(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년04월19일
(11) 등록번호 10-1728624
(24) 등록일자 2017년04월13일

(51) 국제특허분류(Int. Cl.)
G06T 15/00 (2006.01) *G06T 11/40* (2006.01)
(52) CPC특허분류
G06T 15/005 (2013.01)
G06T 11/40 (2013.01)
(21) 출원번호 10-2015-7031663
(22) 출원일자(국제) 2014년03월17일
심사청구일자 2016년10월28일
(85) 번역문제출일자 2015년11월04일
(65) 공개번호 10-2015-0143567
(43) 공개일자 2015년12월23일
(86) 국제출원번호 PCT/US2014/030447
(87) 국제공개번호 WO 2014/168740
국제공개일자 2014년10월16일
(30) 우선권주장
61/811,056 2013년04월11일 미국(US)
14/044,396 2013년10월02일 미국(US)
(56) 선행기술조사문헌
WO2013039812 A1
US20100020090 A1
KR1020110042872 A
KR1020120139284 A

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
프라스카티 크리스토퍼 폴
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
간가니 히텐드라 모한
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
세트하라마이아 아비나쉬
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(74) 대리인
특허법인코리아나

전체 청구항 수 : 총 42 항 심사관 : 조우연

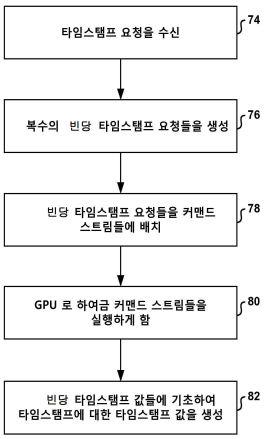
(54) 발명의 명칭 **타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들**

(57) 요 약

본 개시물은 타일 기반 렌더링을 수행하는 그래픽스 시스템에서 인트라-프레임 타임스탬프들을 지원하기 위한 기법들을 설명한다. 인트라-프레임 타임스탬프들을 지원하기 위한 기법들은 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 그래픽스 프로세싱 유닛 (GPU) 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하

(뒷면에 계속)

대 표 도 - 도8



여 시점을 나타내는 타임스탬프 값을 생성하는 것을 수반할 수도 있다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다. 타임스탬프 값은 중앙 프로세싱 유닛 (CPU), GPU, 다른 프로세서, 또는 이들이 임의의 조합에 의해 생성될 수도 있다. 인트라-프레임 타임스탬프 요청들에 대한 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용함으로써, 인트라-프레임 타임스탬프들은 타일 기반 렌더링을 수행하는 그래픽스 시스템에 의해 지원될 수도 있다.

명 세 서

청구범위

청구항 1

타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법으로서,

하나 이상의 프로세서들에 의해, 중앙 프로세싱 유닛 (CPU) 상에서 실행되는 그래픽스 애플리케이션으로부터의 타임스탬프 요청을 프로세싱하는 단계;

상기 하나 이상의 프로세서들 중 적어도 하나의 프로세서로, 상기 타임스탬프 요청에 응답하여 복수의 빈당 (per-bin) 타임스탬프 요청들을 생성하는 단계;

그래픽스 프로세싱 유닛 (GPU) 으로, 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 동안 상기 복수의 빈당 타임스탬프 요청들에 응답하여 복수의 빈당 타임스탬프 값들을 생성하는 단계; 및

상기 하나 이상의 프로세서들 중 적어도 하나의 프로세서로, 상기 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하는 단계로서,

상기 타임스탬프 값을 생성하는 단계는 상기 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수를 실행하는 단계를 포함하고,

상기 함수를 실행하는 단계는 초기 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 초기 참조 타임스탬프를, 상기 그래픽스 프레임을 렌더링하는데 이용된 2 개 이상의 렌더링 패스 반복들에 대한 각각의 참조 타임스탬프 값들과 각각의 빈당 타임스탬프 값들 사이의 차이들의 합계에 추가하는 단계를 포함하는,

상기 타임스탬프 값을 생성하는 단계를 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 2

제 1 항에 있어서,

상기 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 3

제 1 항에 있어서,

상기 복수의 빈당 타임스탬프 값들을 생성하는 단계는 :

상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 제 1 빈당 타임스탬프 값을 생성하는 단계, 및

상기 복수의 렌더링 패스 반복들 중 상기 제 1 렌더링 패스 반복과는 상이한 제 2 렌더링 패스 반복 동안 제 2 빈당 타임스탬프 값을 생성하는 단계로서, 상기 적어도 2 개의 빈당 타임스탬프 값들은 상기 제 1 빈당 타임스탬프 값 및 상기 제 2 빈당 타임스탬프 값을 포함하는, 상기 제 2 빈당 타임스탬프 값을 생성하는 단계를 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 4

