(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 820 716**

(51) Int. CI.:

G06T 15/00 (2011.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **26.01.2015 PCT/US2015/012917**

(87) Fecha y número de publicación internacional: **27.08.2015 WO15126574**

(96) Fecha de presentación y número de la solicitud europea: **26.01.2015 E 15704425 (6)**

(97) Fecha y número de publicación de la concesión europea: **24.06.2020 EP 3108452**

(54) Título: **Tubería de sombreadores con canales de datos compartidos**

(30) Prioridad:

18.02.2014 US 201414182976

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
22.04.2021

(73) Titular/es:

QUALCOMM INCORPORATED (100.0%)
5775 Morehouse Drive
San Diego, CA 92121-1714, US

(72) Inventor/es:

MEI, CHUNHUI;
GOEL, VINEET y
KIM, DONGHYUN

(74) Agente/Representante:

FORTEA LAGUNA, Juan José

DESCRIPCIÓN

Tubería de sombreadores con canales de datos compartidos

CAMPO TÉCNICO

[0001] La presente divulgación se refiere a canales de datos compartidos para almacenar datos producidos y consumidos por etapas de una tubería de procesamiento gráfico.

ANTECEDENTES

[0002] Una unidad de procesamiento gráfico (GPU) de un dispositivo informático puede ejecutar una tubería de procesamiento gráfico que incluye una pluralidad de etapas para procesar comandos gráficos para representar una representación bidimensional de una escena tridimensional. Una escena tridimensional típicamente está compuesta de vértices, y la tubería de procesamiento gráfico incluye una serie de etapas que, para cada vértice de la escena tridimensional, se ejecutan en un orden fijo para representar una representación bidimensional de la escena tridimensional.

[0003] La tubería de procesamiento gráfico puede incluir una cadena de etapas de sombreador que se ejecuta para transformar los vértices de la escena tridimensional. Cada una de las etapas de sombreador consume datos producidos por etapas previas y produce datos para las siguientes etapas. Debido a las enormes cantidades de datos que fluyen a través de la cadena de etapas de sombreador, la forma en que se gestionan los datos para la cadena de etapas de sombreador puede afectar el rendimiento y la eficacia de memoria de la GPU.

[0004] El documento WO2009/145889 A1 divulga un procedimiento para proporcionar datos primitivos teselados a un sombreador de geometría. El procedimiento comprende calcular un conjunto de vértices teselados y un conjunto calculado de datos de conectividad en base a un conjunto original de vértices y un conjunto original de datos de conectividad, generar datos de vértice calculados en base al conjunto original de vértices y el conjunto de vértices teselados, recibir el conjunto calculado de datos de conectividad, solicitar un subconjunto de los datos de vértice calculados y procesar primitivas definidas por el subconjunto de los datos de vértice calculados.

BREVE EXPLICACIÓN

[0005] En un ejemplo de la divulgación, un procedimiento para el procesamiento gráfico puede incluir asignar, por una unidad de procesamiento gráfico (GPU), un canal de datos compartido en la memoria gráfica en chip de la GPU que se comparte por al menos dos etapas de una tubería de procesamiento gráfico. El procedimiento puede incluir además ejecutar, en unidades de sombreador en la GPU, las al menos dos etapas de la tubería de procesamiento gráfico. El procedimiento puede incluir además almacenar, por la GPU en el canal de datos compartido en la memoria gráfica en chip, datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico que se ejecuta en las unidades de sombreador.

[0006] En otro ejemplo de la divulgación, un aparato para el procesamiento gráfico puede incluir una unidad de procesamiento gráfico (GPU) configurada para: asignar un canal de datos compartido en la memoria gráfica en chip de la GPU que se comparte por al menos dos etapas de una tubería de procesamiento gráfico; ejecutar, en unidades de sombreador en la GPU, las al menos dos etapas de la tubería de procesamiento gráfico; y almacenar, en el canal de datos compartido en la memoria gráfica en chip, los datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador.

[0007] En otro ejemplo de la divulgación, un aparato para gráficos para procesamiento gráfico puede incluir medios para asignar un canal de datos compartido en la memoria gráfica en chip de una unidad de procesamiento gráfico (GPU) que se comparte por al menos dos etapas de una tubería de procesamiento gráfico. El aparato puede incluir además medios para ejecutar las al menos dos etapas de la tubería de procesamiento gráfico. El aparato puede incluir además medios para almacenar, en el canal de datos compartido en la memoria gráfica en chip, datos producidos por la ejecución de cada una de las al menos dos etapas de la tubería de procesamiento gráfico.

[0008] En otro ejemplo de la divulgación, un medio de almacenamiento legible por ordenador puede almacenar instrucciones que, cuando se ejecutan, hacen que uno o más procesadores programables: asignen un canal de datos compartido en la memoria gráfica en chip que se comparte por al menos dos etapas de una tubería de procesamiento gráfico; ejecuten, en unidades de sombreador, las al menos dos etapas de la tubería de procesamiento gráfico; y almacenen, en el canal de datos compartido en la memoria gráfica en chip, los datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador.

[0009] Los detalles de uno o más ejemplos se exponen en los dibujos adjuntos y en la descripción a continuación. Otros rasgos característicos, objetivos y ventajas resultarán evidentes a partir de la descripción y de los dibujos, y a partir de las reivindicaciones.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0010]

La FIG. 1 es un diagrama de bloques que ilustra un dispositivo informático de ejemplo que se puede configurar para implementar uno o más aspectos de la presente divulgación para compartir canales de datos entre múltiples etapas de tubería de gráficos.

La FIG. 2 es un diagrama de bloques que ilustra una tubería de procesamiento gráfico de ejemplo que se puede realizar por una GPU para crear una representación bidimensional de una escena tridimensional.

La FIG. 3 es un diagrama de bloques que ilustra con más detalle implementaciones de ejemplo de la CPU, la GPU y la memoria de sistema de la FIG. 1.

La FIG. 4 es un diagrama de bloques que ilustra un ejemplo de canales de datos compartidos en uso en una tubería de procesamiento gráfico.

La FIG. 5 es un diagrama de bloques que ilustra con más detalle el canal compartido en modo caché de la FIG. 4.

La FIG. 6 es un diagrama de flujo que ilustra un proceso de ejemplo para compartir canales de datos por etapas de una tubería de procesamiento gráfico.

DESCRIPCIÓN DETALLADA

[0011] En general, la presente divulgación describe técnicas para una tubería de sombreadores de paso único que usa un modelo productor-consumidor con canales de datos compartidos. Una unidad de procesamiento gráfico (GPU) en un dispositivo informático puede ejecutar una tubería de sombreador en unidades de sombreador que pueden ejecutar múltiples etapas de la tubería de sombreador al mismo tiempo en una GPU. Debido a que los datos almacenados en la memoria en chip en la GPU típicamente se pueden acceder de manera más rápida y eficaz que los datos almacenados en la memoria de sistema del dispositivo informático, la eficacia de las unidades de sombreador en la GPU se puede incrementar consumiendo datos de los canales de datos en la memoria en chip en la GPU y produciendo datos que también se almacenan en canales de datos en la memoria en chip en la GPU.

[0012] En un ejemplo, la GPU puede asignar canales de datos de igual tamaño para almacenar los datos que se consumen y producen por la ejecución de la tubería de sombreador por las unidades de sombreador en la GPU. Sin embargo, debido a que la memoria en chip en la GPU típicamente incluye mucho menos espacio de almacenamiento que la memoria del sistema del dispositivo informático, la memoria en chip en la GPU puede no tener suficiente espacio de almacenamiento para asignar canales de datos separados para todos los datos que se consumen y producen por las unidades de sombreador en la GPU. Además, debido a que las etapas de la tubería de sombreador pueden estar desequilibradas de modo que algunas etapas de la tubería de sombreador tienden a producir más datos que otras etapas de la tubería del sombreador, asignando igual espacio en la memoria en chip para los datos producidos por cada etapa de la tubería de sombreador puede desperdiciar espacio de almacenamiento en la memoria de chip. Además, la memoria en chip puede no tener suficiente espacio de almacenamiento para asignar igual espacio para los datos producidos por cada etapa de la tubería de sombreador, de modo que puede ser necesario almacenar al menos algunos de los datos producidos por las etapas de la tubería de sombreador en la memoria de sistema más lenta, reduciendo de este modo el rendimiento de la GPU.

[0013] De acuerdo con los aspectos de la presente divulgación, la GPU puede asignar canales de datos compartidos en la memoria en chip en la GPU de modo que dos o más etapas de la tubería de sombreador puedan compartir un único canal de datos compartido, de modo que el espacio en un canal de datos compartido que no se usa para almacenar datos de una primera etapa de la tubería de sombreador que comparte el canal de datos compartido se pueda usar para almacenar datos de una segunda etapa de la tubería de sombreador que comparte el canal de datos. De esta manera, la memoria en chip en la GPU se puede utilizar de una manera más eficaz. Además, utilizando la memoria en chip en la GPU de una manera más eficaz para almacenar potencialmente más datos producidos por la tubería de sombreador en comparación con otros enfoques, la memoria en chip en la GPU puede almacenar más datos que están listos para consumirse por las unidades de sombreador que ejecutan etapas de la tubería de sombreador, incrementando de este modo la utilización de las unidades de sombreador e incrementando el rendimiento de la GPU.

[0014] La FIG. 1 es un diagrama de bloques que ilustra un dispositivo informático de ejemplo que se puede configurar para implementar uno o más aspectos de la presente divulgación para compartir canales de datos entre múltiples etapas de tubería de gráficos. Como se muestra en la FIG. 1, el dispositivo 2 puede ser un dispositivo informático que incluye, pero sin limitarse a, dispositivos de vídeo, reproductores multimedia, descodificadores, teléfonos inalámbricos tales como teléfonos móviles y los denominados teléfonos inteligentes, asistentes digitales personales (PDA), ordenadores de escritorio, ordenadores portátiles, consolas de videojuegos, unidades de videoconferencia, dispositivos informáticos de tableta y similares. En el ejemplo de la FIG. 1, el dispositivo 2 puede

incluir una unidad central de procesamiento (CPU) 6, una memoria de sistema 10 y una GPU 12. El dispositivo 2 también puede incluir un procesador de visualización 14, un módulo transceptor 3, una interfaz de usuario 4 y un visualizador 8. El módulo transceptor 3 y el procesador de visualización 14 pueden formar parte ambos del mismo circuito integrado (IC) que la CPU 6 y/o la GPU 12, pueden ser externos ambos al IC o a los IC que incluyen la CPU 6 y/o la GPU 12, o se pueden formar en un IC que sea externo al IC que incluye la CPU 6 y/o la GPU 12.

