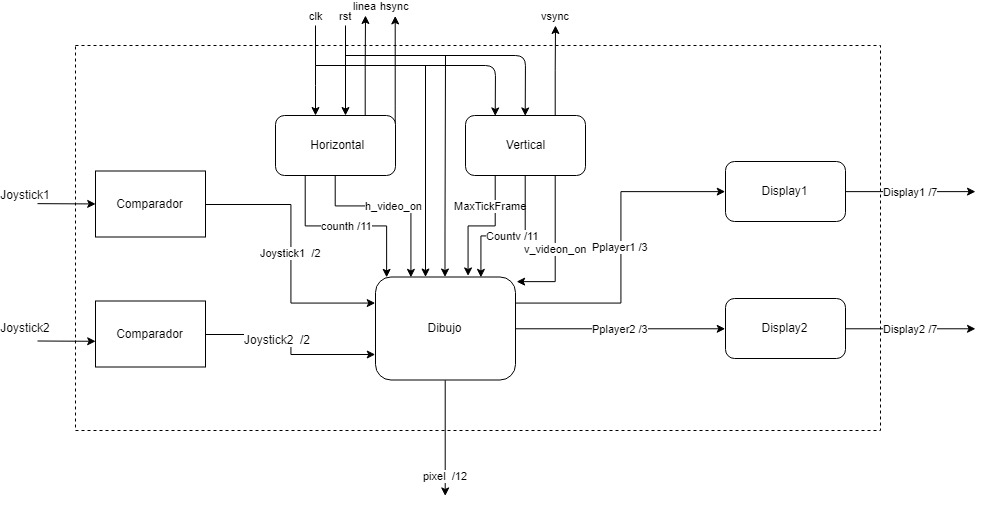


Fig. 7 . Diagrama en bloques general del sistema.

En la figura 7 aparece representado el diagrama de bloques del sistema realizado, para el cual ingresan dos entradas análogas provenientes de los *joystick*, las cuales se procesan por medio de comparadores para obtener 3 estados y salen 2 bits de cada bloque comparador, de forma que al tener movimiento hacia arriba se tiene “00”, hacia abajo se tiene “11” y en el centro se tiene “10” o “01” . Estas señales son procesadas en el bloque de dibujo, el cual contiene los controles de las dos barras y el control de la bolita. Adicionalmente maneja los bloques horizontal y vertical, de los cuales se generan las señales hsync y vsync.

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 el 7 segmentos para cada uno de los dos dígitos correspondientes al puntaje que lleva cada jugador. Finalmente de 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 8 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.

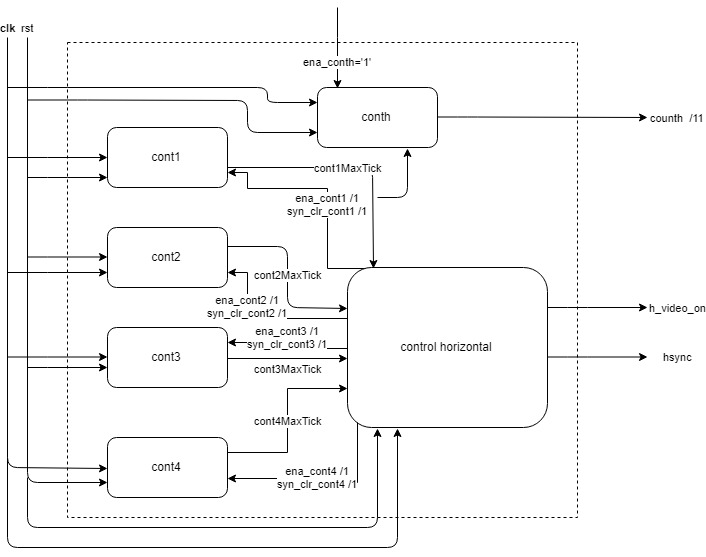


Fig. 8. Diagrama en bloques control horizontal.

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.

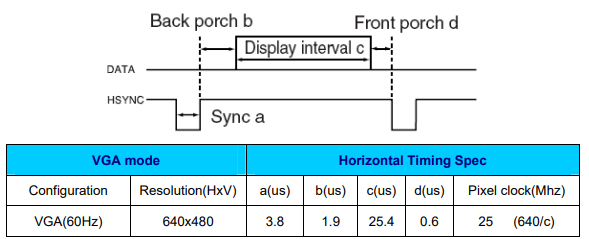


Fig. 9. Especificaciones de tiempo de la tarjeta para el control horizontal [3].

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*, el cual produce las señales que controlan el barrido vertical de líneas de pixeles durante las 480 líneas que se deben mostrar en la pantalla.

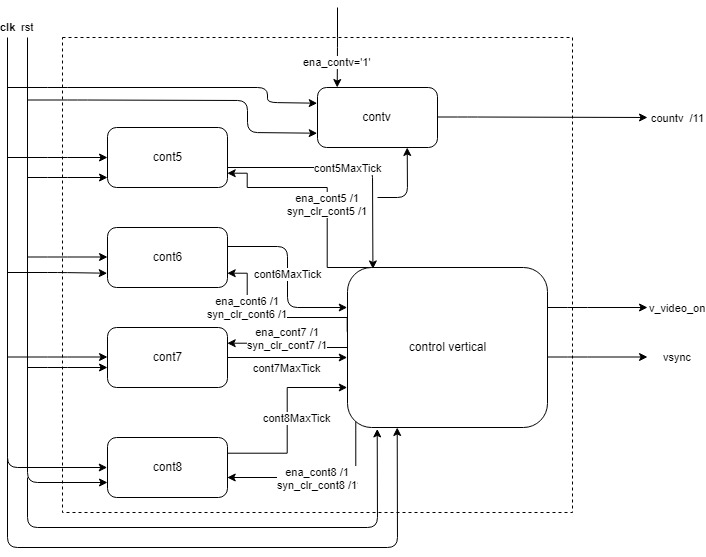


Fig.10 . Diagrama en bloques control vertical.