제 3 항에 있어서,

상기 GPU 로, 상기 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 및 상기 렌더링 패스 반복들 중 상기 각각의 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 제 1 빈당 타임스탬프 값 또는 상기 제 2 빈당 타임스탬프 값과 각각 연관된 제 1 참조 타임스탬프 값 또는 제 2 참조 타임스탬프 값을 생성하

는 단계를 더 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 5

제 1 항에 있어서,

상기 함수를 실행하는 단계는, 다음의 수식 :

$$Value = TSVGPU(0) + \sum_{y=0}^{N-1}\left(TSV(y) - TSVGPU(y)\right)$$

을 푸는 단계를 포함하며,

Value 는 상기 타임스탬프 값이고, TSV(y) 는 제 y 렌더링 패스 반복 동안 생성되는 각각의 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 각각의 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 총 수인, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 6

제 1 항에 있어서,

상기 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들에서의 적어도 2 개의 드로우 콜 커맨드들 사이에 포지셔닝되는 상기 타임스탬프 요청에 응답하여 상기 타임스탬프 값을 생성하는 단계를 더 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 7

제 6 항에 있어서,

상기 GPU 에 의해, 상기 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 복수의 렌더링 패스 반복들을 수행하는 단계를 더 포함하며,

상기 적어도 2 개의 빈당 타임스탬프 값들은 제 1 빈당 타임스탬프 값 및 제 2 빈당 타임스탬프 값을 포함하고,

상기 제 1 빈당 타임스탬프 값은 상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 수행되는 적어도 2 개의 빈당 드로우 콜들의 수행 사이에 일어나는 시점을 나타내며, 상기 제 1 렌더링 패스 반복 동안 수행되는 상기 적어도 2 개의 빈당 드로우 콜들 각각은 상기 적어도 2 개의 드로우 콜 커맨드들 중 각각의 드로우 콜 커맨드와 연관되며,

상기 제 2 빈당 타임스탬프 값은 상기 복수의 렌더링 패스 반복들 중 제 2 렌더링 패스 반복 동안 수행되는 적어도 2 개의 빈당 드로우 콜들의 수행 사이에 일어나는 시점을 나타내며, 상기 제 2 렌더링 패스 반복 동안 수행되는 상기 적어도 2 개의 빈당 드로우 콜들 각각은 상기 적어도 2 개의 드로우 콜 커맨드들 중 각각의 드로우 콜 커맨드와 연관되며, 상기 제 2 렌더링 패스 반복은 상기 제 1 렌더링 패스 반복과는 상이한, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 8

제 1 항에 있어서,

복수의 빈당 타임스탬프 요청들 중 각각의 빈당 타임스탬프 요청에 응답하여 상기 복수의 빈당 타임스탬프 값들 각각을 생성하는 단계로서, 상기 빈당 타임스탬프 요청들 각각은 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치되는, 상기 복수의 빈당 타임스탬프 값들 각각을 생성하는 단계; 및

상기 GPU 에 의해, 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 상기 커맨드 스트림들 각각을 실행하는 단계로서, 각각의 렌더링 패스 반복은 렌더 타겟의 복수의 서브-영역들 중 하나의 서브-영역을 렌더링하는, 상기 커맨드 스트림들 각각을 실행하는 단계를 더 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 9

제 1 항에 있어서,

상기 타임스탬프 값을 생성하는 단계는 :

상기 복수의 빈당 타임스탬프 값들에 기초하여 복수의 타임스탬프 값들을 생성하는 단계를 포함하며,

상기 타임스탬프 값들 각각은 상기 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들에 포함된 복수의 타임스탬프 요청들 중 각각의 타임스탬프 요청에 대응하며, 상기 타임스탬프 요청들 각각은 상기 그래픽스 애플리케이션에 의해 요청되는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 10

제 9 항에 있어서,

상기 타임스탬프 요청들 중 적어도 2 개의 타임스탬프 요청들은 상기 그래픽스 프레임에 대해 실행될 상기 순서화된 시퀀스의 커맨드들에서의 연이은 드로우 콜 커맨드들의 각각의 쌍들 사이에 포지셔닝되며,

상기 복수의 타임스탬프 값들을 생성하는 단계는 상기 순서화된 시퀀스의 커맨드들에서의 상기 타임스탬프 요청들에 대해 반환된 상기 타임스탬프 값들이 상기 순서화된 시퀀스의 커맨드들의 시작에서 상기 순서화된 시퀀스의 커맨드들의 끝까지 값이 단조 증가하도록 상기 빈당 타임스탬프 값들에 기초하여 상기 복수의 타임스탬프 값들을 생성하는 단계를 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 11

제 9 항에 있어서,

상기 복수의 타임스탬프 값들은 상기 그래픽스 프레임의 상기 렌더링 동안 상기 순서화된 시퀀스의 커맨드들에서의 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 나타내는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 12