[0015] El dispositivo 2 puede incluir módulos o unidades adicionales no mostrados en la FIG. 1 para propósitos de claridad. Por ejemplo, el dispositivo 2 puede incluir un altavoz y un micrófono, de los que no se muestra ninguno en la FIG. 1, para efectuar comunicaciones telefónicas en ejemplos donde el dispositivo 2 es un teléfono móvil inalámbrico, o un altavoz donde el dispositivo 2 es un reproductor multimedia. El dispositivo 2 también puede incluir una cámara de vídeo. Además, los diversos módulos y unidades mostrados en el dispositivo 2 pueden no ser necesarios en cada ejemplo del dispositivo 2. Por ejemplo, la interfaz de usuario 4 y el visualizador 8 pueden ser externos al dispositivo 2 en ejemplos donde el dispositivo 2 es un ordenador de escritorio u otro dispositivo que esté equipado para interconectarse con una interfaz de usuario o visualizador externos.

[0016] Los ejemplos de interfaz de usuario 4 incluyen, pero no se limitan a, una rueda de desplazamiento, un ratón, un teclado y otros tipos de dispositivos de entrada. La interfaz de usuario 4 también puede ser una pantalla táctil y se puede incorporar como parte de un visualizador 8. El módulo transceptor 3 puede incluir circuitos para permitir la comunicación inalámbrica o alámbrica entre el dispositivo informático 2 y otro dispositivo o una red. El módulo transceptor 3 puede incluir moduladores, desmoduladores, amplificadores y otros circuitos de este tipo para la comunicación alámbrica o inalámbrica.

[0017] La CPU 6 puede ser un microprocesador, tal como una unidad central de procesamiento (CPU) configurada para procesar instrucciones de un programa informático para su ejecución. La CPU 6 puede comprender un procesador de propósito general o de propósito especial que controle el funcionamiento del dispositivo informático 2. Un usuario puede proporcionar datos de entrada al dispositivo informático 2 para hacer que la CPU 6 ejecute una o más aplicaciones de software. Las aplicaciones de software que se ejecutan en la CPU 6 pueden incluir, por ejemplo, un sistema operativo, una aplicación de procesador de textos, una aplicación de correo electrónico, una aplicación de hoja de cálculo, una aplicación de reproductor multimedia, una aplicación de videojuegos, una aplicación de interfaz gráfica de usuario u otro programa. Adicionalmente, la CPU 6 puede ejecutar el controlador de GPU 22 para controlar el funcionamiento de la GPU 12. El usuario puede proporcionar datos de entrada al dispositivo informático 2 por medio de uno o más dispositivos de entrada (no mostrados) tales como un teclado, un ratón, un micrófono, un panel táctil u otro dispositivo de entrada que esté acoplado al dispositivo informático 2 por medio de la interfaz de usuario 4.

[0018] Las aplicaciones de software que se ejecutan en la CPU 6 pueden incluir una o más instrucciones de renderización de gráficos que den instrucciones a la CPU 6 para provocar la renderización de datos de gráficos en el visualizador 8. En algunos ejemplos, las instrucciones de software se pueden ajustar a una interfaz de programación de aplicaciones (API) gráficas, tal como, por ejemplo, una API de Biblioteca Gráfica Abierta (OpenGL®), una API de Sistemas Embebidos de Biblioteca Gráfica Abierta (OpenGL ES), una API Direct3D, una API X3D, una API RenderMan, una API WebGL o cualquier otra API gráfica estándar pública o patentada. Para procesar las instrucciones de renderización de gráficos, la CPU 6 puede emitir uno o más comandos de renderización de gráficos a la GPU 12 (por ejemplo, a través del controlador de GPU 22) para provocar que la GPU 12 realice una parte o la totalidad de la renderización de los datos de gráficos. En algunos ejemplos, los datos de gráficos que se van a renderizar pueden incluir una lista de primitivas de gráficos, por ejemplo, puntos, líneas, triángulos, cuadriláteros, tiras de triángulos, etc.

[0019] La GPU 12 se puede configurar para realizar operaciones de gráficos para renderizar una o más primitivas de gráficos en el visualizador 8. Por tanto, cuando una de las aplicaciones de software que se ejecuta en la CPU 6 requiere procesamiento de gráficos, la CPU 6 puede proporcionar comandos de gráficos y datos de gráficos a la GPU 12 para su renderización en el visualizador 8. Los datos de gráficos pueden incluir, por ejemplo, comandos de dibujo, información de estado, información de primitivas, información de textura, etc. La GPU 12 se puede construir, en algunos casos, con una estructura altamente paralela que proporcione un procesamiento más eficaz de operaciones complejas relacionadas con gráficos que la CPU 6. Por ejemplo, la GPU 12 puede incluir una pluralidad de elementos de procesamiento, tales como unidades de sombreador, que están configurados para funcionar en múltiples vértices o píxeles de manera paralela. La naturaleza altamente paralela de la GPU 12, en algunos casos, puede permitir que la GPU 12 dibuje imágenes de gráficos (por ejemplo, GUI y escenas de gráficos bidimensionales (2D) y/o tridimensionales (3D)) en el visualizador 8 más rápidamente que dibujando las escenas directamente en el visualizador 8 usando la CPU 6.

[0020] La GPU 12, en algunos casos, se puede integrar en una placa base del dispositivo informático 2. En otros casos, la GPU 12 puede estar presente en una tarjeta de gráficos que esté instalada en un puerto en la placa base del dispositivo informático 2 o se puede incorporar de otro modo dentro de un dispositivo periférico configurado para funcionar conjuntamente con el dispositivo informático 2. La GPU 12 puede incluir uno o más procesadores, tales como uno o más microprocesadores, circuitos integrados específicos de la aplicación (ASIC), matrices de puertas programables *in situ* (FPGA), procesadores de señales digitales (DSP) u otro circuito lógico integrado o discreto

equivalente. La GPU 12 también puede incluir uno o más núcleos de procesador, de modo que la GPU 12 se puede denominar procesador multinúcleo.

[0021] La GPU 12 se puede acoplar directamente a la memoria gráfica 40. Por tanto, la GPU 12 puede leer datos de y escribir datos en la memoria gráfica 40 sin usar un bus. En otras palabras, la GPU 12 puede procesar datos usando localmente un almacenamiento local, en lugar de una memoria fuera de chip. Dicha memoria gráfica 40 se puede denominar memoria en chip. Esto permite que la GPU 12 funcione de una manera más eficaz al eliminar la necesidad de que la GPU 12 lea y escriba datos por medio de un bus, que puede experimentar un tráfico de bus pesado. Sin embargo, en algunos casos, la GPU 12 puede no incluir una memoria separada, sino que utiliza la memoria de sistema 10 por medio de un bus. La memoria gráfica 40 puede incluir una o más memorias volátiles o no volátiles o dispositivos de almacenamiento, tales como, por ejemplo, memoria de acceso aleatorio (RAM), RAM estática (SRAM), RAM dinámica (DRAM), ROM programable y borrable (EPROM), ROM programable y borrable eléctricamente (EEPROM), memoria Flash, un medio de datos magnético o un medio de almacenamiento óptico.

[0022] En algunos ejemplos, la GPU 12 puede almacenar una imagen completamente formada en la memoria de sistema 10. El procesador de visualización 14 puede recuperar la imagen de la memoria de sistema 10 y emitir valores que hacen que los píxeles del visualizador 8 se iluminen para visualizar la imagen. El visualizador 8 puede el visualizador del dispositivo 2 que visualiza el contenido de imagen generada por la GPU 12. El visualizador 8 puede ser una pantalla de cristal líquido (LCD), una pantalla de diodos orgánicos emisores de luz (OLED), una pantalla de tubo de rayos catódicos (CRT), una pantalla de plasma u otro tipo de dispositivo de visualización.

[0023] De acuerdo con los aspectos de la presente divulgación, la GPU 12 puede ejecutar una pluralidad de etapas de tubería de gráficos en sus unidades de sombreador. La GPU 12 puede crear un canal de datos compartido en la memoria gráfica 40 que se comparte por dos etapas de la pluralidad de etapas de tubería de gráficos que se ejecutan en la unidad de sombreador. La GPU 12 puede almacenar datos producidos por cada una de las dos etapas de la pluralidad de etapas de tubería de gráficos que se ejecutan en unidades de sombreador en el canal de datos compartido.

[0024] La FIG. 2 es un diagrama de bloques que ilustra una tubería de procesamiento gráfico 24 de ejemplo que se puede realizar por la GPU 12 para crear una representación bidimensional de una escena tridimensional. La tubería de procesamiento gráfico 24 puede incluir una pluralidad de etapas de procesamiento gráfico que funcionan en conjunto para ejecutar comandos de procesamiento gráfico. Como se muestra en la FIG. 2, la tubería de procesamiento gráfico 24 puede incluir el ensamblador de entrada 26, la etapa de sombreador de vértices 28, la etapa de sombreador de casco 30, la etapa de teselador 32, la etapa de sombreador de dominios 34, la etapa de sombreador de geometría 36 y la etapa de sombreador de píxeles 38. Cada uno de los componentes en la tubería de procesamiento gráfico 24 se puede implementar como componentes de función fija, componentes programables (por ejemplo, como parte de un programa de sombreador que se ejecuta en una unidad de sombreador programable), o como una combinación de componentes de función fija y programables.

[0025] La GPU 12 se puede configurar para recibir uno o más comandos de procesamiento gráfico desde la CPU 6, por medio del controlador de GPU 22, y para ejecutar los comandos de procesamiento gráfico por medio de la tubería de procesamiento gráfico 24 para generar imágenes de gráficos visualizables. Como se analiza anteriormente, la tubería de procesamiento gráfico 24 incluye una pluralidad de etapas que funcionan en conjunto para ejecutar comandos de procesamiento gráfico.

[0026] El ensamblador de entrada 26 en la tubería de procesamiento gráfico 24 puede ser una etapa de función fija que, en general, es responsable de suministrar datos gráficos (por ejemplo, triángulos, líneas y puntos) a la tubería de procesamiento gráfico 24. Por ejemplo, la etapa de ensamblador de entrada 26 puede recopilar datos de vértice para superficies de alto orden, primitivas y similares, y emitir datos y atributos de vértice en la etapa de sombreador de vértices 28. En consecuencia, la etapa de ensamblador de entrada 26 puede leer los vértices de una memoria fuera de chip, tal como la memoria de sistema 10, usando operaciones de función fija. A continuación, la etapa de ensamblador de entrada 26 puede crear elementos de trabajo de tubería a partir de estos vértices, mientras que también genera identificadores de vértice ("VertexID"), identificadores de instancia ("InstanceID", que están disponibles para el sombreador de vértices) e identificadores de primitiva ("PrimativeID", que están disponibles para el sombreador de geometría y el sombreador de píxeles). La etapa de ensamblador de entrada 26 puede generar automáticamente los VertexID, InstanceID y PrimitiveID tras leer los vértices.

[0027] La etapa de sombreador de vértices 28 puede procesar los datos y atributos de vértice recibidos. Por ejemplo, la etapa de sombreador de vértices 28 puede realizar un procesamiento por vértice tal como transformaciones, animación esquelética (*skinning*), desplazamiento de vértices y cálculo de atributos de material por vértice. En algunos ejemplos, la etapa de sombreador de vértices 28 puede generar coordenadas de textura, color de vértice, iluminación de vértice, factores de niebla y similares. La etapa de sombreador de vértices 28, en general, toma un único vértice de entrada y emite un único vértice de salida procesado.