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.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.

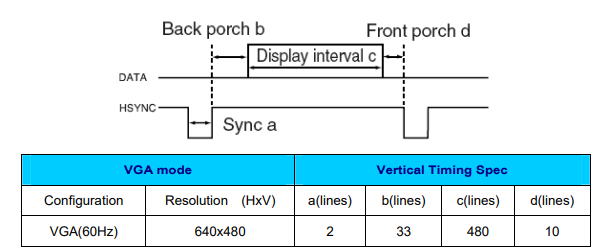


Fig. 11. Especificaciones de tiempo de la tarjeta para el control vertical [3].

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 estas señales. (se cuenta en términos de líneas horizontales de la pantalla *VGA*).

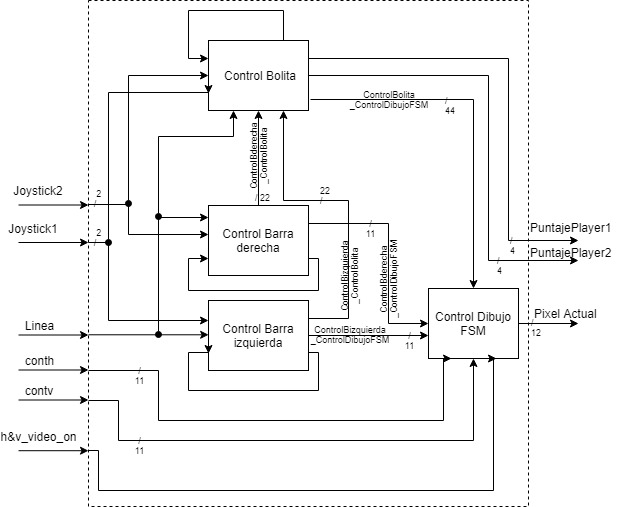


Fig.12 . Diagrama en bloques control vertical.

En la figura 12 se presenta el diagrama de bloques de control dibujo, en el cual se controla todo el dibujo de la pantalla, por esto dentro de control dibujo se encuentra el control de las dos barras/paletas junto con el control de la bolita. En el planteamiento de las barras las dos paletas son controladas de tal forma que poseen grosores fijos y alturas fijas, pero que únicamente se desplazan verticalmente hasta los límites superior e inferior de la pantalla dependiendo de la posición que indica el joystick de cada una de las anteriores, ambos controles deben tener como entrada sus posiciones anteriores para su correcto funcionamiento.

En el planteamiento de la bolita se debían conocer las posiciones de ambas paletas para poder definir todos los límites dentro de los cuales la bolita sigue funcionando normalmente,y por consiguiente los casos cuando no lo esté y provoque que un jugador gane un punto, también es necesario conocer las posiciones de ambos joystick para dar inicio a su movimiento cuando cualquiera de los dos cambie su posición del centro. De igual manera el control de la bolita le llega como entrada su posición anterior. Para el control del dibujo se realizó una máquina de estados finitos la cual alterna continuamente entre el color del fondo y el color de los elementos como las raquetas, la línea central, etc.

## Descripción de Bloques del sistema:

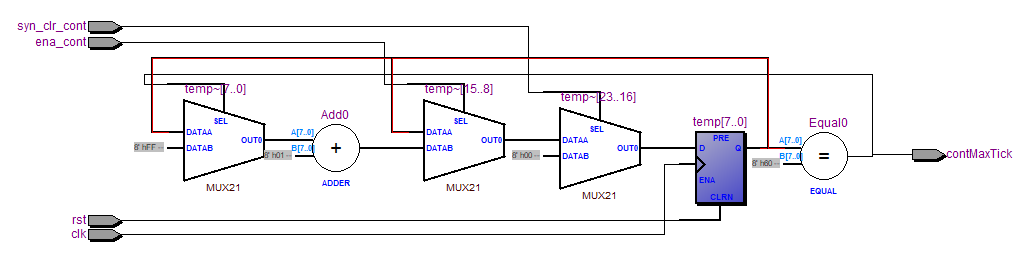


Fig. 13. Diagrama esquemático de bloques contador

En el presente diagrama se presenta el bloque genérico usado para los contadores, de forma que al terminar los conteos escogidos para cada contador, se obtiene una señal de un bit a la salida que indica que ya se ha contado hasta este valor, y se reinician los conteos. Este bloque fue implementado por medio de *PROCESS* e *IF*.

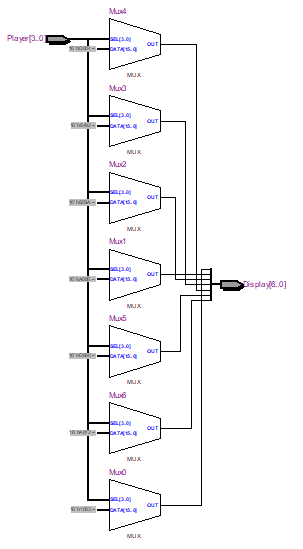


Fig.14 . Diagrama esquemático de bloques bin to sseg para mostrar en el 7 segmentos.

En el presente diagrama se presenta el bloque genérico usado para generar los números de los display siete segmentos a partir de una señal binaria de 4 bits, cuya salida con los 7 cables correspondientes a cada segmento, de forma que se muestra en tiempo real el valor decimal al cual corresponden los números binarios en la entrada. Este bloque fue implementado por medio de un *PROCESS* y un *FOR*.