제 9 항에 있어서,

상기 복수의 타임스탬프 값들을 생성하는 단계는, 다음의 수식 :

$$Value(x) = TSVGPU(0) + \sum_{y=0}^{N-1}\left(TSV(x,y) - TSVGPU(y)\right)$$

에 기초하여 상기 복수의 타임스탬프 값들을 생성하는 단계를 포함하며,

Value(x) 는 상기 순서화된 시퀀스의 커맨드들에서의 제 x 타임스탬프 요청에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSV(x,y) 는 제 y 렌더링 패스 반복 동안 생성되고 상기 순서화된 시퀀스의 커맨드들에서의 상기 제 x 타임스탬프 요청에 대응하는 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수인, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 13

제 1 항에 있어서,

상기 타임스탬프 값을 생성하는 단계는 :

상기 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치하는 단계;

상기 GPU 에 의해, 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 상기 커맨드 스트림들 각각을 실행하는 단계로서, 각각의 렌더링 패스 반복은 렌더 타겟의 복수의 서브-영역들 중 하나의 서브-영역을 렌더링하는, 상기 커맨드 스트림들 각각을 실행하는 단계; 및

상기 커맨드 스트림들에 배치된 상기 빈당 타임스탬프 요청들에 응답하여 상기 GPU 에 의해 생성된 상기 빈당

타임스탬프 값들에 기초하여 상기 타임스탬프 값을 생성하는 단계를 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 14

제 1 항에 있어서,

상기 하나 이상의 프로세서들은 상기 중앙 프로세싱 유닛 (CPU) 을 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 15

제 1 항에 있어서,

상기 하나 이상의 프로세서들은 상기 GPU 를 포함하는, 타일 기반 렌더링을 위한 인트라-프레임 타임스탬프들을 지원하기 위한 방법.

청구항 16

타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스로서,

중앙 프로세싱 유닛 (CPU) 상에서 실행되는 그래픽스 애플리케이션으로부터의 타임스탬프 요청을 프로세싱하고 상기 타임스탬프 요청에 응답하여 복수의 빈당 (per-bin) 타임스탬프 요청들을 생성하도록 구성된 하나 이상의 프로세서들;

상기 하나 이상의 프로세서들 중 적어도 하나에 커플링된 그래픽스 프로세싱 유닛 (GPU) 으로서, 상기 GPU 는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 동안 상기 복수의 빈당 타임스탬프 요청들에 응답하여 복수의 빈당 타임스탬프 값들을 생성하도록 구성되는, 상기 GPU 를 포함하며,

상기 하나 이상의 프로세서들 중 적어도 하나의 프로세서는 또한, 상기 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하는 것으로서,

상기 타임스탬프 값을 생성하기 위해, 상기 하나 이상의 프로세서들은 상기 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수를 실행하도록 구성되고,

상기 함수를 실행하기 위해, 상기 하나 이상의 프로세서들은 초기 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 초기 참조 타임스탬프를, 상기 그래픽스 프레임을 렌더링하는데 이용된 2 개 이상의 렌더링 패스 반복들에 대한 각각의 참조 타임스탬프 값들과 각각의 빈당 타임스탬프 값들 사이의 차이들의 합계에 추가하도록 구성되는,

상기 타임스탬프 값을 생성하도록 구성되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 17

제 16 항에 있어서,

상기 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 18

제 16 항에 있어서,

상기 타임스탬프 값들을 생성하기 위해, 상기 하나 이상의 프로세서들은 :

상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 제 1 빈당 타임스탬프 값을 생성하며,

상기 복수의 렌더링 패스 반복들 중 상기 제 1 렌더링 패스 반복과는 상이한 제 2 렌더링 패스 반복 동안 제 2 빈당 타임스탬프 값을 생성하는 것으로서, 상기 적어도 2 개의 빈당 타임스탬프 값들은 상기 제 1 빈당 타임스탬프 값 및 상기 제 2 빈당 타임스탬프 값을 포함하는, 상기 제 2 빈당 타임스탬프 값을 생성하도록 구성되는,

타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 19

제 18 항에 있어서,

상기 GPU 는 또한, 상기 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 및 상기 렌더링 패스 반복들 중 상기 각각의 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 제 1 빈당 타임스탬프 값 또는 상기 제 2 빈당 타임스탬프 값과 각각 연관된 제 1 참조 타임스탬프 값 또는 제 2 참조 타임스탬프 값을 생성하도록 구성되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 20

제 16 항에 있어서,

상기 함수를 실행하기 위해, 상기 하나 이상의 프로세서들은, 다음의 수식 :

$$Value = TSVGPU(0) + \sum_{y=0}^{N-1}\left(TSV(y) - TSVGPU(y)\right)$$

을 풀도록 구성되며,

Value 는 상기 타임스탬프 값이고, TSV(y) 는 제 y 렌더링 패스 반복 동안 생성되는 각각의 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 각각의 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 총 수인, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 21

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 또한, 상기 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들에서의 적어도 2 개의 드로우 콜 커맨드들 사이에 포지셔닝되는 상기 타임스탬프 요청에 응답하여 상기 타임스탬프 값을 생성하도록 구성되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 22

