[86.41] Sistemas Digitales
TP: Diseño de un motor de rotación gráfico 3D basado en el algoritmo CORDIC

Sampayo, Sebastián Lucas

Segundo Cuatrimestre 2015

Índice

1. Objetivo

El objetivo del presente Trabajo Práctico es desarrollar una arquitectura de rotación de objetos 3D basada en el algoritmo CORDIC, implementando tanto la unidad aritmética de cálculo como así también el controlador de video asociado.

Para esto se utilizará el Kit Nexys 2 que se encuentra en la facultad. Este consiste de una FPGA SPARTAN 3S500E FG320, y conexiones con periféricos de uso común, como lo son un puerto VGA, RS232, memoria RAM externa, pulsadores, llaves y leds.

La estructura a implementar, propuesta por la cátedra, es la siguiente:

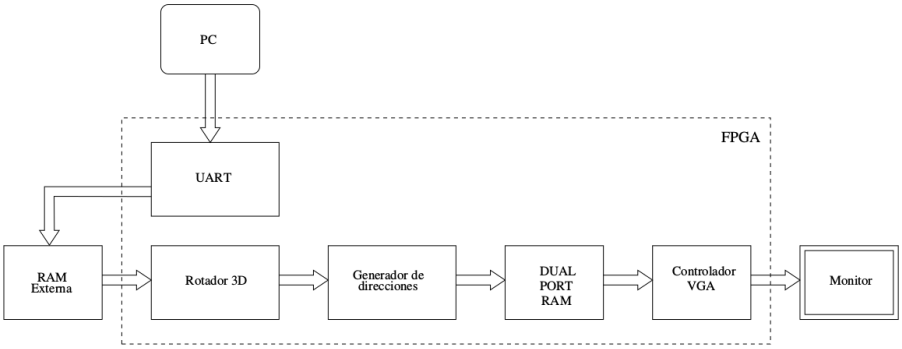

Figura 1: Diagrama en bloques propuesto.

2. Desarrollo

Al analizar el problema, se destacaron los siguientes puntos:

2.1. Memoria RAM externa

La memoria RAM externa a utilizar consta de 8M x 16 bits, por lo tanto almacena en cada dirección, palabras de 16 bits. En cada ciclo de escritura/lectura, transfiere 16 bits. De este modo, utilizar números binarios de 16 bits de precisión es la manera más cómoda de trabajar con esta memoria. Por esta razón, las coordenadas de los puntos 3D que provienen de la PC, se pasan previamente a números enteros con signo, en el rango $(-2^{15}+1, 2^{15}-1)$ mediante un script de Octave.

Por otro lado, para leer y escribir los datos se utilizaron 2 contadores para llevar el registro de cada dirección pedida en cada caso. El control global comanda sus señales de control.

2.2. UART

La UART transfiere datos de 8 bits en cada transmisión/recepción, por lo tanto se requieren como mínimo 2 transferencias para comunicar 1 solo número de 16 bits. Para alcanzar esta velocidad máxima, los puntos fueron almacenados en la PC en un archivo binario, el cual se cargó en la terminal de manera directa.

2.3. Máquina de Estados

La frecuencia de actualización de pantalla del controlador de la VGA es de 50Hz, esto implica un período de 20ms para barrer la pantalla completa. Al escribir datos en la memoria de video, se debe esperar a que el controlador de la VGA pueda efectivamente leerlo e imprimirlo, antes de borrar y escribir nuevos datos. Este control se ejecutó mediante una máquina de estados global implementada en el bloque 'global control'. Los estados correspondientes son (de manera ordenada secuencialmente): IDLE (en espera), CLEANING (borrado de la memoria de video), READING (lectura de la RAM externa y rotación CORDIC), WAITING (esperando que la VGA llegue al principio de la porción de pantalla a utilizar) y REFRESHING (cuando la VGA barre la porción de pantalla en la que queremos imprimir). Distintas señales de control funcionan como eventos para cambiar estos estados. Además, este bloque controla los acumuladores de los ángulos a rotar para cada eje.

2.4. XYZ Loader

Este bloque lee 3 veces de la memoria RAM externa, para formar un punto X,Y,Z, y una vez listo, lo envía a su salida junto a una señal de control de validez (RxRdy).

2.5. Rotador XYZ

El núcleo del sistema se encuentra en esta entidad, ya que aloja los bloques CORDIC que generan las rotaciones de los puntos. Esencialmente, consta de 3 CORDICs en cascada (con retardos en el medio, debido a que alguna componente no es rotada en cada caso). Además se agregó un registro de entrada y otro de salida para acortar el camino combinacional que implica la cascada con otros bloques de lógica combinacional.