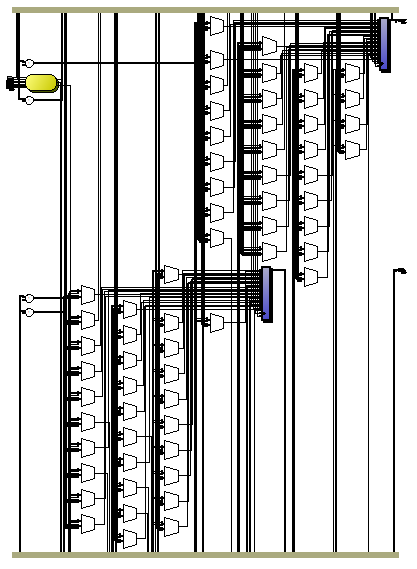


Fig.15 . Diagrama esquemático de bloque control barra.

En el presente diagrama se presenta el bloque genérico usado para los controladores de las barras, de forma que reciben como entrada la posición dictada por el *JoyStick* y el valor de los límites superiores e inferiores actuales de sí misma, para los cuales evalúa la dirección de movimiento y el cumplimiento de los límites para determinar la siguiente posición de la misma, teniendo como salidas los nuevos valores de los límites de las barras. Si el movimiento indicado por el *joystick*es hacia arriba, se evalúa si se puede mover hacia arriba sin salir de los límites superior e inferior de la pantalla, y se dar el caso se mueve 2 pixeles hacia esta dirección, de no ser el caso se mantiene estática y solo se puede mover hacia abajo donde se sigue el mismo principio de funcionamiento. La velocidad de movimiento de las barras es de 2 pixeles por frame, y el bloque fue implementado por medio de máquinas de estados finitos.

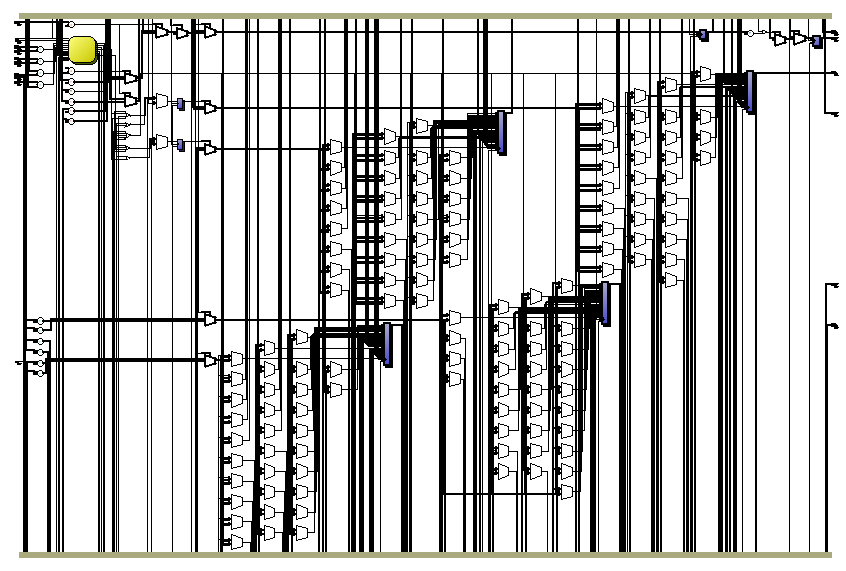


Fig.16 . Diagrama esquemático de bloque control bolita.

En el presente diagrama se presenta el bloque usado para el controlador de la bolita, de forma que se reciben como entradas las posiciones de ambos *JoyStick* y los valores de los límites superiores e inferiores de ambas barras junto con los cuatro límites de sí misma, es decir, superior, inferior, derecho e izquierdo. Por medio de esta información se evalúa el caso, el cual determina la presencia de movimiento y la dirección del mismo, de forma que si se detectan movimientos en cualquiera de los dos joystick la bolita da inicio y entra a un modo de funcionamiento normal donde las condiciones la dirigen en una dirección vertical y en una dirección horizontal, de forma que al encontrarse con cualquiera de los límites superior o inferior, cambia su dirección vertical, y al encontrarse con los límites laterales delimitados por las paletas, cambia su dirección horizontal. Al no encontrarse dentro de los límites de las paletas en el umbral de las mismas, se da por perdido el punto y el mismo control de la bolita determina en cuál límite se falló la respuesta para anotar el punto al jugador y vuelve a iniciar en la posición central a la espera de movimiento en los *joystick*. Dentro del control de la bolita también se da el reinicio de los contadores de puntaje en el case de que alguno de los jugadores logre anotar 9 puntos, y este sería el ganador.

La velocidad de movimiento de la bolita es de 3 píxeles por frame, de forma que se desplaza 3 pixeles hacia arriba, pero 2 hacia los lados por cada frame, de manera que se logra un ángulo de golpe repetitivo de un periodo considerablemente grande. La implementación de este bloque se realizó por medio de máquinas de estados finitos para determinar la modalidad de movimiento de la bolita, teniendo como salidas los nuevos valores de los cuatro límites de la misma.

## 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, manda un valor inicial de donde se encuentran los pixeles de las barras y la bola inicialmente y va variando este valor por medio de dos máquinas de estados que van verificando el estado de la bola y las barras a medida que va avanzando el juego.

Dentro de la máquina de estados principal, el manejo de los límites con los cuales se define si se saca un pixel blanco o un píxel negro se realiza dentro de la máquina de estados de control barras. Dentro de esta máquina se definen la posición inicial de las barras y dependiendo del estado de los *Joystick* se le suma o resta 2 al valor de la posición inicial, modificando así las variables límite superior e inferior de la máquina de estados principal. La máquina de estados control dibujo, compara el valor del límite generado por la máquina de estados de control barra con el contador de píxeles y líneas para así saber si entra al estado de blanco o negro.

La máquina de estados del control bolita me genera la posición de la bola verificando si ha chocado con algún borde, barra o si la bola ha salido de los límites de las barras realizándose un punto. La máquina espera a que se genere algún movimiento en los joystic para iniciar el desplazamiento de la bola sumándole o restándole 3 a la posición inicial en su componente vertical y horizontal dada una comparación entre los límites de la barra o de la pantalla con la posición en la cual se encuentra la bola.