제 21 항에 있어서,

상기 GPU 는 상기 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 복수의 렌더링 패스 반복들을 수행하도록 구성되고,

상기 적어도 2 개의 빈당 타임스탬프 값들은 제 1 빈당 타임스탬프 값 및 제 2 빈당 타임스탬프 값을 포함하고,

상기 제 1 빈당 타임스탬프 값은 상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 수행되는 적어도 2 개의 빈당 드로우 콜들의 수행 사이에 일어나는 시점을 나타내며, 상기 제 1 렌더링 패스 반복 동안 수행되는 상기 적어도 2 개의 빈당 드로우 콜들 각각은 상기 적어도 2 개의 드로우 콜 커맨드들 중 각각의 드로우 콜 커맨드와 연관되며,

상기 제 2 빈당 타임스탬프 값은 상기 복수의 렌더링 패스 반복들 중 제 2 렌더링 패스 반복 동안 수행되는 적어도 2 개의 빈당 드로우 콜들의 수행 사이에 일어나는 시점을 나타내며, 상기 제 2 렌더링 패스 반복 동안 수행되는 상기 적어도 2 개의 빈당 드로우 콜들 각각은 상기 적어도 2 개의 드로우 콜 커맨드들 중 각각의 드로우 콜 커맨드와 연관되며, 상기 제 2 렌더링 패스 반복은 상기 제 1 렌더링 패스 반복과는 상이한, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 23

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 또한, 복수의 빈당 타임스탬프 요청들 중 각각의 빈당 타임스탬프 요청에 응답

하여 상기 복수의 빈당 타임스탬프 값들 각각을 생성하도록 구성되며, 상기 빈당 타임스탬프 요청들 각각은 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치되고, 상기 GPU 는 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 상기 커맨드 스트림들 각각을 실행하도록 구성되며, 각각의 렌더링 패스 반복은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 24

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 또한, 상기 복수의 빈당 타임스탬프 값들에 기초하여 복수의 타임스탬프 값들을 생성하도록 구성되며, 상기 타임스탬프 값들 각각은 상기 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들에 포함된 복수의 타임스탬프 요청들 중 각각의 타임스탬프 요청에 대응하며, 상기 타임스탬프 요청들 각각은 상기 그래픽스 애플리케이션에 의해 요청되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 25

제 24 항에 있어서,

상기 타임스탬프 요청들 중 적어도 2 개의 타임스탬프 요청들은 상기 그래픽스 프레임에 대해 실행될 상기 순서화된 시퀀스의 커맨드들에서의 연이은 드로우 콜 커맨드들의 각각의 쌍들 사이에 포지셔닝되며,

상기 하나 이상의 프로세서들은 또한, 상기 순서화된 시퀀스의 커맨드들에서의 상기 타임스탬프 요청들에 대해 반환된 상기 타임스탬프 값들이 상기 순서화된 시퀀스의 커맨드들의 시작에서 상기 순서화된 시퀀스의 커맨드들의 끝까지 값이 단조 증가하도록 상기 빈당 타임스탬프 값들에 기초하여 상기 복수의 타임스탬프 값들을 생성하도록 구성되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 26

제 24 항에 있어서,

상기 복수의 타임스탬프 값들은 상기 그래픽스 프레임의 상기 렌더링 동안 상기 순서화된 시퀀스의 커맨드들에서의 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 나타내는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 27

제 24 항에 있어서,

상기 하나 이상의 프로세서들은 또한, 다음의 수식 :

$$Value(x) = TSVGPU(0) + \sum_{y=0}^{N-1} \left(TSV(x,y) - TSVGPU(y) \right)$$

에 기초하여 상기 복수의 타임스탬프 값들을 생성하도록 구성되며,

Value(x) 는 상기 순서화된 시퀀스의 커맨드들에서의 제 x 타임스탬프 요청에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSV(x,y) 는 제 y 렌더링 패스 반복 동안 생성되고 상기 순서화된 시퀀스의 커맨드들에서의 상기 제 x 타임스탬프 요청에 대응하는 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수인, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 28

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 또한 :

상기 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치하는 것으로서, 상기 커맨드 스트림들 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 상기 GPU 에 의해 실행되도록 구성되며, 각각의 렌더링 패스 반복은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 상기 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치하고;

상기 GPU 로 하여금 상기 커맨드 스트림들을 실행하게 하며;

상기 커맨드 스트림들에 배치된 상기 빈당 타임스탬프 요청들에 응답하여 상기 GPU 에 의해 생성된 상기 빈당 타임스탬프 값들에 기초하여 상기 타임스탬프 값을 생성하도록 구성되는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 29

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 상기 CPU 의 적어도 부분을 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 30

제 16 항에 있어서,

상기 하나 이상의 프로세서들은 상기 GPU 를 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 31

제 16 항에 있어서,

상기 디바이스는 무선 통신 디바이스를 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 32

제 16 항에 있어서,

상기 디바이스는 모바일 폰 핸드셋을 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 디바이스.