[0028] La etapa de sombreador de casco 30, el teselador 32 y la etapa de sombreador de dominios 34 se pueden denominar conjuntamente etapas de teselación. Las etapas de teselación convierten las superficies de subdivisión de

bajo detalle en primitivas de mayor detalle, y organiza superficies de alto orden en superficies adecuadas (por ejemplo, triángulos) para su renderización. La etapa de sombreador de casco 30 recibe primitivas de la etapa de sombreador de vértices 28 y es responsable de llevar a cabo al menos dos acciones. Primero, la etapa de sombreador de casco 30 es típicamente responsable de determinar un conjunto de factores de teselación. La etapa de sombreador de casco 30 puede generar factores de teselación una vez por primitiva. Los factores de teselación se pueden usar en la etapa de teselador 32 para determinar qué tan finamente teselar una primitiva dada (por ejemplo, dividir la primitiva en partes más pequeñas). La etapa de sombreador de casco 30 también es responsable de generar puntos de control que luego se usarán por la etapa de sombreador de dominios 34. Es decir, por ejemplo, la etapa de sombreador de casco 30 es responsable de generar puntos de control que se usarán por la etapa de sombreador de dominio 34 para crear vértices teselados reales, que finalmente se usan en el renderizado.

[0029] Cuando la etapa de teselador 32 recibe datos de la etapa de sombreador de casco 30, la etapa de teselador 32 usa uno de varios algoritmos para determinar un patrón de muestreo apropiado para el tipo de primitiva actual. Por ejemplo, en general, la etapa de teselador 32 convierte una cantidad solicitada de teselación (como se determina por la etapa de sombreador de casco 30) en un grupo de puntos de coordenadas dentro de un "dominio" actual. Es decir, dependiendo de los factores de teselación de la etapa de sombreador de casco 30, así como de la configuración particular de la etapa de teselador 32, la etapa de teselador 32 determina qué puntos de una primitiva actual es necesario muestrear para teselar la primitiva de entrada en partes más pequeñas. La salida de la etapa de teselador 32 puede ser un conjunto de puntos de dominio, que puede incluir coordenadas baricéntricas.

[0030] La etapa de sombreador de dominios 34 toma los puntos de dominio, además de los puntos de control producidos por la etapa de sombreador de casco 30, y usa los puntos de dominio para crear nuevos vértices. La etapa de sombreador de dominios 34 puede usar la lista completa de puntos de control generados para la primitiva actual, texturas, algoritmos de procedimiento o cualquier otra cosa, para convertir la "localización" baricéntrica de cada punto teselado en la geometría de salida que se pasa a la siguiente etapa en la tubería.

[0031] La etapa de sombreador de geometría 36 puede recibir una primitiva definida por sus datos de vértice (por ejemplo, tres vértices para un triángulo, dos vértices para una línea o un único vértice para un punto) y procesar además la primitiva. Por ejemplo, la etapa de sombreador de geometría 36 puede realizar un procesamiento por primitiva tal como detección de borde de silueta y extrusión de volumen de sombra, entre otras posibles operaciones de procesamiento. En consecuencia, la etapa de sombreador de geometría 36 puede recibir una primitiva como entrada (que puede incluir uno o más vértices) y emite cero, una o múltiples primitivas (que nuevamente pueden incluir uno o más vértices). La primitiva de salida puede contener más datos de los que pueden ser posibles sin la etapa de sombreador de geometría 36. La cantidad total de datos de salida puede ser igual al tamaño de vértice multiplicado por el recuento de vértices, y puede estar limitado por invocación. La salida de flujo desde la etapa de sombreador de geometría 36 puede permitir que las primitivas que llegan a esta etapa se almacenen en la memoria fuera de chip, tal como la memoria de sistema 10. La salida de flujo está típicamente vinculada a la etapa de sombreador de geometría 36, y ambas se pueden programar conjuntamente (por ejemplo, usando una API).

[0032] La etapa de rasterizador 37 es típicamente una etapa de función fija que es responsable de recortar las primitivas y preparar las primitivas para la etapa de sombreador de píxeles 38. Por ejemplo, la etapa de rasterizador 37 puede realizar recortes (incluyendo límites de recorte personalizados), división de perspectiva, selección e implementación de ventana gráfica/tijera, selección de destino de renderizado y configuración de primitiva. De esta manera, la etapa de rasterizador 37 puede generar una serie de fragmentos para sombrear por la etapa de sombreador de píxeles 38.

[0033] La etapa de sombreador de píxeles 38 recibe fragmentos desde la etapa de rasterizador 37 y genera datos por píxel, tales como el color. La etapa de sombreador de píxeles 38 también puede realizar un procesamiento por píxel, tal como combinación de texturas y cálculo de modelo de iluminación. En consecuencia, la etapa de sombreador de píxeles 38 puede recibir un píxel como entrada y puede emitir un píxel en la misma posición relativa (o un valor cero para el píxel).

[0034] De acuerdo con los aspectos de la presente divulgación, dos o más etapas de la tubería de procesamiento gráfico 24 pueden compartir un canal de datos compartido en la memoria gráfica 40. Por ejemplo, los vértices producidos por la etapa de sombreador de vértices 28 y la etapa de sombreador de dominios 34 se pueden almacenar en un canal de datos compartido. Además, las primitivas producidas por la etapa de sombreador de casco 30 y la etapa de sombreador de geometría 36 se pueden almacenar en otro canal de datos compartido. De esta manera, la GPU 12 puede utilizar más eficazmente la memoria gráfica 40.

[0035] La FIG. 3 es un diagrama de bloques que ilustra con mayor detalle implementaciones de ejemplo de la CPU 6, la GPU 12 y la memoria de sistema 10 de la FIG. 1. Como se muestra en la FIG. 3, la CPU 6 puede incluir al menos una aplicación de software 18, una API gráfica 20 y un controlador de GPU 22, de los que cada uno puede ser una o más aplicaciones de software o servicios que se ejecutan en la CPU 6.

[0036] La memoria disponible para la CPU 6 y la GPU 12 puede incluir la memoria de sistema 10 y la memoria intermedia de tramas 16. La memoria intermedia de tramas 16 puede formar parte de la memoria de sistema 10 o

puede estar separada de la memoria de sistema 10. La memoria intermedia de tramas 16 puede almacenar datos de imagen renderizados.

[0037] La aplicación de software 18 puede ser cualquier aplicación que utiliza la funcionalidad de la GPU 12. Por ejemplo, la aplicación de software 18 puede ser una aplicación de GUI, un sistema operativo, una aplicación de correlación portátil, un programa de diseño asistido por ordenador para aplicaciones de ingeniería o artísticas, una aplicación de videojuegos u otro tipo de aplicación de software que usa gráficos 2D o 3D.

[0038] La aplicación de software 18 puede incluir una o más instrucciones de dibujo que dan la instrucción a la GPU 12 de renderizar una interfaz gráfica de usuario (GUI) y/o una escena gráfica. Por ejemplo, las instrucciones de dibujo pueden incluir instrucciones que definan un conjunto de una o más primitivas de gráficos que se van a renderizar por la GPU 12. En algunos ejemplos, las instrucciones de dibujo, conjuntamente, pueden definir la totalidad o parte de una pluralidad de superficies de ventanas usadas en una GUI. En ejemplos adicionales, las instrucciones de dibujo, conjuntamente, pueden definir la totalidad o parte de una escena de gráficos que incluya uno o más objetos de gráficos dentro de un espacio modelo o espacio mundial definido por la aplicación.

[0039] La aplicación de software 18 puede invocar al controlador de GPU 22, por medio de la API de gráficos 20, para emitir uno o más comandos a la GPU 12 para renderizar una o más primitivas de gráficos en imágenes de gráficos visualizables. Por ejemplo, la aplicación de software 18 puede invocar al controlador de GPU 22, por medio de la API de gráficos 20, para proporcionar definiciones de primitivas a la GPU 12. En algunos casos, las definiciones de primitivas se pueden proporcionar a la GPU 12 en forma de una lista de primitivas de dibujo, por ejemplo, triángulos, rectángulos, abanicos de triángulos, tiras de triángulos, etc. Las definiciones de primitivas pueden incluir especificaciones de vértices que especifiquen uno o más vértices asociados a las primitivas que se vayan a renderizar. Las especificaciones de vértices pueden incluir coordenadas de posición para cada vértice y, en algunos casos, otros atributos asociados al vértice, tales como, por ejemplo, coordenadas de color, vectores normales y coordenadas de textura. Las definiciones de primitivas también pueden incluir información de tipo de primitiva (por ejemplo, triángulo, rectángulo, abanico de triángulos, tira de triángulos, etc.), información de escalado, información de rotación y similares. En base a las instrucciones emitidas por la aplicación de software 18 al controlador de GPU 22, el controlador de GPU 22 puede formular uno o más comandos que especifiquen una o más operaciones para que la GPU 12 la(s) realice para renderizar la primitiva. Cuando la GPU 12 recibe un comando desde la CPU 6, la tubería de procesamiento gráfico 24 descodifica el comando y configura la tubería de procesamiento gráfico 24 para realizar la operación especificada en el comando. Por ejemplo, el ensamblador de entrada 26 en la tubería de procesamiento gráfico 24 puede leer los datos de primitiva y ensamblar los datos en primitivas para su uso por las otras etapas de tubería de gráficos en la tubería de procesamiento gráfico 24. Después de realizar las operaciones especificadas, la tubería de procesamiento gráfico 24 emite los datos renderizados a la memoria intermedia de tramas 16 asociada a un dispositivo de visualización.

[0040] La memoria intermedia de tramas 16 almacena píxeles de destino para la GPU 12. Cada píxel de destino puede estar asociado a una localización única de píxel en pantalla. En algunos ejemplos, la memoria intermedia de tramas 16 puede almacenar componentes de color y un valor alfa de destino para cada píxel de destino. Por ejemplo, la memoria intermedia de tramas 16 puede almacenar componentes Rojo, Verde, Azul, Alfa (RGBA) para cada píxel, donde los componentes "RGB" corresponden a valores de color y el componente "A" corresponde a un valor alfa de destino. Aunque la memoria intermedia de tramas 16 y la memoria de sistema 10 se ilustran como unidades de memoria separadas, en otros ejemplos, la memoria intermedia de tramas 16 puede formar parte de la memoria de sistema 10.

[0041] En algunos ejemplos, la etapa de sombreador de vértices 28, la etapa de sombreador de casco 30, la etapa de sombreador de dominios 34, la etapa de sombreador de geometría y la etapa de sombreador de píxeles 38 de la tubería de procesamiento gráfico 24 se pueden considerar etapas de sombreador. Estas etapas de sombreador se pueden implementar como uno o más programas de sombreador que se ejecutan en las unidades de sombreador 46 en la GPU 12. Las unidades de sombreador 46 se pueden configurar como una tubería programable de componentes de procesamiento. En algunos ejemplos, la unidad de sombreado 46 se puede denominar "procesadores de sombreador" o "sombreadores unificados", y puede realizar operaciones de sombreado de geometría, vértices, píxeles u otras para renderizar gráficos. Las unidades de sombreador 46 pueden incluir núcleos de procesador 48, de los que cada uno puede incluir uno o más componentes para buscar y descodificar operaciones, una o más unidades lógicas aritméticas para llevar a cabo cálculos aritméticos, una o más memorias, memorias caché y registros.