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.

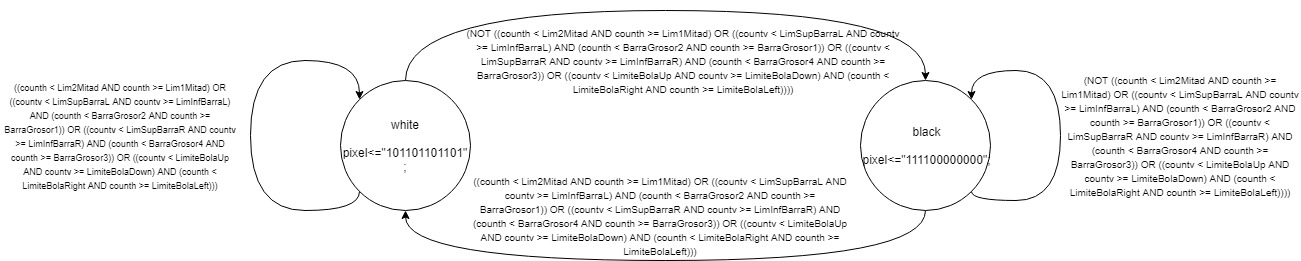


Fig.17 . Diagrama de máquina de estados control dibujo.

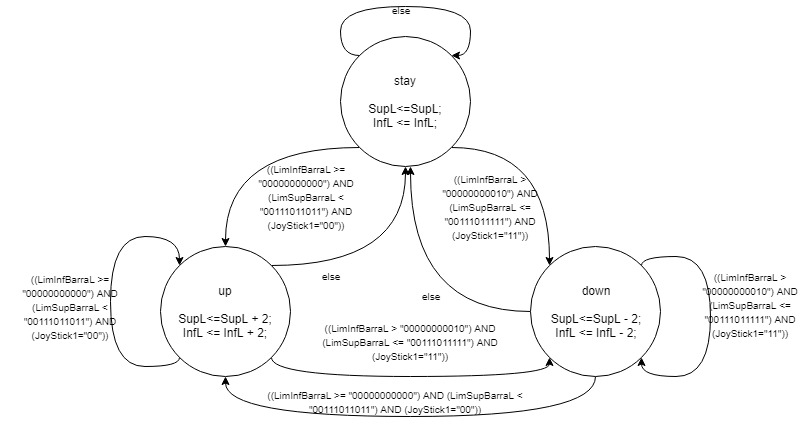


Fig. 18. Diagrama de máquina de estados control barra L, el control barra R es igual a este pero cada variable en vez de L va con R.

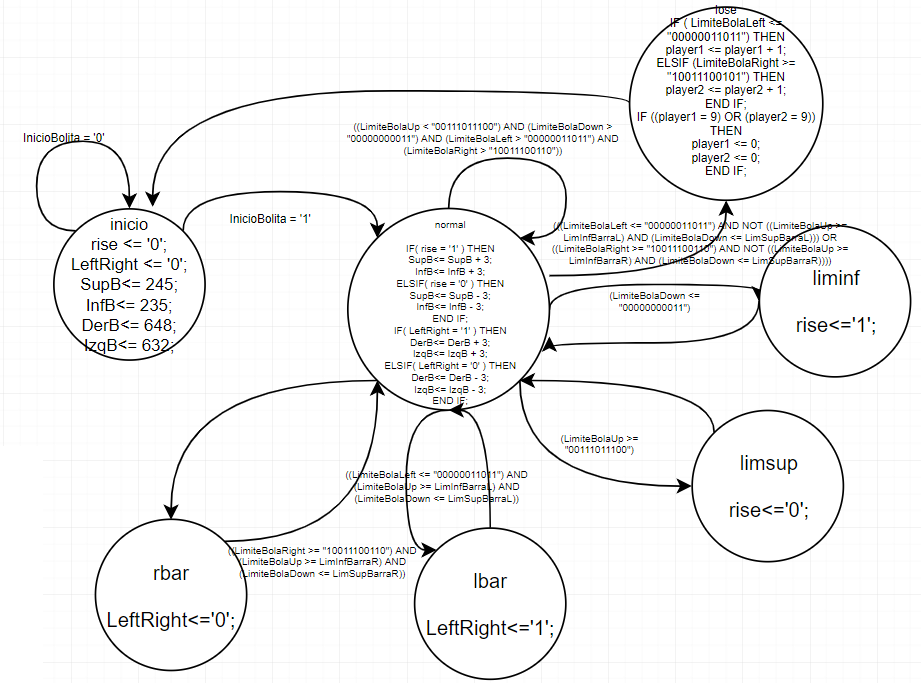


Fig. 19. Diagrama de máquina de estados control bolita.

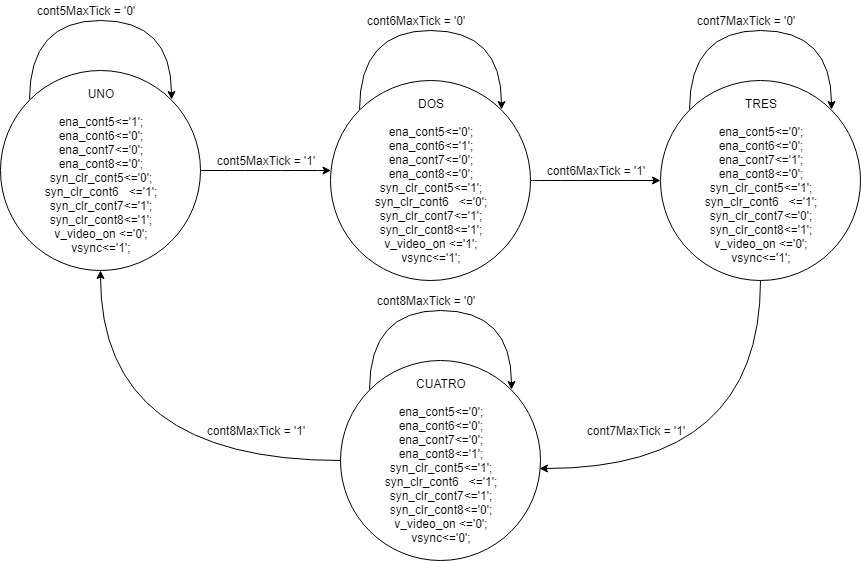


Fig. 20. Diagrama de máquina de estados control vertical.