청구항 33

타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 장치로서,

중앙 프로세싱 유닛 (CPU) 상에서 실행되는 그래픽스 애플리케이션으로부터의 타임스탬프 요청을 프로세싱하는 수단;

상기 타임스탬프 요청에 응답하여 복수의 빈당 (per-bin) 타임스탬프 요청들을 생성하는 수단;

그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 동안 상기 복수의 빈당 타임스탬프 요청들에 응답하여 복수의 빈당 타임스탬프 값들을 생성하는 수단; 및

상기 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하는 수단으로서,

상기 타임스탬프 값을 생성하는 수단은 상기 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수를 실행하는 수단을 포함하고,

상기 함수를 실행하는 수단은 초기 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 GPU 에 의해 생성되는 초기 참조 타임스탬프를, 상기 그래픽스 프레임을 렌더링하는데 이용된 2 개 이상의 렌더링 패스 반복들에 대한 각각의 참조 타임스탬프 값들과 각각의 빈당 타임스탬프 값들 사이의 차이들의 합계에 추가하는 수단을 포함하는,

상기 타임스탬프 값을 생성하는 수단을 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원

하기 위한 장치.

청구항 34

제 33 항에 있어서,

상기 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 장치.

청구항 35

제 33 항에 있어서,

상기 복수의 빈당 타임스탬프 값들을 생성하는 수단은 :

상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 제 1 빈당 타임스탬프 값을 생성하는 수단, 및

상기 복수의 렌더링 패스 반복들 중 상기 제 1 렌더링 패스 반복과는 상이한 제 2 렌더링 패스 반복 동안 제 2 빈당 타임스탬프 값을 생성하는 수단으로서, 상기 적어도 2 개의 빈당 타임스탬프 값들은 상기 제 1 빈당 타임스탬프 값 및 상기 제 2 빈당 타임스탬프 값을 포함하는, 상기 제 2 빈당 타임스탬프 값을 생성하는 수단을 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 장치.

청구항 36

제 33 항에 있어서,

상기 함수를 실행하는 수단은, 다음의 수식 :

$$Value = TSVGPU(0) + \sum_{y=0}^{N-1}\left(TSV(y) - TSVGPU(y)\right)$$

을 푸는 수단을 포함하며,

Value 는 상기 타임스탬프 값이고, TSV(y) 는 제 y 렌더링 패스 반복 동안 생성되는 각각의 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 각각의 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수인, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 장치.

청구항 37

제 33 항에 있어서,

상기 타임스탬프 값을 생성하는 수단은 상기 CPU 또는 상기 GPU 중 적어도 하나를 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 장치.

청구항 38

타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 명령들을 포함하는 비일시적 컴퓨터 판독가능 저장 매체로서,

상기 명령들은, 하나 이상의 프로세서들에 의한 실행 시에, 상기 하나 이상의 프로세서들로 하여금 :

중앙 프로세싱 유닛 (CPU) 상에서 실행되는 그래픽스 애플리케이션으로부터의 타임스탬프 요청을 프로세싱하게 하고;

상기 타임스탬프 요청에 응답하여 복수의 빈당 (per-bin) 타임스탬프 요청들을 생성하게 하고;

그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 동안 상기 복수의 빈당 타임스탬프 요청들에 응답하여 복수의 빈당 타임스탬프 값들을 생성하게 하며;

상기 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하게 하는 것으로서,

상기 하나 이상의 프로세서들로 하여금, 상기 타임스탬프 값을 생성하게하는 명령들은, 실행 시에, 상

기 하나 이상의 프로세서들로 하여금, 상기 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수를 실행하게 하는 명령들을 포함하고,

상기 하나 이상의 프로세서들로 하여금, 상기 함수를 실행하게 하는 명령들은, 실행 시에, 상기 하나 이상의 프로세서들로 하여금, 초기 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 GPU 에 의해 생성되는 초기 참조 타임스탬프를, 상기 그래픽스 프레임을 렌더링하는데 이용된 2 개 이상의 렌더링 패스 반복들에 대한 각각의 참조 타임스탬프 값들과 각각의 빈당 타임스탬프 값들 사이의 차이들의 합계에 추가하게 하는 명령들을 포함하는,

상기 타임스탬프 값을 생성하게 하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 비일시적 컴퓨터 판독가능 저장 매체.

청구항 39

제 38 항에 있어서,

상기 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 비일시적 컴퓨터 판독가능 저장 매체.