2.6. Memoria de Video

La memoria de video fue implementada con una Dual Port Ram Asimétrica, donde el lado A soo escribe y el lado B solo lee.

2.7. Funciones adicionales

A modo de facilitar el testeo del sistema se agregaron 2 botones para puentear el bloque de rotaciones y por otro lado el escalado. De esta manera se puede ver directamente lo que hay en la RAM externa, descartando problemas en esoos bloques y aislando el cordic.

2.8. Esquema general (informal)

Para facilitar la lectura del código se confeccionó el siguiente diagrama en bloques conceptual, que remite a la estructura del sistema final.

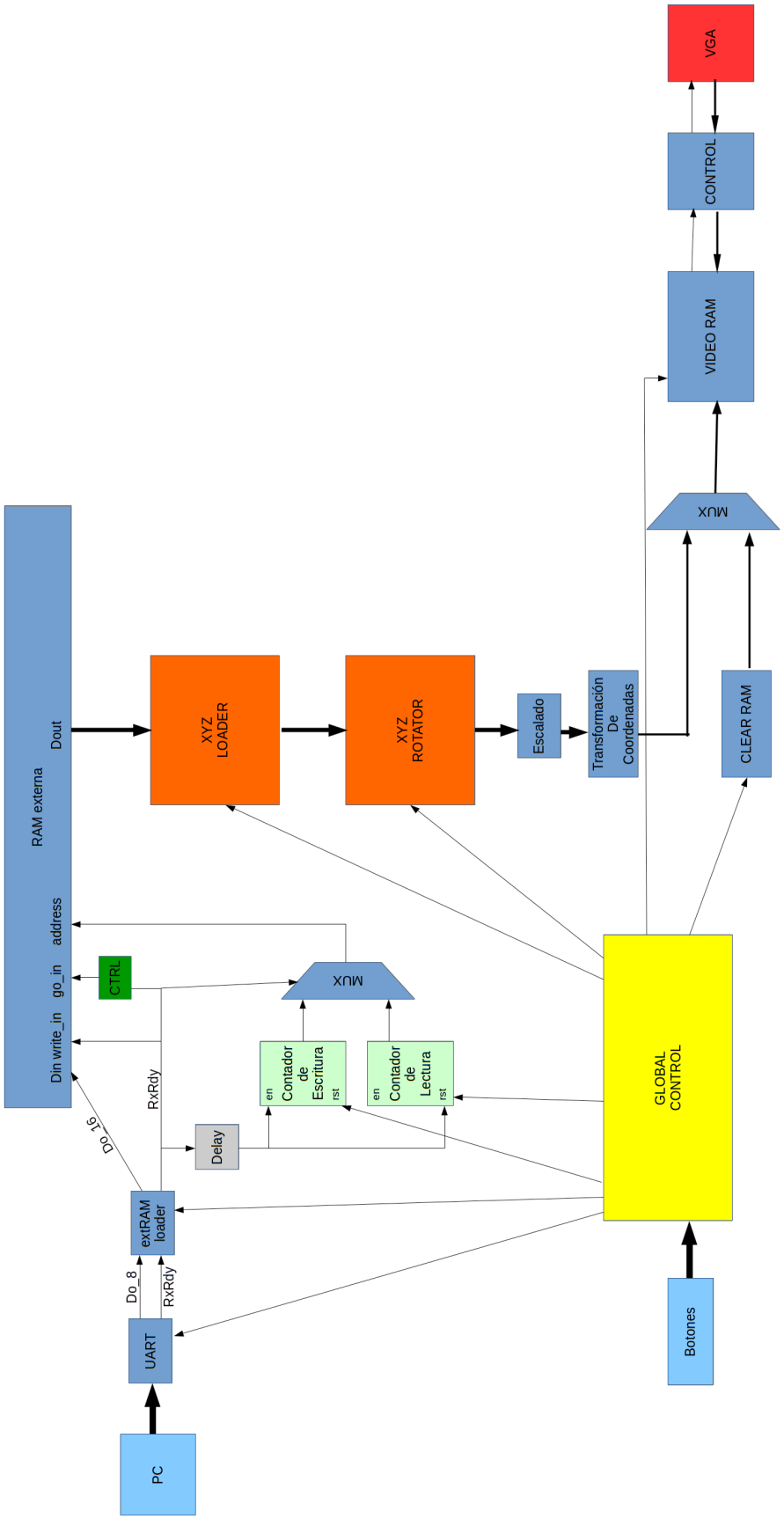

Figura 2: Diagrama en bloques desarrollado (informal).

3. Implementación - ISE

4. Capturas