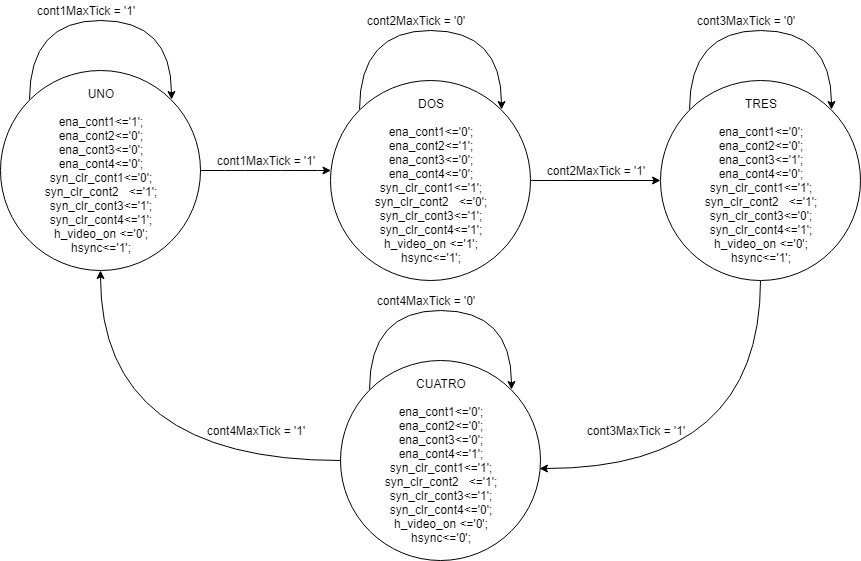


Fig.21 . Diagrama de máquina de estados control horizontal.

## 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 la posición de la bolita, la posición de las barras y la línea central para pintar de blanco los píxeles donde se encuentran estas cosas, luego si no se comparan estas cosas se pasa al estado black, o se queda en el estado white hasta que termine de comparar y pintar todos los píxeles de 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 la posición de la bolita, la posición de las barras y la línea central para pintar de blanco los píxeles donde se encuentran estas cosas, luego si no se comparan estas cosas se pasa al estado white, o se queda en el estado black hasta que termine de comparar y pintar todos los píxeles de negro.

Diagrama de máquina de estados control barra:

* stay: Este estado es cuando la señal del Joystick no es ni ‘00’ ni ‘11’, lo cual indica que la barra debe estar quieta, de lo contrario si esta en ‘00’ pasa al estado up si la barra aún se puede mover donde de los límites superior e inferior de la pantalla, o si está en ‘11’ pasa al estado down si la barra aún se puede mover donde de los límites superior e inferior de la pantalla.
* up: Se queda en este estado mientras el Joystick está en ‘00’ y la barra puede seguirse moviendo hacia arriba y se compara con el límite superior, si ya están los límites iguales se pasa al estado stay, o si el Joystick cambia a ‘11’ se pasa al estado down donde comienza a bajar la barra.
* down: Se queda en este estado mientras el Joystick está en ‘11’ y la barra puede seguirse moviendo hacia abajo y se compara con el límite inferior, si ya están los límites iguales se pasa al estado stay, o si el Joystick cambia a ‘00’ se pasa al estado up donde comienza a bajar la barra.

Diagrama de máquina de estados control bolita:

* inicio: Este es el estado inicial que espera a que uno de los dos Joysticks sea movido para cambiar al estado normal.
* normal: Está en el estado normal cuando la bolita no ha pegado con las barras o con los límites o se ha anotado punto. En cualquiera de estos casos la se pasa a los estados limsup, liminf, lbar, rbar o lose respectivamente se requiera y depende cómo se compara la posición de la bolita en ese momento.
* limsup: Este estado es para cuando la bolita pega con el límite superior y sirve para cambiar a la posición bolita de dirección. Inmediatamente pasa a el estado normal.
* liminf: Este estado es para cuando la bolita pega con el límite inferior y sirve para cambiar a la posición bolita de dirección. Inmediatamente pasa al estado normal.
* lbar: Este estado es para cuando la bolita pega con la barra izquierda y sirve para cambiar a la posición bolita de dirección. Inmediatamente pasa al estado normal.
* rbar: Este estado es para cuando la bolita pega con la barra derecha y sirve para cambiar a la posición bolita de dirección. Inmediatamente pasa al estado normal.
* lose: Este estado es para cuando la bolita anota un punto y se actualizan los contadores del jugador que mete la bolita en el otro tablero. Inmediatamente pasa al estado inicio.

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.

# 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 control dibujo para poder mostrar la imagen.

Se realizaron pruebas de todo el sistema, el control dibujo, los controles de las barras y el control bolita, A continuación se muestra la simulación realizada en test bench para el caso.