[0042] La GPU 12 puede designar unidades de sombreador 46 para realizar una variedad de operaciones de sombreado, tales como sombreado de vértices, sombreado de casco, sombreado de dominios, sombreado de geometría, sombreado de píxeles y similares, enviando comandos a las unidades de sombreador 46 para ejecutar una o más etapas de sombreador de vértices 28, etapas de sombreador de casco 30, etapas de sombreador de dominios 34, etapas de sombreador de geometría 36 y etapas de sombreador de píxeles 38 en la tubería de procesamiento gráfico 24. En algunos ejemplos, el accionador de GPU 22 se puede configurar para compilar uno o más programas de sombreador y para descargar los programas de sombreador compilados en una o más unidades de sombreador programables contenidas dentro de la GPU 12. Los programas de sombreador se pueden escribir en un lenguaje de sombreado de alto nivel, tal como, por ejemplo, un Lenguaje de Sombreado OpenGL (GLSL), un

Lenguaje de Sombreado de Alto Nivel (HLSL), un lenguaje de sombreado C para Gráficos (Cg), etc. Los programas de sombreador compilados pueden incluir una o más instrucciones que controlen el funcionamiento de las unidades de sombreador 46 dentro de la GPU 12. Por ejemplo, los programas de sombreador pueden incluir programas de sombreador de vértices que se pueden ejecutar por las unidades de sombreador 46 para realizar las funciones de la etapa de sombreador de vértices 28, programas de sombreador de casco que se pueden ejecutar por las unidades de sombreador 46 para realizar las funciones de etapa de sombreador de casco 30, programas de sombreador de dominios que se pueden ejecutar por la unidad de sombreador 46 para realizar las funciones de etapa de sombreador de dominios 34, programas de sombreador de geometría que se pueden ejecutar por la unidad de sombreador 46 para realizar las funciones de etapa de sombreador de geometría 36 y/o programas de sombreador de píxeles que se pueden ejecutar por las unidades de sombreador 46 para realizar las funciones de sombreador de píxeles 38. Un programa de sombreador de vértices puede controlar la ejecución de una unidad de sombreador de vértices programable o una unidad de sombreador unificada e incluir instrucciones que especifiquen una o más operaciones por vértice.

[0043] La memoria gráfica 40 es almacenamiento o memoria en chip que se integró físicamente en el circuito integrado de la GPU 12. Debido a que la memoria gráfica 40 está en chip, la GPU 12 puede leer valores de o escribir valores en la memoria gráfica 40 más rápidamente que leer valores de o escribir valores en la memoria de sistema 10 por medio de un bus de sistema. Como tal, el rendimiento de las unidades de sombreador 46 se puede incrementar almacenando y leyendo datos producidos y consumidos por las etapas de sombreador de la tubería de procesamiento gráfico 24 desde la memoria gráfica 40.

[0044] De acuerdo con los aspectos de la presente divulgación, las unidades de sombreador 46 pueden realizar múltiples operaciones de sombreado al mismo tiempo en los núcleos de procesador 48. La GPU 12 puede enviar comandos a la unidad de sombreado 46 que posibilita que se ejecuten diferentes etapas de sombreado de la tubería de procesamiento gráfico 24 en diferentes núcleos de procesador 48, intercalando de este modo las etapas de la tubería de procesamiento gráfico 24. Por ejemplo, la GPU 12 puede enviar comandos a la unidad de sombreado 46 que hace que la unidad de sombreado 46 ejecute la etapa de sombreador de vértices 28 y la etapa de sombreador de geometría 36 al mismo tiempo en diferentes núcleos de procesador 48 de las unidades de sombreador 46. En otro ejemplo, la GPU 12 puede enviar comandos a la unidad de sombreado 46 que hace que la unidad de sombreado 46 ejecute múltiples instancias de la etapa de sombreador de geometría 36 al mismo tiempo en múltiples procesadores.

[0045] De acuerdo con los aspectos de la presente divulgación, la memoria gráfica 40 puede incluir uno o más canales de datos compartidos 50A-50N ("canales de datos compartidos 50") que posibilitan que los datos producidos por diferentes etapas de la tubería de procesamiento gráfico 24 compartan un único canal de datos, posibilitando de este modo que la GPU 12 utilice más eficazmente el espacio limitado en la memoria gráfica 40 y también posibilitando que el grupo de procesadores de sombreador 46 incremente la utilización de sus núcleos de procesador 48 para ejecutar simultáneamente múltiples etapas de la tubería de procesamiento gráfico 24.

[0046] Cada canal de datos compartido en los canales de datos compartidos 50 puede almacenar datos producidos por las dos o más etapas de la tubería de procesamiento gráfico 24. Al compartir un canal de datos compartido en los canales de datos compartidos 50, a diferencia de asignar canales de datos para etapas individuales de la tubería de procesamiento gráfico 24, si una etapa en la tubería de procesamiento gráfico 24 produce menos datos, entonces otra etapa que comparte el mismo canal de datos compartido puede aprovechar ese hecho almacenando más de los datos que produce en el canal de datos compartido.

[0047] De acuerdo con un aspecto de la presente divulgación, la unidad de procesamiento de geometría (GPC) 42 puede programar la ejecución del grupo de procesadores de sombreador 46 en base al estado de los canales de datos compartidos 50. La GPC 42 puede supervisar los canales de datos compartidos 50 para determinar si hay suficientes datos en los canales de datos compartidos 50 que se van a consumir por las etapas de la tubería de procesamiento gráfico 24 que se van a ejecutar por el grupo de procesadores de sombreador 46. La GPC 42 también puede supervisar los canales de datos compartidos 50 para determinar si hay suficiente espacio libre en los canales de datos compartidos 50 para almacenar datos producidos por las etapas de la tubería de procesamiento gráfico 24 que se van a ejecutar por el grupo de procesadores de sombreador 46. Si la GPC 42 determina que hay suficientes datos y espacio libre en los canales de datos compartidos 50, la GPC 42 puede enviar comandos de ejecución al grupo de procesadores de sombreador 46 para ejecutar un lote de etapas de la tubería de procesamiento gráfico 24. En respuesta a completar la ejecución del lote de etapas, el grupo de procesadores de sombreador 46 puede enviar una señal a la GPC 42 que indica que el grupo de procesadores 46 ha completado la ejecución del lote de etapas. En respuesta, el gestor de canales de datos 44 puede actualizar los punteros de lectura y escritura relevantes para los canales de datos compartidos 50. La GPC 42 puede incluir el gestor de canales de datos 44 que gestiona los canales de datos compartidos 50. El gestor de canales de datos 44 puede gestionar los punteros de lectura y escritura para los canales de datos compartidos 50 que apuntan a localizaciones dentro de los canales de datos compartidos 50 para escribir datos en y leer datos de los canales de datos compartidos 50.

[0048] De acuerdo con los aspectos de la presente divulgación, el canal de datos compartido 50A puede ser un canal de datos que se comparte por dos o más etapas de la tubería de procesamiento gráfico 24, de modo que el canal de datos compartido 50A pueda almacenar tanto los datos 55A emitidos por una primera etapa de la tubería de

procesamiento gráfico 24 como los datos 55B emitidos por una segunda etapa de la tubería de procesamiento gráfico 24. El canal de datos compartido 50A puede ser una memoria intermedia cíclica de modo que los datos 55A y 55B puedan ambos incrementar y disminuir dinámicamente su tamaño cuando se producen y/o consumen, permitiendo de este modo un uso más eficaz del bloque de memoria asignado al canal de datos compartido 50A. La GPC 42 puede gestionar los punteros de escritura 51A y 51B y los punteros de lectura 53A y 53B. El puntero de escritura 51A puede apuntar a la localización de memoria del canal de datos compartido 50A para escribir datos 55A, y el puntero de lectura 53A puede apuntar a la localización de memoria del canal de datos compartido 50A del que leer los datos 55A.

[0049] Típicamente, la GPU 12 almacena los datos 55A y 55B en el canal de datos compartido 50A en orden de primero en entrar, primero en salir (FIFO), de modo que los punteros de lectura 53A y 53B apunten a las localizaciones de memoria del canal de datos compartido 50A que almacena la porción de datos más antigua en los datos 55A y 55B, respectivamente, a veces denominada la cabeza de la cola, y de modo que los punteros de escritura 51A y 51B apunten a las localizaciones de memoria del canal de datos compartido 50A que almacena la porción de datos más reciente en los datos 55A y 55B, respectivamente, a veces denominada la cola de la cola.

[0050] El canal de datos compartido 50A también puede funcionar en modo FIFO de modo que los datos leídos de los datos 55A y 55B se eliminen del canal de datos compartido 50A y esas localizaciones de memoria se puedan desasignar. Como se puede ver, cuando la GPU 12 lee los datos 55A del canal de datos compartido 50A, el espacio libre 57 en el canal de datos compartido 50A se incrementa, dejando de este modo espacio adicional en el canal de datos compartido 50A para que la GPU 12 escriba datos en los datos 55B. De forma similar, cuando la GPU 12 lee los datos 55B del canal de datos compartido 50A, el espacio libre 59 en el canal de datos compartido 50A se incrementa, dejando de este modo espacio adicional en el canal de datos compartido 50A para que la GPU 12 escriba datos en los datos 55A. Aunque solo se ha descrito en detalle anteriormente el canal de datos compartido 50A, se debe entender que cada canal de datos compartido en los canales de datos compartidos 50 puede compartir los rasgos característicos descritos anteriormente con respecto al canal de datos compartido 50A.

[0051] La FIG. 4 es un diagrama de bloques que ilustra un ejemplo de canales de datos compartidos 50 en uso en la tubería de procesamiento gráfico 24. Como se muestra en la FIG. 4, el canal de datos compartido 50A se puede ser compartir por las etapas de la tubería de procesamiento gráfico 24 para almacenar datos producidos por las etapas.

Específicamente, el canal de datos compartido 50A puede almacenar datos 52 producidos por la etapa de sombreador de casco 30 de la tubería de procesamiento gráfico 24 y puede almacenar además datos 54 producidos por la etapa de sombreador de geometría 36 de la tubería de procesamiento gráfico 24. Los datos 52 se pueden consumir por la etapa de sombreador de dominios 34 de la tubería de procesamiento gráfico 24 y los datos 54 se pueden consumir por la etapa de sombreador de píxeles de la tubería de procesamiento gráfico 24.