청구항 40

제 38 항에 있어서,

상기 하나 이상의 프로세서들로 하여금, 상기 복수의 빈당 타임스탬프 값들을 생성하게 하는 명령들은,

실행 시에, 상기 하나 이상의 프로세서들로 하여금, 상기 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 동안 제 1 빈당 타임스탬프 값을 생성하게 하는 명령들, 및

실행 시에, 상기 하나 이상의 프로세서들로 하여금, 상기 복수의 렌더링 패스 반복들 중 상기 제 1 렌더링 패스 반복과는 상이한 제 2 렌더링 패스 반복 동안 제 2 빈당 타임스탬프 값을 생성하게 하는 명령들로서, 상기 적어도 2 개의 빈당 타임스탬프 값들은 상기 제 1 빈당 타임스탬프 값 및 상기 제 2 빈당 타임스탬프 값을 포함하는, 상기 제 2 빈당 타임스탬프 값을 생성하게 하는 명령들을 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 비일시적 컴퓨터 판독가능 저장 매체.

청구항 41

제 38 항에 있어서,

상기 하나 이상의 프로세서들로 하여금, 상기 함수를 실행하게 하는 명령들은, 실행 시에, 상기 하나 이상의 프로세서들로 하여금, 다음의 수식 :

$$Value = TSVGPU(0) + \sum_{y=0}^{N-1}(TSV(y) - TSVGPU(y))$$

을 풀게 하는 명령들을 포함하며,

Value 는 상기 타임스탬프 값이고, TSV(y) 는 제 y 렌더링 패스 반복 동안 생성되는 각각의 빈당 타임스탬프 값이고, TSVGPU(y) 는 상기 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 상기 GPU 에 의해 생성되는 각각의 참조 타임스탬프 값이며, N 은 상기 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수인, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 비일시적 컴퓨터 판독가능 저장 매체.

청구항 42

제 38 항에 있어서,

상기 하나 이상의 프로세서들은 상기 CPU 및 상기 GPU 중 적어도 하나를 포함하는, 타일 기반 렌더링을 위한 인프라-프레임 타임스탬프들을 지원하기 위한 비일시적 컴퓨터 판독가능 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 출원은 그 전체 내용이 본 명세서에 참조에 의해 통합되는 2013년 4월 11일자로 출원된 미국 가출원 제 61/811,056호의 이익을 주장한다.

[0002] 본 개시물은 그래픽스 프로세싱 시스템들에 관한 것으로, 보다 특히, 그래픽스 프로세싱 시스템에서 타임스탬프들을 이용하는 것에 관한 것이다.

배 경 기 술

[0003] 컴퓨팅 디바이스들은 종종 그래픽스 프로세싱 유닛 (GPU) 을 활용하여 디스플레이를 위한 그래픽스 데이터의 렌더링을 가속화한다. 이러한 컴퓨팅 디바이스들은 예를 들어, 컴퓨터 워크스테이션들, 모바일 폰들 (예를 들어, 소위 스마트폰들), 임베디드 시스템들, 개인용 컴퓨터들, 태블릿 컴퓨터들, 및 비디오 게임 콘솔들을 포함할 수도 있다. 렌더링은 일반적으로 하나 이상의 3 차원 (3D) 그래픽스 오브젝트들을 포함할 수도 있는 3D 그래픽스 장면을 2 차원 (2D) 래스터화된 이미지 데이터로 변환하는 프로세스를 나타낸다. 그래픽스 장면은 하나 이상의 프레임들의 시퀀스로서 렌더링될 수도 있으며, 여기서 각각의 프레임은 시간에 있어서의 특정 인스턴스에서의 그래픽스 장면을 나타낸다.

[0004] GPU 는 3D 그래픽스 장면의 렌더링을 위한 적어도 부분 하드웨어 가속을 제공하기 위해 3D 렌더링 파이프라인을 포함할 수도 있다. 장면에서의 3D 그래픽스 오브젝트들은 그래픽스 애플리케이션에 의해 하나 이상의 3D 그래픽스 프리미티브들 (예를 들어, 포인트들, 라인들, 삼각형들, 패치 (patch) 들 등) 로 서브분할될 수도 있고, GPU 는 렌더링될 프레임들 각각에 대해 2D 래스터화된 이미지 데이터로 장면의 3D 그래픽스 프리미티브들을 변환할 수도 있다. 따라서, GPU 렌더링의 특정 맥락에서, 렌더링은 그래픽스 장면에서의 3D 오브젝트들에 대응하는 3D 그래픽스 프리미티브들을 2D 래스터화된 이미지 데이터로 변환하는 프로세스를 지칭할 수도 있다.

[0005] 특정 프레임에 대한 3D 그래픽스 프리미티브들을 렌더링하기 위해, 호스트 중앙 프로세싱 유닛 (CPU) 상에서 실행되는 그래픽스 애플리케이션은 렌더링될 프리미티브들에 대응하는 지오메트리 데이터를 GPU 액세스가능 메모리에 배치하고, 하나 이상의 GPU 상태 셋업 커맨드들을 커맨드 스트림에 배치하며, GPU 로 하여금 지오메트리 데이터에 기초하여 프리미티브들을 렌더링하게 하는 하나 이상의 드로우 콜 (draw call) 들을 커맨드 스트림에 배치할 수도 있다. GPU 는 커맨드들이 커맨드 스트림에 배치되었던 순서로 커맨드 스트림에 포함된 커맨드들을 프로세싱하여, 장면을 렌더링할 수도 있다.