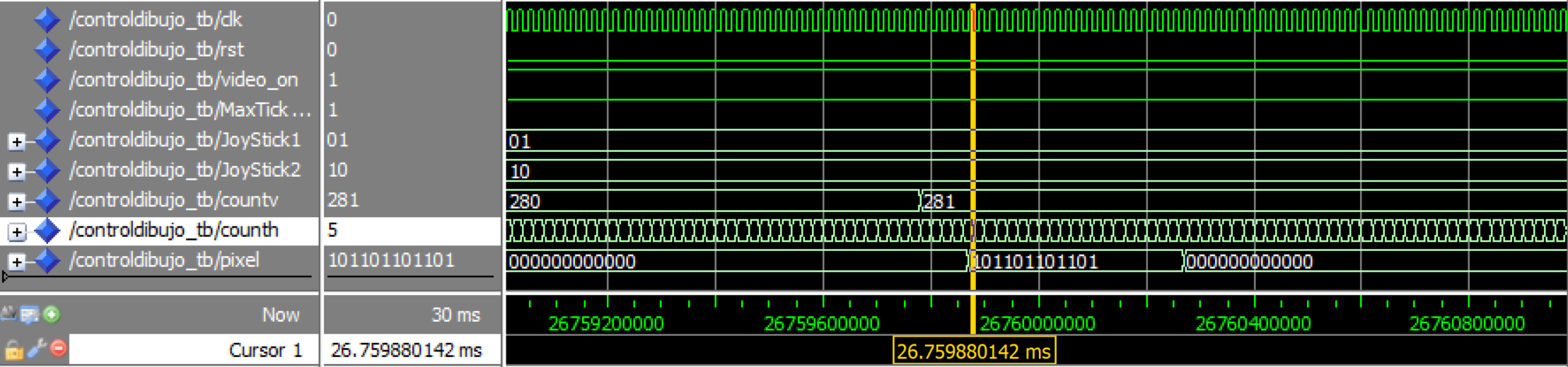


Fig.22 . Simulación control dibujo.

En la figura 22 se muestra la simulación en test bench del control dibujo en donde se puede observar la salida de los contadores countv y counth que son los que cuentan cada línea de VGA y con video\_on en ‘1’, lo que hace que se pueda visualizar en la pantalla. Como se puede ver en la señal píxel se imprime un color dependiendo qué color se quiere que sea ese píxel, en este caso los dos colores son negro para el fondo de la pantalla y cyan para las barras, la bolita y la línea central. En este caso podemos ver que color se imprime dependiendo el píxel.

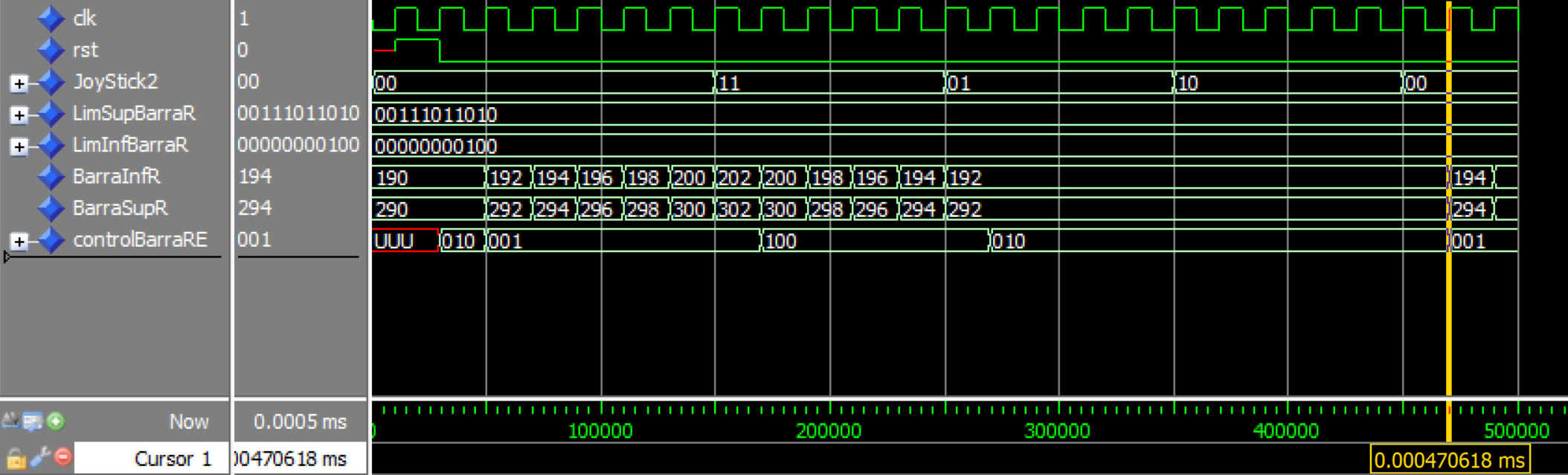


Fig.23 . Simulación control barra.

En la figura 23 se muestra la simulación en test bench del control de la barra derecha que es el mismo para la barra izquierda. Inicialmente se puede observar que cuando el Joystick2 que es el que maneja la barra derecha cambia su posición a ‘00’ cambia del estado stay a up, es decir, la posición inicial de la barra es en BarraInfR en 190 y en BarraSupR, al momento es que se cambia el Joystick2 y se encuentra en el estado up comienzan a aumentar las posiciones de la barra a 2 píxeles por frame. Luego en aproximadamente 0.00015 ms la posición del Joystick2 cambia a ‘11’, esto quiere decir que pasa del estado up al estado down, entonces la barra comienza a bajar las posiciones a 2 píxeles por frame.

Finalmente el Joystick no se mueve, es decir que queda en la posición ‘01’ por lo tanto el control barra se encuentra en estado stay y la barra se queda en la misma posición que estaba hasta que se vuelve a cambiar la posición del Joystick2. El funcionamiento del control barra derecha es exactamente el mismo.