[0052] Los datos 52 y los datos 54 almacenados en el canal de datos compartido 50A por la etapa de sombreador de casco 30 y la etapa de sombreador de geometría 36 pueden incluir puntos de control de parche que se emiten por la etapa de sombreador de casco 30 y vértices que se emiten por la etapa de sombreador de geometría 36, respectivamente. Debido a que el canal de datos 50A no almacena en caché los datos 52 y 54, los datos 52 y 54 pueden actuar cada uno como una cola FIFO en la que los datos leídos de los datos 52 y 54 se eliminan del canal de datos compartido 50A.

[0053] En algunos ejemplos, los mismos datos producidos por algunas etapas de la tubería de procesamiento gráfico 24 se pueden consumir varias veces por otras etapas de la tubería de procesamiento gráfico 24. Si los datos se almacenaron en uno de los canales de datos compartidos 50 que actúa como una cola FIFO, puede ser necesario ejecutar varias veces las etapas de la tubería de procesamiento gráfico 24 que producen los datos para producir los mismos datos porque los datos almacenados en una cola FIFO se pueden eliminar cuando se leen de la cola FIFO. En lugar de ejecutar el sombreador de vértices 28 o el sombreador de dominios 34 varias veces para producir el mismo vértice varias veces, la GPU 12 puede en su lugar almacenar en caché los datos producidos por el sombreador de vértices 28 y el sombreador de dominios 34 en el canal compartido en modo caché 56.

[0054] Por ejemplo, los datos producidos por la etapa de sombreador de vértices 28 de la tubería de procesamiento gráfico 24, incluyendo los vértices transformados por la etapa de sombreador de vértices 28, se pueden consumir por la etapa de sombreador de casco 30 de la tubería de procesamiento gráfico 24. De forma similar, los datos producidos por la etapa de sombreador de dominio 34 de la tubería de procesamiento gráfico 24, tales como las posiciones de vértice emitidas por la etapa de sombreador de dominios 34, se pueden consumir por la etapa de sombreador de geometría 36 de la tubería de procesamiento gráfico 24. Por ejemplo, debido a que las primitivas adyacentes (por ejemplo, triángulos) pueden compartir vértices, se puede usar el mismo vértice para formar dos triángulos adyacentes. Por tanto, los datos de vértice producidos por la etapa de sombreador de vértices 28 y la etapa de sombreador de dominios 34 se pueden consumir varias veces. Debido a que los datos producidos por la etapa de sombreador de vértices 28 y la etapa de sombreador de dominios 34 se pueden consumir varias veces, los datos producidos por estas etapas se pueden almacenar en caché en el canal compartido en modo caché 56, de modo que los datos almacenados en caché no se puedan eliminar en respuesta a su lectura desde el canal compartido en modo de caché 56.

[0055] La FIG. 5 es un diagrama de bloques que ilustra el canal compartido en modo caché 56. Como se muestra en la FIG. 5, el canal compartido en modo caché 56 puede incluir dos canales de datos compartidos: la cola de primitivas compartida 50B y la memoria caché de vértices compartida 50C, así como la ventana de memoria caché 70. La memoria caché de vértices compartida 50C puede funcionar en modo de memoria caché, de modo que los datos almacenados en la memoria caché de vértices compartida 50C no se pueden eliminar tras leerlos de la memoria caché de vértices compartida 50C. Los datos 62 y los datos 64 almacenados en la cola de primitivas compartida 50B pueden incluir datos de primitivas producidos por la etapa de sombreador de vértices 28 y la etapa de sombreador de dominios 34. Por ejemplo, los datos 62 pueden incluir índices de vértices y las localizaciones de datos de vértices almacenados en la memoria caché de vértices compartida 50C producida por la etapa de sombreador de vértices 28 para cada primitiva, y los datos 64 pueden incluir índices de vértices y las localizaciones de datos de vértices almacenados en la memoria caché de vértices compartida 50C producida por la etapa de sombreador de dominios 34 para cada primitiva. Los datos 62 y 64 también pueden incluir indicadores de desasignación para cada una de las primitivas asociadas. Los datos 66 almacenados en la memoria caché de vértices compartida 50C pueden incluir vértices transformados por la etapa de sombreador de vértices 28, mientras que los datos 68 almacenados en la memoria caché de vértices compartida 50C pueden incluir las posiciones de vértices emitidas por la etapa de sombreador de dominios 34. La GPC 42 puede verificar el espacio libre tanto de la cola de primitivas compartida 50B como de la memoria caché de vértices compartida 50C para determinar si el canal compartido en modo caché 56 tiene suficiente espacio libre para aceptar datos.

[0056] La ventana de memoria caché 70 puede almacenar una indicación si un vértice particular ya está almacenado en una ventana limitada de memoria caché de vértices compartida 50C. Por ejemplo, la ventana de memoria caché 70 puede actuar como memoria caché totalmente asociativa y almacenar el índice de vértices, la localización de datos del vértice dentro de la memoria caché de vértices compartida 50C, y una indicación, tal como un indicador, del sombreador que puede consumir el vértice.

[0057] La GPC 42 procesa geometría de primitiva a primitiva. Para el sombreador de vértices 28 y el sombreador de dominio 34, si la GPC 42 determina, en base a la verificación de la ventana de memoria caché 70 en el índice de vértices, y/o el sombreador al que pertenece el vértice, que el vértice particular de una primitiva no está en la memoria caché de vértices compartida 50C, se puede producir un error de memoria caché, y la GPC 42 puede enviar comandos a las unidades de sombreador 46 para ejecutar la etapa de sombreador apropiada (por ejemplo, el sombreador de vértices 28 o el sombreador de dominios 34) para producir el vértice deseado y para almacenar los datos de vértice producidos en el canal compartido en modo caché 56. La GPC 42 puede añadir índices de vértices y localizaciones de datos de vértices en la memoria caché de vértices compartida 50C en la cola de primitivas compartida 50B. La GPC 42 puede añadir a la ventana de memoria caché 70 con los datos apropiados para el vértice ahora almacenados en caché en el canal compartido en modo caché 56. La ventana de memoria caché 70 puede actuar de forma de primero en entrar, primero en salir (FIFO) de modo que si no hay espacio en la ventana de memoria caché 70 después de un error de memoria caché, a continuación, el vértice asociado en la ranura más antigua en la ventana de memoria caché 70 y que tiene su indicador de desasignación configurado en la cola de primitivas compartida 50B se puede configurar con la información con respecto al vértice más reciente añadido al canal compartido en modo caché 56. Sin embargo, si la GPC 42 determina que el vértice particular está almacenado en caché en el canal compartido en modo caché 56, la GPC 42 puede usar la localización de memoria en la memoria caché de vértices compartida 50C del vértice deseado y añadir índices de vértices y localizaciones de datos de vértice en la memoria caché de vértices compartida 50C en la cola de primitiva compartida 50B. De esta manera, la GPU 12 puede reducir el procesamiento superfluo de etapas en la tubería de procesamiento gráfico 24.

[0058] Para ejecutar el sombreador de casco 30 y el sombreador de geometría 36, la GPC 42 puede consumir datos tanto de la cola de primitivas compartida 50B como de la memoria caché de vértices compartida 50C. La GPC 42 puede leer índices de vértices y localizaciones de datos de vértice en la memoria caché de vértices compartida 50C desde la cola de primitivas compartida 50B. A continuación, la GPC 42 puede leer datos de vértice desde la memoria caché de vértices compartida 50C usando localizaciones leídas desde la cola de primitivas compartida 50B. La GPC 42 puede mover el puntero de lectura de la cola de primitivas compartida 50B después de leer los datos. Sin embargo, debido a que la siguiente primitiva también puede usar el mismo vértice que se acaba de leer de la memoria caché de vértices compartida 50C, la GPC 42 puede no mover inmediatamente el puntero de lectura de la memoria caché de vértices compartida 50C justo después de que se lea el vértice almacenado en caché desde la memoria caché de vértices compartida 50C. Si se establece el indicador de desasignación asociado en la cola de primitivas compartida 50B para la primitiva que consume el vértice, entonces se puede permitir que la GPC 42 mueva los punteros de lectura y desasigne el vértice del canal compartido en modo caché 56. La GPC 42 puede enviar comandos a las unidades de sombreador 46 para ejecutar la etapa de sombreador (por ejemplo, el sombreador de casco 30 y el sombreador de geometría 36) para consumir los datos de vértice, y para producir el vértice para la siguiente etapa de sombreador y para almacenar los datos de vértice producidos en el canal de datos compartido 50A.

[0059] La GPC 42 puede supervisar el canal compartido en modo caché 56 y el canal de datos compartido 50A para el punto muerto. En un ejemplo, se puede producir un punto muerto si el canal compartido en modo caché 56 está lleno de datos producidos por la etapa de sombreador de vértices 28, y si el canal de datos compartido 50A está lleno de datos producidos por la etapa de sombreador de casco 30. En este caso, debido a que la etapa de sombreador de casco 30 consume datos producidos por la etapa de vértices 28, la etapa de sombreador de casco 30 no puede

consumir datos producidos por la etapa de sombreador de vértices 28 y almacenados en el canal compartido en modo caché 56 para producir datos que se almacenan en el canal de datos compartido 50A porque no hay espacio libre en el canal de datos compartido 50A para almacenar los datos recién producidos. Además, debido a que el canal compartido en modo caché 56 está lleno de datos producidos por la etapa de sombreador de vértices 28, y ninguno de esos datos se puede consumir por el sombreador de casco 30, ninguno de esos datos se puede desasignar para liberar espacio para el canal compartido en modo caché 56 para almacenar datos producidos por el sombreador de dominios 34. Además, debido a que el sombreador de dominios 34 consume datos producidos por la etapa de sombreador de casco 30 y almacenados en el canal de datos compartido 50A, ninguno de los datos producidos por el sombreador de casco 30A y almacenados en el canal de datos compartido 50A se puede consumir por el sombreador de dominios 34 para liberar espacio en el canal de datos compartido 50A para el canal de datos compartido 50A para almacenar datos producidos por el sombreador de geometría 36.

[0060] Para evitar situaciones de punto muerto entre el canal compartido en modo caché 56 y el canal de datos compartido 50A, la GPC 42 puede reservar espacio en el canal compartido en modo caché 56 y el canal de datos compartido 50A para almacenar datos producidos por el sombreador de dominios 34 y el sombreador de geometría 36, respectivamente, de modo que el canal compartido en modo caché 56 y el canal de datos compartido 50A no solo almacenen datos producidos por el sombreador de vértices 28 y el sombreador de casco 30, respectivamente. La GPC 42 puede determinar la cantidad de espacio del canal compartido en modo caché 56 tanto en la cola de primitivas compartida 50B como en la memoria caché de vértices compartida 50C de componentes, y la cantidad de espacio del canal de datos compartido 50A para reservar, por ejemplo, determinando la cantidad de espacio necesario para almacenar la salida del sombreador de dominios 34 y el sombreador de geometría 36 para un número dado de ondas en el grupo de sombreadores 46.

