Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

Implementación en Verilog de Unidad de Generacion de Rayos para GPU Theia

Por:

Josué David Vargas Amador

Ciudad Universitaria "Rodrigo Facio", Costa Rica

Diciembre de 2015

Implementación en Verilog de Unidad de Generacion de Rayos para GPU Theia

Por:

Josué David Vargas Amador

IE-0499 Proyecto eléctrico

Aprobado por el Tribunal:

MSc. Diego Valverde Garro
Profesor guía

MSc. Carlos Duarte Martínez
Profesor lector

MSc. Rodolfo Brenes Fernández
Profesor lector

Índice general

Índice de figuras

Índice de tablas

1 Introducción

1.1 Justificación

Los sistemas computacionales actuales poseen, dentro de su arquitectura, módulos de hardware especializados llamados Unidades de Procesamiento Gráfico (GPU, por sus siglas en inglés) encargados de acelerar el proceso de representación de objetos tridimensionales en la pantalla del computador.

Las unidades de procesamiento gráfico permiten la visualización de objetos mediante el cálculo de las primitivas que conforman el modelo abstracto de las imágenes. Las GPU implementan distintos algoritmos de representación gráfica, entre estos, uno es el algoritmo de Ray Casting.

El algoritmo de Ray Casting genera vectores (rayos) normalizados desde la perspectiva del usuario y calcula la intersección de los rayos con los objetos del escenario, además colabora con la formación de los colores, con la finalidad de crear las imágenes mostradas en pantalla.

Entonces los cálculos para la representación de objetos visuales en una GPU de tipo raycasting requieren de una arquitectura interna capaz de la generación de vectores (rayos) normalizados, para luego usar estas estructuras de datos en los módulos dedicados a la intersección de rayos. La generación de rayos requiere de instrucciones capaces de realizar cálculos aritméticos como multiplicaciones, sumas y restas, así como operaciones especializadas que permitan aproximar los valores de raíces cuadradas, por lo que el diseño lógico de una unidad dedicada facilitaría el proceso de creación rayos y permitiría añadir flexibilidad y modularidad al diseño de todo el GPU. Existen proyectos de hardware relacionados al diseño de arquitecturas de trazado de rayos que implementan unidades de generación de rayos propias como SaarCor de la Universidad de Saarland (Schmittler et al. (2004)) y RayCore de la Universidad de Sejong (Nah et al. (2014)).

En el caso de la GPU de tipo raycasting Theia, las especificaciones arquitectónicas indican la necesidad de una Unidad de Generación de Rayos (RGU, por sus siglas en inglés). La RGU debe poseer un conjunto de instrucciones necesarias para el cálculo de la normalización de vectores tridimensionales empleados en las siguientes etapas de funcionamiento del GPU.

1.2 Alcances del proyecto

La GPU Theia se encuentra en su tercera iteración, y en esta etapa tiene dos módulos principales dentro de su descripción de RTL en el lenguaje Verilog: la unidad de generación de rayos normalizados (RGU) y el módulo de intersección de rayos de tipo AABB (siglas en inglés de Axis Aligned Bounding Boxes).

El módulo de la RGU es un módulo que posee dentro de su descripción las instrucciones necesarias para el funcionamiento apropiado de la generación de rayos normalizados. Estas instrucciones deben ser capaces de proveer la información necesaria para programar el módulo de la RGU de manera que permita la normalización de los vectores mediante cálculo aproximado del inverso de la raíz cuadrada empleando el método iterativo para aproximación de raíces Newton-Raphson.

Posterior a esto se debe plantear un ambiente de verificación funcional que permita afirmar que el módulo RGU está cumpliendo con su papel dentro de la arquitectura y que puede generar la información requerida por los módulos de intersección de rayos.

1.3 Objetivos

Objetivo general

Desarrollar el modelo por comportamiento en Verilog de una Unidad de Generación de Rayos de un GPU tipo ray casting.

Objetivos específicos

Desarrollar el modelo por comportamiento en Verilog de una Unidad de Generación de Rayos de un GPU tipo ray casting.

- Investigar bibliografía sobre el mecanismo generación de rayos.
- Definir el mecanismo de generación de rayos normalizados en el GPU.
- Verificar el comportamiento funcional de la Unidad de Generación de Rayos en el GPU.

1.4 Metodología

1. Se procederá a investigar los conceptos fundamentales de la arquitectura de la GPU, el algoritmo de ray casting y sobre los posibles mecanismos de la generación de rayos normalizados.

2. Se buscará la implementación final de la arquitectura interna de la RGU de modo que contenga las instrucciones necesarias para la normalización.

3. Se simulará la ejecución del código en la RGU para generar los rayos normalizados necesitados por los módulos de intersección de rayos de tipo AABB.

4. Se verificará el comportamiento funcional del módulo RGU con la finalidad de establecer un marco de referencia para la futura validación del resto de la versión actual del GPU Theia.

1.5 Desarrollo

Este proyecto se estructura por medio de capítulos, cada uno tiene como tarea aclarar los siguientes tópicos:

1. Capítulo I: Introducción. Muestra la justificación del proyecto, los alcances y limitaciones, los objetivos y la metodología que permite cumplir los mismos.

2. Capítulo II: Antecedentes y Marco Teórico. Introduce al lector conceptos claves de arquitectura de unidades de procesamiento gráfico, algoritmo de raycasting, y plantea los casos de proyectos donde se han implementado chips.

3. Capítulo III: Implementación final de la unidad de generación de rayos. Aquí se describe la arquitectura final de la unidad, así como el método empleado usando las instrucciones de ésta para implementar la unidad.

4. Capítulo IV: Prueba de verificación funcional. Se comprueba la funcionalidad del módulo conductual de lenguaje Verilog por medio de un ambiente de verificación apropiado.

5. Capítulo V: Conclusiones y recomendaciones. Se muestran posibles resultados del proyecto y reflexiones sobre el futuro del proyecto.

Bibliografía

Nah, J., KWON, H., Kim, D., Jeong, C., Park, J., HAN, T., Manocha, D., y Park, W. (2014). Raycore: A ray-tracing hardware architecture for mobile devices. *ACM Transactions on Graphics, Vol. 33*.

Schmittler, J., Woop, S., Wagner, D., Paul, W., y Slusallek, P. (2004). Realtime ray tracing of dynamic scenes on an fpga chip. *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*.