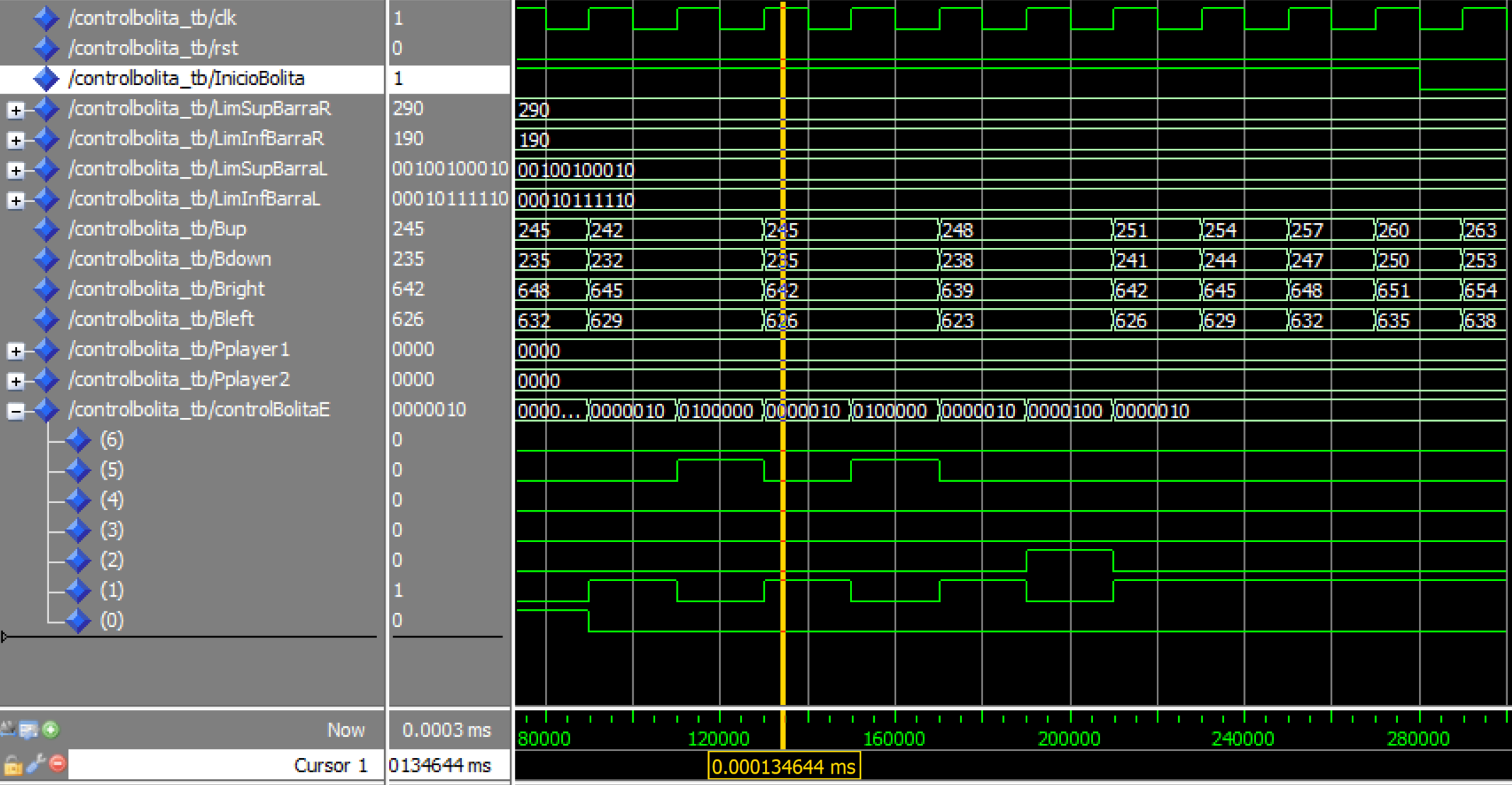


Fig.24 . Simulación control bolita.

En la figura 24 se muestra la simulación del control bolita. Inicialmente se tiene a la bolita en su posición inicial, luego al poner InicioBolita en ‘1’ se pasa del estado inicio al estado normal, en donde la bolita comienza a moverse, es decir que todas sus posiciones cambian a 3 píxeles por frame y se mueve con cierto ángulo. Cuando cambia de estado la bolita cambia de dirección dependiendo el estado.