[0061] La FIG. 6 es un diagrama de flujo que ilustra un proceso de ejemplo para compartir canales de datos por etapas de una tubería de procesamiento gráfico. Como se muestra en la FIG. 6, el proceso puede incluir asignar, por la GPU 12, el canal de datos compartido 50A en la memoria gráfica en chip 40 de la GPU 12 que se comparte por al menos dos etapas de la tubería de procesamiento gráfico 24 (502). El proceso puede incluir además ejecutar, en las unidades de sombreador 46 en la GPU 12, las al menos dos etapas de la tubería de procesamiento gráfico 24 (504). El proceso puede incluir además almacenar, por la GPU 12 en el canal de datos compartido 50A en la memoria gráfica en chip 40, datos producidos por las al menos dos etapas de la tubería de procesamiento gráfico 24 que se ejecutan en las unidades de sombreador 46 (506).

[0062] En algunos ejemplos, el proceso puede incluir además asignar, por la GPU 12, un segundo canal compartido en modo caché 56 en la memoria gráfica en chip 40 de la GPU 12 que se comparte por unas segundas al menos dos etapas de tubería de procesamiento gráfico 24, en el que el canal de datos compartido 50A es un primer canal de datos compartido. En algunos ejemplos, el proceso puede incluir además ejecutar, en las unidades de sombreador 46 en la GPU 12, las segundas al menos dos etapas de la tubería de procesamiento gráfico 24. En algunos ejemplos, el proceso puede incluir además almacenar, por la GPU 12 en el segundo canal compartido en modo caché 56, los segundos datos producidos por cada una de las segundas al menos dos etapas de la tubería de procesamiento gráfico 24 que se ejecutan en las unidades de sombreador 46.

[0063] En algunos ejemplos, el proceso puede incluir además programar, por la GPU 12, la ejecución de una o más etapas de la tubería de procesamiento gráfico 24 por las unidades de sombreador 46 en base al menos en parte a un estado del primer canal de datos compartido 50A o el segundo canal compartido en modo caché 56 de modo que los datos estén disponibles en el primer canal de datos compartido 50A o el segundo canal compartido en modo caché 56 para consumirse por las una o más etapas de la tubería de procesamiento gráfico 24 cuando se ejecuten en las unidades de sombreador 46 y el espacio libre esté disponible en el primer canal de datos compartido 50A o el segundo canal compartido en modo caché 56 para almacenar datos producidos por las una o más etapas de la tubería de procesamiento gráfico 24 cuando se ejecuten en las unidades de sombreador 46.

[0064] En algunos ejemplos, las al menos dos etapas de la tubería de procesamiento gráfico 24 incluyen el sombreador de vértices 28 y el sombreador de dominio 34. En algunos ejemplos, las segundas al menos dos etapas de la tubería de procesamiento gráfico 24 incluyen el sombreador de casco 30 y un sombreador de geometría 36.

[0065] En algunos ejemplos, el proceso puede incluir además reservar, por la GPU 12, espacio libre en al menos uno del primer canal de datos compartido 50A y el segundo canal compartido en modo caché 56 para evitar el punto muerto entre el primer canal de datos compartido 50A y el segundo canal compartido en modo caché 56.

[0066] En uno o más ejemplos, las funciones descritas se pueden implementar en hardware, software, firmware o en cualquier combinación de los mismos. Si se implementan en software, las funciones se pueden almacenar en, o transmitir sobre, un medio legible por ordenador como una o más instrucciones o código. Los medios legibles por ordenador pueden incluir medios de almacenamiento de datos informáticos o medios de comunicación, incluyendo cualquier medio que facilite la transferencia de un programa informático desde un lugar a otro. Los medios de almacenamiento de datos pueden ser cualquier medio disponible al que se pueda acceder por uno o más ordenadores o uno o más procesadores para recuperar instrucciones, código y/o estructuras de datos para la implementación de las técnicas descritas en la presente divulgación. A modo de ejemplo y no de limitación, dichos medios legibles por

ordenador pueden comprender RAM, ROM, EEPROM, CD-ROM u otros dispositivos de almacenamiento en disco óptico, almacenamiento en disco magnético u otro almacenamiento magnético, o cualquier otro medio que se pueda usar para transportar o almacenar código de programa deseado en forma de instrucciones o estructuras de datos y al que se pueda acceder por un ordenador. Además, cualquier conexión recibe apropiadamente la denominación de medio legible por ordenador. Por ejemplo, si el software se transmite desde una página web, servidor u otra fuente remota usando un cable coaxial, cable de fibra óptica, par trenzado, línea digital de abonado (DSL) o tecnologías inalámbricas tales como infrarrojos, radio y microondas, entonces el cable coaxial, cable de fibra óptica, par trenzado, DSL o tecnologías inalámbricas, tales como infrarrojos, radio y microondas, se incluyen en la definición de medio. Los discos, como se usan en el presente documento, incluyen disco compacto (CD), disco láser, disco óptico, disco versátil digital (DVD), disco flexible y disco Blu-ray, donde algunos discos reproducen normalmente datos de forma magnética, mientras que otros discos reproducen los datos de forma óptica con láseres. Las combinaciones de lo anterior también se deben incluir dentro del alcance de los medios legibles por ordenador.

[0067] El código se puede ejecutar por uno o más procesadores, tales como uno o más procesadores de señales digitales (DSP), microprocesadores de propósito general, circuitos integrados específicos de la aplicación (ASIC), matrices lógicas programables *in situ* (FPGA) u otros circuitos lógicos, integrados o discretos equivalentes. En consecuencia, el término "procesador" o "unidad de procesamiento", como se usa en el presente documento, se puede referir a cualquiera de las estructuras anteriores o a cualquier otra estructura adecuada para la implementación de las técnicas descritas en el presente documento. Además, en algunos aspectos, la funcionalidad descrita en el presente documento se puede proporcionar dentro de módulos de hardware y/o software dedicados configurados para codificar y descodificar, o incorporar en un códec combinado. Además, las técnicas se podrían implementar por completo en uno o más circuitos o elementos lógicos.

[0068] Las técnicas de la presente divulgación se pueden implementar en una gran diversidad de dispositivos o aparatos, incluyendo un teléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (es decir, un conjunto de chips). En la presente divulgación se describen diversos componentes, módulos o unidades para destacar los aspectos funcionales de los dispositivos configurados para realizar las técnicas divulgadas, pero no requieren necesariamente su realización por diferentes unidades de hardware. En su lugar, como se describe anteriormente, diversas unidades se pueden combinar en una unidad de hardware de códec o proporcionarse por un grupo de unidades de hardware interoperativas, que incluya uno o más procesadores como se describe anteriormente, junto con software y/o firmware adecuados.

[0069] Se han descrito diversos ejemplos. Estos y otros ejemplos están dentro del alcance de las siguientes reivindicaciones.

REIVINDICACIONES

1. Un procedimiento de procesamiento gráfico que comprende:

asignar, por una unidad de procesamiento gráfico (GPU), una memoria intermedia cíclica como un canal de datos compartido en la memoria gráfica en chip de la GPU, en el que la memoria intermedia cíclica se comparte por al menos dos etapas de una tubería de procesamiento gráfico;

ejecutar, en unidades de sombreador en la GPU, las al menos dos etapas de la tubería de procesamiento gráfico;

almacenar, por la GPU en la memoria intermedia cíclica en la memoria gráfica en chip, datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador como colas de los datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico; y

leer, por la GPU desde la memoria intermedia cíclica en la memoria gráfica en chip, los datos producidos por una primera etapa de las al menos dos etapas de la tubería de procesamiento gráfico, incluyendo eliminar de la memoria intermedia cíclica los datos producidos por la primera etapa de las al menos dos etapas de la tubería de procesamiento gráfico que se lee desde la memoria intermedia cíclica, incrementando de este modo el espacio en la memoria intermedia cíclica para que la GPU almacene datos adicionales producidos por una segunda etapa de las al menos dos etapas de la tubería de procesamiento gráfico.

2. El procedimiento de la reivindicación 1, que comprende además:

asignar, por la GPU, un segundo canal de datos compartido en la memoria gráfica en chip de la GPU que se comparte por unas segundas al menos dos etapas de la tubería de procesamiento gráfico, en el que el canal de datos compartido es un primer canal de datos compartido;

ejecutar, en las unidades de sombreador en la GPU, las segundas al menos dos etapas de la tubería de procesamiento gráfico; y

almacenar, por la GPU en el segundo canal de datos compartido, los segundos datos producidos por cada una de las segundas al menos dos etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador.

3. El procedimiento de la reivindicación 2, que comprende además:

programar, por la GPU, la ejecución de una o más etapas de la tubería de procesamiento gráfico en las unidades de sombreador en base, al menos en parte, a un estado del primer canal de datos compartido o el segundo canal de datos compartido de modo que estén disponibles datos en el primer canal de datos compartido o el segundo canal de datos compartido para consumirse por las una o más etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador y que esté disponible espacio libre en el primer canal de datos compartido o el segundo canal de datos compartido para almacenar datos producidos por las una o más etapas de la tubería de procesamiento gráfico que se ejecutan en las unidades de sombreador.

4. El procedimiento de la reivindicación 2, en el que el segundo canal de datos compartido funciona en modo caché para almacenar en caché datos almacenados en el segundo canal de datos compartido, y el primer canal de datos compartido funciona en modo primero en entrar, primero en salir (FIFO).

5. El procedimiento de la reivindicación 2, en el que el primer canal de datos compartido es una primera memoria intermedia cíclica, y el segundo canal de datos compartido es una segunda memoria intermedia cíclica.

6. El procedimiento de la reivindicación 2, en el que los datos almacenados en el primer canal de datos compartido comprenden datos de vértices producidos por las al menos dos etapas de la tubería de procesamiento gráfico, y los segundos datos almacenados en el segundo canal de datos compartido comprenden primitivas producidas por las segundas al menos dos etapas de la tubería de procesamiento gráfico.

7. El procedimiento de la reivindicación 6, en el que las al menos dos etapas de la tubería de procesamiento gráfico comprenden un sombreador de vértices y un sombreador de dominios.

8. El procedimiento de la reivindicación 6, en el que las segundas al menos dos etapas de la tubería de procesamiento gráfico comprenden un sombreador de casco y un sombreador de geometría.

9. El procedimiento de la reivindicación 2, que comprende además:

reservar, por la GPU, espacio libre en al menos uno del primer canal de datos compartido y el segundo canal de datos compartido para evitar un punto muerto entre el primer canal de datos compartido y el segundo canal de datos compartido.

10. Un aparato que comprende:

medios para asignar una memoria intermedia cíclica como un canal de datos compartido en la memoria gráfica en chip de una unidad de procesamiento gráfico (GPU), en el que la memoria intermedia cíclica se comparte por al menos dos etapas de una tubería de procesamiento gráfico;

medios para ejecutar las al menos dos etapas de la tubería de procesamiento gráfico;

medios para almacenar, en la memoria intermedia cíclica en la memoria gráfica en chip, los datos producidos por la ejecución de cada una de las al menos dos etapas de la tubería de procesamiento gráfico como colas de los datos producidos por cada una de las al menos dos etapas de la tubería de procesamiento gráfico; y

medios para leer desde la memoria intermedia cíclica en la memoria gráfica en chip, los datos producidos por una primera etapa de las al menos dos etapas de la tubería de procesamiento gráfico, incluyendo eliminar de la memoria intermedia cíclica los datos producidos por la primera etapa de las al menos dos etapas de la tubería de procesamiento gráfico que se leen desde la memoria intermedia cíclica, incrementando de este modo el espacio en la memoria intermedia cíclica para que la GPU almacene datos adicionales producidos por una segunda etapa de las al menos dos etapas de la tubería de procesamiento gráfico.