발명의 내용

과제의 해결 수단

[0006] 본 개시물은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프 요청들을 지원하기 위한 기법들을 설명한다. 타일 기반 렌더링은 렌더 타겟 (예를 들어, 프레임) 을 복수의 빈들 (예를 들어, 서브-영역들 또는 타일들) 로 서브분할하는 것, 및 빈들 각각에 대해 별개의 렌더링 패스 반복 (rendering pass iteration) 을 수행하는 것을 수반할 수도 있다. 인트라-프레임 타임스탬프 요청은 렌더링될 그래픽스 프레임과 연관되는 그래픽스 커맨드 스트림에서 임의의 위치들에 배치될 수 있는 타임스탬프 요청을 지칭할 수도 있다. 타임스탬프 요청은 타임스탬프 요청이 타임스탬프 요청을 프로세싱하는 디바이스 (예를 들어, GPU 또는 CPU) 에 의해 프로세싱되는 시간에 있어서의 인스턴스를 나타내는 타임스탬프 값에 대한 요청을 지칭할 수도 있다. 본 개시물의 인트라-프레임 타임스탬프 생성 기법들은 타일 기반 렌더링을 수행하는 동안 그래픽스 프로세싱 유닛 (GPU) 에 의해 생성되는 하나 이상의 빈당 (per-bin) 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용하는 것은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에 의해 인트라-프레임 타임스탬프들이 지원되는 것을 허용할 수도 있다.

[0007] 하나의 예에서, 본 개시물은 하나 이상의 프로세서들로, 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하는 단계를 포함하는 방법을 설명한다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다.

[0008] 다른 예에서, 본 개시물은 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하도록 구성된 하나 이상의 프로

세서들을 포함하는 디바이스를 설명한다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다.

[0009] 다른 예에서, 본 개시물은 GPU 를 포함하는 장치를 설명한다. 장치는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하는 수단을 더 포함한다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다.

[0010] 다른 예에서, 본 개시물은 실행될 때, 하나 이상의 프로세서들로 하여금, 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 그래픽스 프로세싱 유닛 (GPU) 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하게 하는 명령들을 저장하는 컴퓨터 판독가능 저장 매체를 설명한다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다.

[0011] 본 개시물의 하나 이상의 예들의 상세들은 첨부한 도면들 및 이하의 설명에 기재된다. 본 개시물의 다른 특징들, 목적들, 및 이점들은 설명 및 도면들로부터, 및 청구항들로부터 명백해질 것이다.

도면의 간단한 설명

[0012] 도 1 은 본 개시물의 인트라-프레임 타임스탬프 생성 기법들을 구현하는데 이용될 수도 있는 예시적인 컴퓨팅 디바이스를 예시하는 블록 다이어그램이다.

도 2 는 도 1 에 도시된 컴퓨팅 디바이스의 CPU, GPU 및 메모리를 더욱 상세히 예시하는 블록 다이어그램이다.

도 3 은 복수의 서브-영역들 (예를 들어, 타일들) 로 서브분할되는 예시적인 렌더 타겟 및 서브분할된 렌더 타겟 상에 디스플레이된 예시적인 세트의 프리미티브들을 예시하는 개념적 다이어그램이다.

도 4 는 본 개시물에 따른 그래픽스 애플리케이션에 의해 이슈된 예시적인 커맨드 스트림을 예시하는 개념적 다이어그램이다.

도 5 는 본 개시물에 따른 렌더링 패스를 수행하기 위한 예시적인 실행 타임라인을 예시하는 개념적 다이어그램이다.

도 6 은 본 개시물에 따른 복수의 렌더링 패스 반복들을 수행하기 위한 예시적인 커맨드 스트림들을 예시하는 개념적 다이어그램이다.

도 7 은 본 개시물에 따른 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프들을 지원하기 위한 예시적인 기법을 예시하는 플로우 다이어그램이다.

도 8 은 본 개시물에 따른 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프들을 지원하기 위한 다른 예시적인 기법을 예시하는 플로우 다이어그램이다.