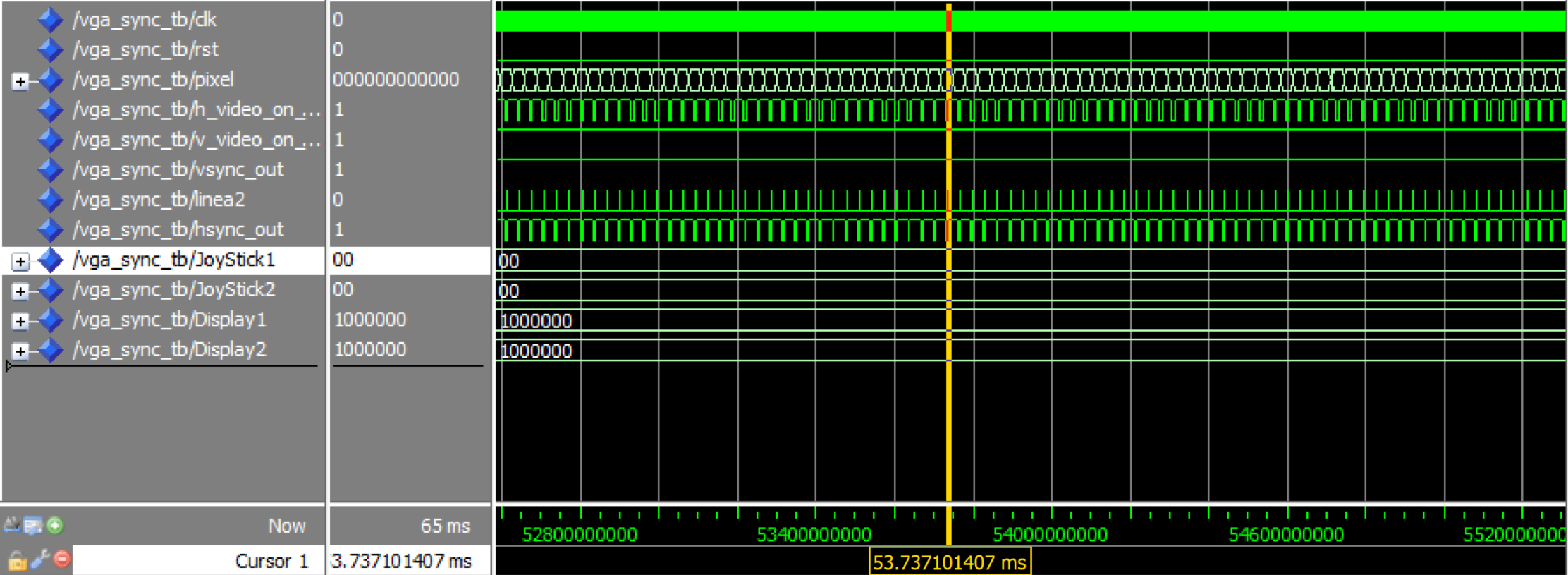


Fig.25 . Simulación vga-sync pong.

En la figura 25 se muestra la simulación completa del proyecto vga-sync y se muestra el proceso completo de la impresión en la pantalla vga con las señales importantes para este proceso hsync, vsync, v\_video\_on y h\_video\_on y su funcionamiento.

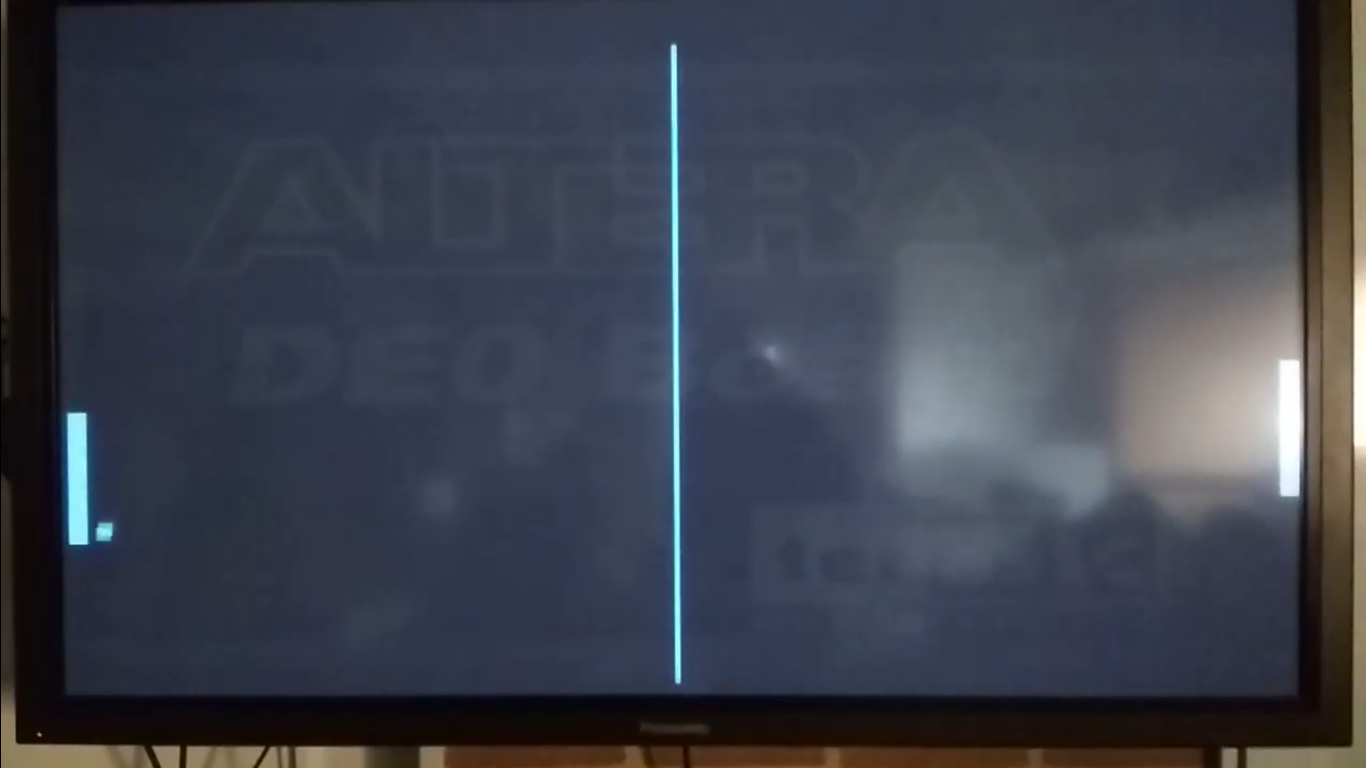


Fig. 26. Pantallazo 1 implementación juego de ‘pong’.



Fig. 27. Pantallazo 2 implementación juego de ‘pong’.



Fig. 28. Pantallazo 3 implementación juego de ‘pong’.

En las figuras 26, 27 y 28 se muestra una pequeña serie de pantallazos que muestran un poco del funcionamiento del juego. Pues en la figura 26 se muestra como la bola rebota (cambia de dirección) cuando los límites de la barra y los límites de la bolita se juntan, luego de rebotar en la figura 27 se observa el funcionamiento de la bolita avanzando a 3 píxeles por frame y también se observa un cambio en la posición de las barras. Finalmente en la figura 28 se observa como la bolita rebotó en el límite inferior del juego y cambió su dirección y en ese cambio de la dirección se dirige a un lugar donde no se encuentra la barra.

Finalmente se anexa el link del video completo de la implementación de la versión del juego ‘pong’ realizada en una pantalla con entrada *VGA*: <https://www.youtube.com/watch?v=Se08EuopNTM>

# Conclusiones

Para la realización del presente 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.

##### Referencias

1. E. Gerlein, “Project No. 1 Digital design in fpga: PONG”. 2019.
2. 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].
3. 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].