11. El aparato de la reivindicación 10, que comprende además:

medios para asignar un segundo canal de datos compartido en la memoria gráfica en chip de la GPU que se comparte por unas segundas al menos dos etapas de la tubería de procesamiento gráfico, en el que el canal de datos compartido es un primer canal de datos compartido;

medios para ejecutar las segundas al menos dos etapas de la tubería de procesamiento gráfico; y medios para almacenar, en el segundo canal de datos compartido, segundos datos producidos por la ejecución de cada una de las segundas al menos dos etapas de la tubería de procesamiento gráfico.

12. El aparato de la reivindicación 11, que comprende además:

medios que programan la ejecución de una o más etapas de la tubería de procesamiento gráfico en base, al menos en parte, a un estado del primer canal de datos compartido o el segundo canal de datos compartido de modo que estén disponibles datos en el primer canal de datos compartido o el segundo canal de datos compartido para consumirse por las una o más etapas de la tubería de procesamiento gráfico y que esté disponible espacio libre en el primer canal de datos compartido o el segundo canal de datos compartido para almacenar datos producidos por las una o más etapas de la tubería de procesamiento gráfico.

13. El aparato de la reivindicación 11, en el que, el segundo canal de datos compartido funciona en modo caché para almacenar en caché datos almacenados en el segundo canal de datos compartido, y el primer canal de datos compartido funciona en modo primero en entrar, primero en salir (FIFO).

14. El aparato de la reivindicación 11, en el que el primer canal de datos compartido es una primera memoria intermedia cíclica, y el segundo canal de datos compartido es una segunda memoria intermedia cíclica.

15. Un medio de almacenamiento legible por ordenador que almacena instrucciones que, cuando se ejecutan, hacen que uno o más procesadores programables realicen el procedimiento de las reivindicaciones 1 a 9.

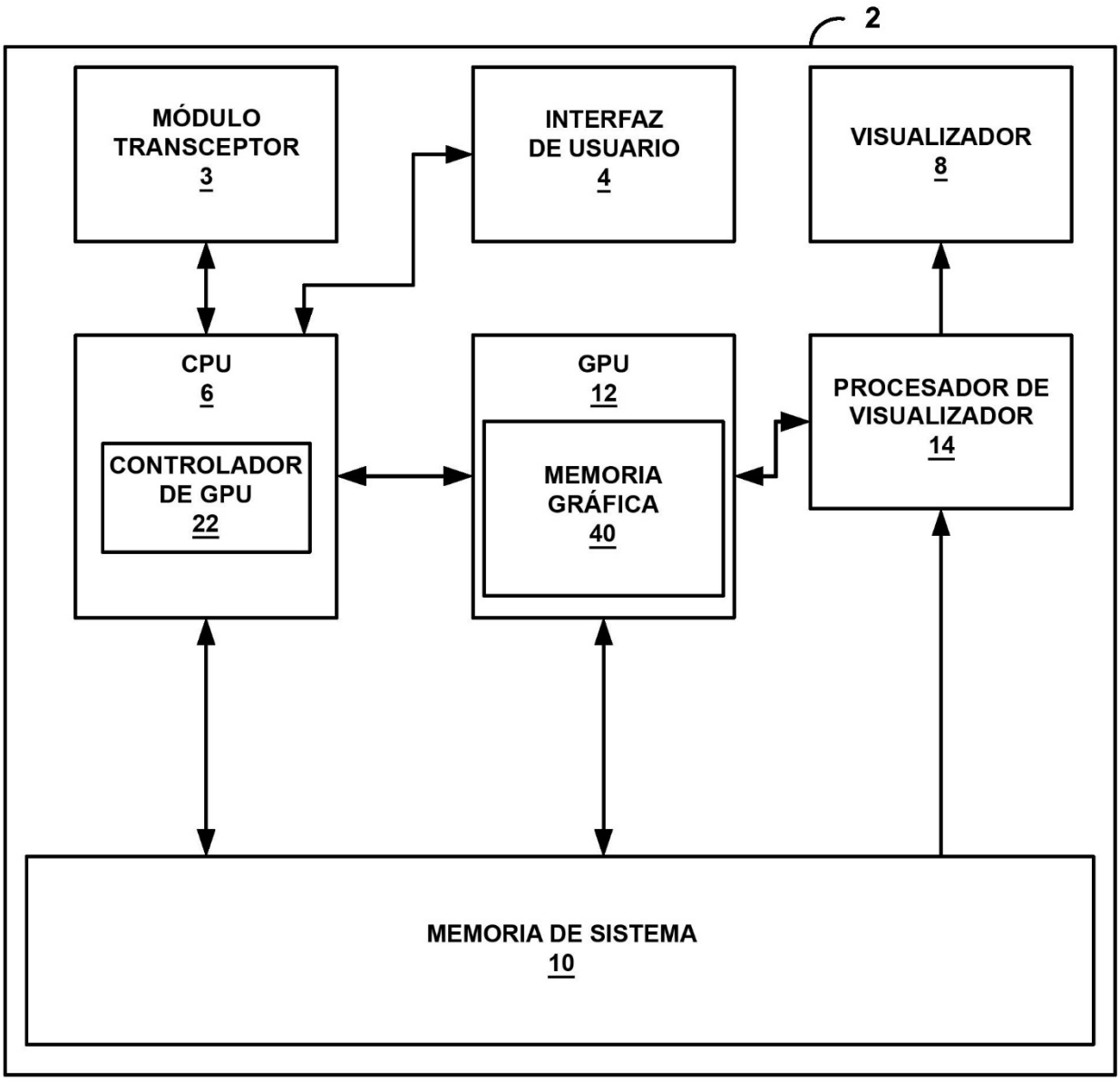
2
MÓDULO TRANSCEPTOR 3
INTERFAZ DE USUARIO 4
VISUALIZADOR 8
CPU 6
CONTROLADOR DE GPU 22
GPU 12
MEMORIA GRÁFICA 40
PROCESADOR DE VISUALIZADOR 14
MEMORIA DE SISTEMA 10

FIG. 1


TUBERÍA DE PROCESAMIENTO GRÁFICO 24
ENSAMBLADOR DE ENTRADA 26
ETAPA DE SOMBREADOR DE VÉRTICES 28
ETAPA DE SOMBREADOR DE CASCO 30
ETAPA DE TESELADOR 32
SOMBREADOR DE DOMINIOS 34
SOMBREADOR DE GEOMETRÍA 36
ETAPA DE RASTERIZADOR 37
SOMBREADOR DE PÍXELES 38

FIG. 2

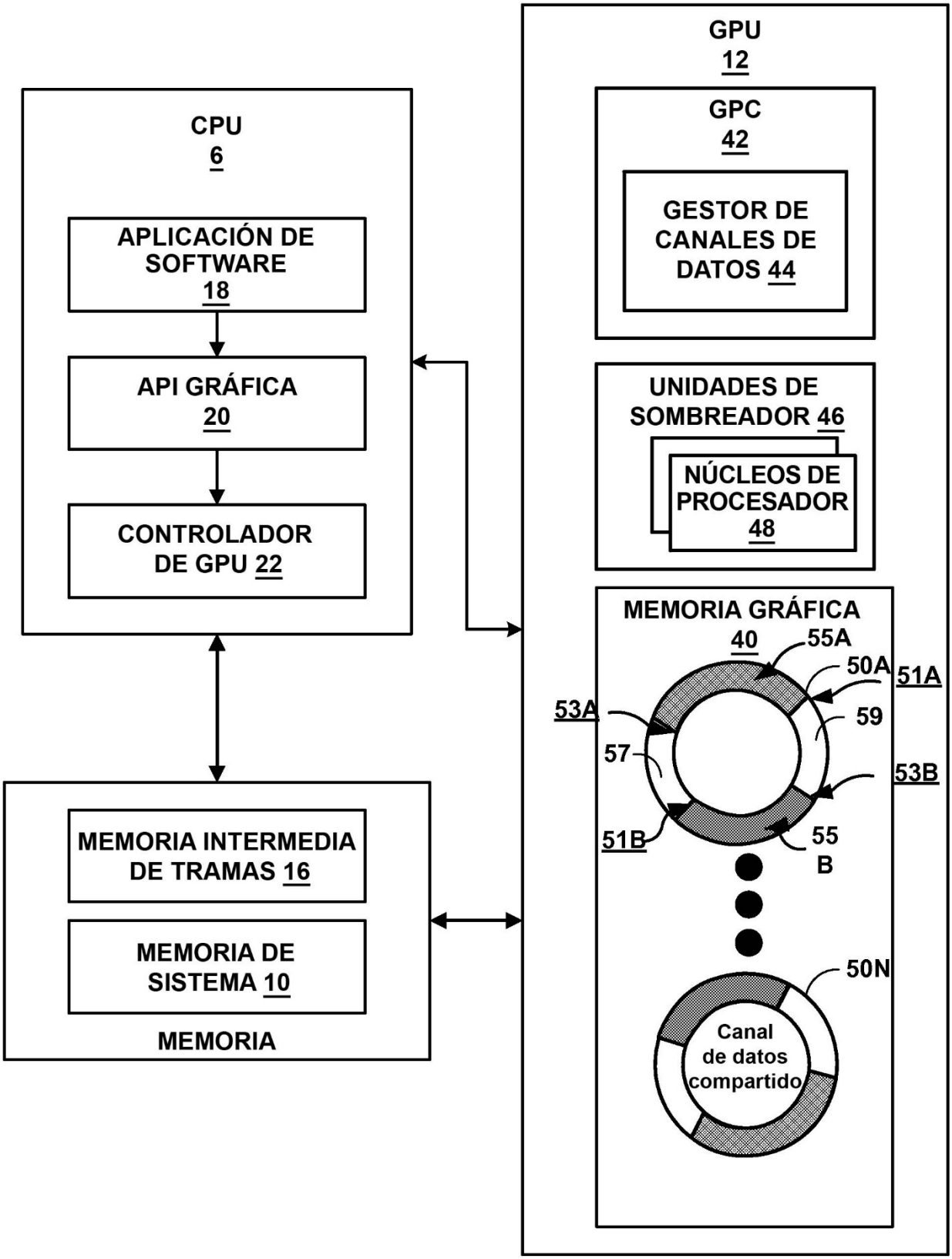
CPU
6
APLICACIÓN DE SOFTWARE 18
API GRÁFICA 20
CONTROLADOR DE GPU 22
MEMORIA INTERMEDIA DE TRAMAS 16
MEMORIA DE SISTEMA 10
MEMORIA
GPU
12
GPC
42
GESTOR DE CANALES DE DATOS 44
UNIDADES DE SOMBREADOR 46
NÚCLEOS DE PROCESADOR 48
MEMORIA GRÁFICA
40
55A
50A
51A
53A
59
57
53B
51B
55 B
50N
Canal de datos compartido

FIG. 3

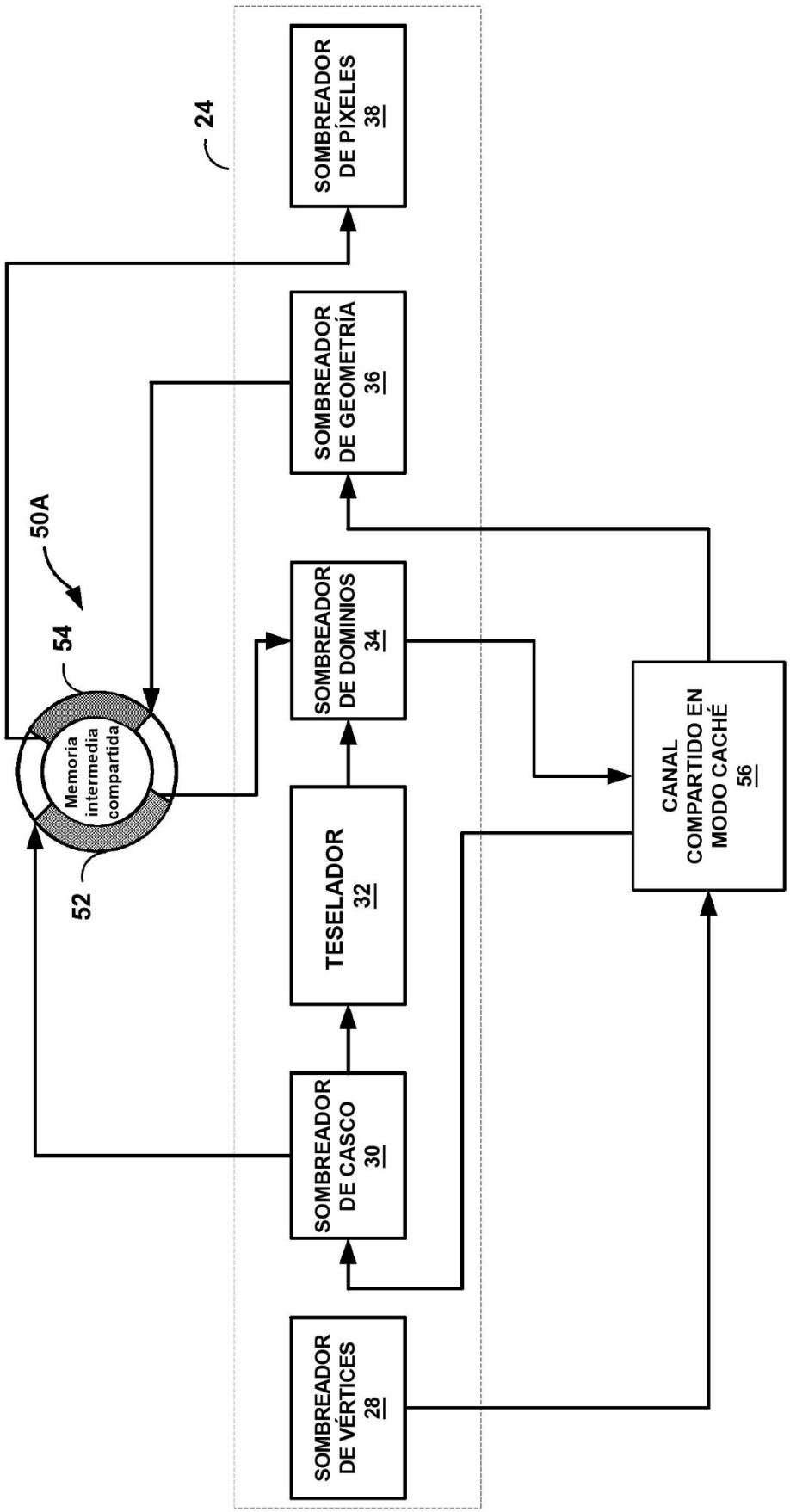
24
SOMBREADOR DE PÍXELES 38
SOMBREADOR DE GEOMETRÍA 36
50A
54
Memoria intermedia compartida
52
SOMBREADOR DE DOMINIOS 34
TESELADOR 32
SOMBREADOR DE CASCO 30
SOMBREADOR DE VÉRTICES 28
CANAL COMPARTIDO EN MODO CACHÉ 56

FIG. 4

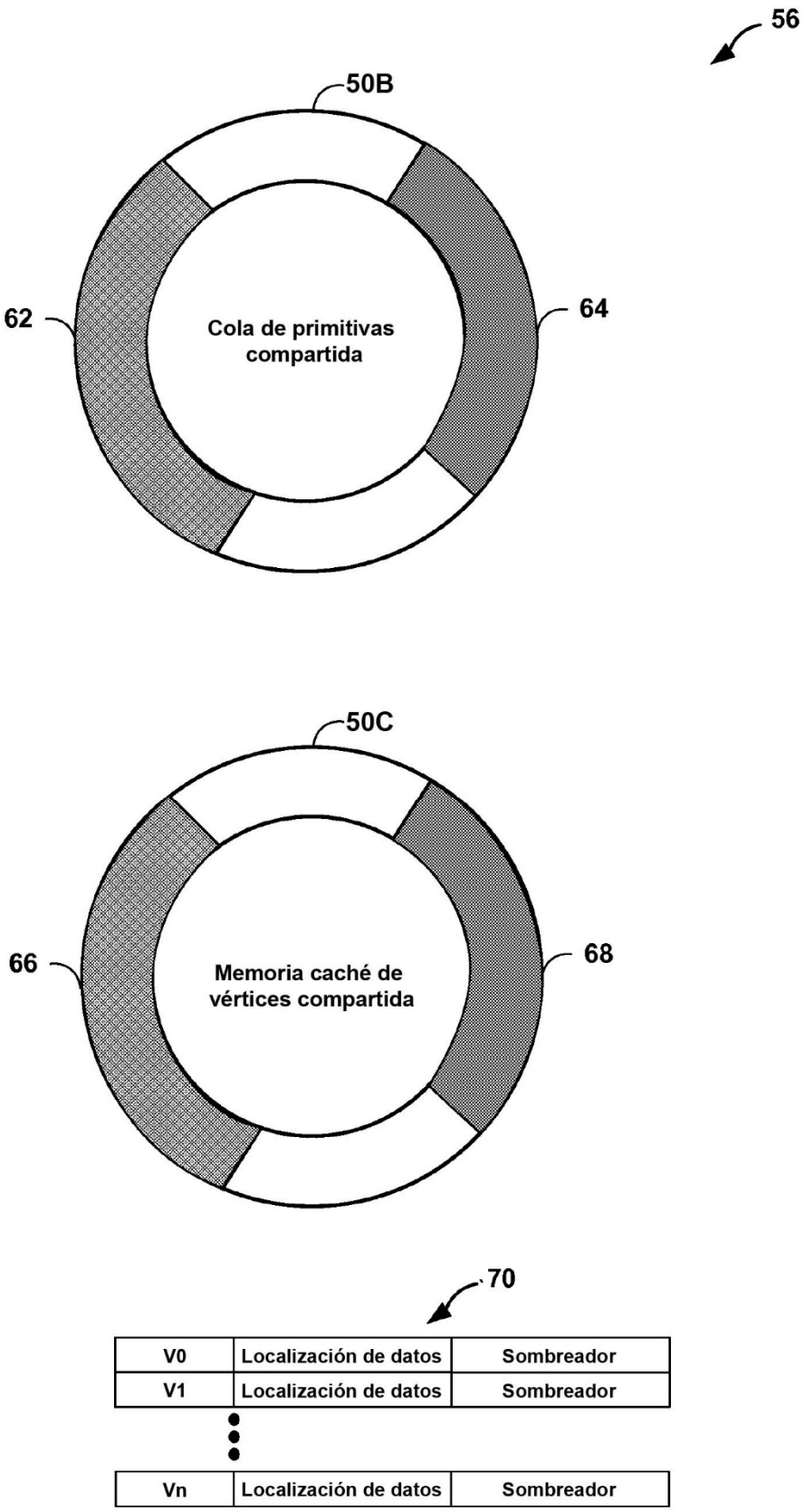
56
50B
62
Cola de primitivas compartida
64
50C
66
Memoria caché de vértices compartida
68
70
V0
Localización de datos
Sombreador
V1
Localización de datos
Sombreador
Vn
Localización de datos
Sombreador

FIG. 5

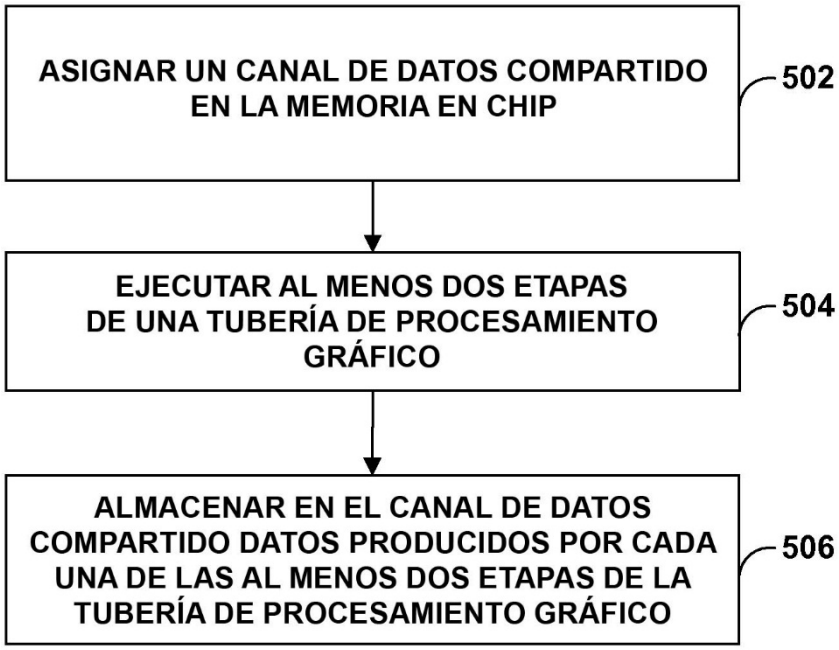
ASIGNAR UN CANAL DE DATOS COMPARTIDO EN LA MEMORIA EN CHIP
502
EJECUTAR AL MENOS DOS ETAPAS DE UNA TUBERÍA DE PROCESAMIENTO GRÁFICO
504
ALMACENAR EN EL CANAL DE DATOS COMPARTIDO DATOS PRODUCIDOS POR CADA UNA DE LAS AL MENOS DOS ETAPAS DE LA TUBERÍA DE PROCESAMIENTO GRÁFICO
506

FIG. 6