발명을 실시하기 위한 구체적인 내용

[0013] 본 개시물은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프 요청들을 지원하기 위한 기법들을 설명한다. 타일 기반 렌더링은 렌더 타겟 (예를 들어, 프레임) 을 복수의 빈들 (예를 들어, 서브-영역들 또는 타일들) 로 서브분할하는 것, 및 빈들 각각에 대해 별개의 렌더링 패스 반복을 수행하는 것을 수반할 수도 있다. 인트라-프레임 타임스탬프 요청은 렌더링될 그래픽스 프레임과 연관되는 그래픽스 커맨드 스트림에서 임의의 위치들에 배치될 수 있는 타임스탬프 요청을 지칭할 수도 있다. 타임스탬프 요청은 타임스탬프 요청이 타임스탬프 요청을 프로세싱하는 디바이스 (예를 들어, GPU 또는 CPU) 에 의해 프로세싱되는 시간에 있어서의 인스턴스를 나타내는 타임스탬프 값에 대한 요청을 지칭할 수도 있다. 본 개시물의 인트라-프레임 타임스탬프 생성 기법들은 타일 기반 렌더링을 수행하는 동안 그래픽스 프로세싱 유닛 (GPU) 에 의해 생성되는 하나 이상의 빈당 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용하는 것은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에 의해 인트라-프레임 타임스탬프들이 지원되는 것을 허용할 수도 있다.

[0014] 2 개의 상이한 드로우 콜 커맨드들 사이에 위치되는 인트라-프레임 타임스탬프 요청을 이슈하는 그래픽스 애플리케이션은 예를 들어, 요청에 응답하여 반환된 타임스탬프 값을 제 1 드로우 콜 커맨드의 실행과 제 2 드로우 콜 커맨드의 실행 사이인 시간에 대응시킬 것으로 예상할 수도 있다. 그러나, 타일 기반 렌더링 시스템들은

하나의 드로우 콜 커맨드의 실행이 동일한 그래픽스 프레임과 연관된 다른 드로우 콜 커맨드들의 실행과 인터리빙되도록 비연속적인 방식으로 그래픽스 프레임에 대해 드로우 콜 커맨드들을 실행할 수도 있다. 예를 들어, 타일 기반 렌더링 시스템은 렌더링될 그래픽스 프레임과 연관된 드로우 콜 커맨드들의 실행을 복수의 빈당 드로우 콜들로 서브분할하고, 빈당 드로우 콜들을 빈별로 그룹화하며, 빈당 드로우 콜들의 그룹들 각각을 별개의 렌더링 패스 반복의 부분으로서 실행할 수도 있다. 드로우 콜 커맨드들을 실행하는 이 비연속적인 인터리빙된 방식은 타일 기반 렌더링 시스템들이 인트라-프레임 타임스탬프들을 지원하기 어렵게 만든다.

[0015] 본 개시물에 설명된 기법들은 타일 기반 렌더링 시스템이 비연속적인 인터리빙된 방식으로 드로우 콜 커맨드들을 실행하는 경우들에도, 타일 기반 렌더링 시스템이 인트라-프레임 타임스탬프들을 지원하는 것을 허용할 수도 있다. 예를 들어, 본 개시물의 인트라-프레임 타임스탬프 생성 기법들은 타일 기반 렌더링을 수행하는 동안 GPU 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프 값을 생성할 수도 있다. 애플리케이션 요청 타임스탬프 값을 생성하는데 이용된 빈당 타임스탬프 값들 중 적어도 일부는 상이한 렌더링 패스 반복들의 부분으로서 생성될 수도 있다. 상이한 렌더링 패스 반복들 동안 생성되는 빈당 타임스탬프 값들을 이용하는 것은 그래픽스 프레임의 렌더링 동안 상이한 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 적어도 어느 정도까지는 반영하는 애플리케이션 요청 타임스탬프 값들을 그래픽스 프로세싱 시스템이 생성하는 것을 허용할 수도 있다. 이렇게 하여, 유용한 타임스탬프 값들이 타일 기반 렌더링 시스템들에 의해, 상이한 드로우 콜 커맨드들에 의해 걸리는 상대적 실행 시간량들에 관한 타이밍 통계들을 채용하는 그래픽스 애플리케이션들에 제공될 수도 있다.

[0016] 본 명세서에 사용한 바와 같이, 애플리케이션 요청 타임스탬프 값은 그래픽스 애플리케이션 (예를 들어, 소프트웨어 스택에서 드라이버 레벨보다 위인 애플리케이션) 에 의해 생성되는 타임스탬프 요청에 응답하여 생성되는 타임스탬프 값을 지칭할 수도 있다. 빈당 타임스탬프 값은 특정 빈 (예를 들어, 렌더 타겟의 서브-영역) 에 대해 렌더링 패스 반복을 수행하는 동안 GPU 에 의해 생성되는 타임스탬프 값을 지칭할 수도 있다. 그래픽스 애플리케이션에 의해 생성되는 타임스탬프 요청은 애플리케이션 생성 타임스탬프 요청으로 지칭될 수도 있다. 일부 경우들에서, 빈당 타임스탬프 요청은 GPU 드라이버에 의해 또는 GPU (사용자 애플리케이션 계층보다 아래인 소프트웨어/하드웨어 계층) 에 의해 생성되는 타임스탬프 요청을 지칭할 수도 있다.

[0017] 일부 예들에서, GPU 드라이버 또는 다른 애플리케이션은 커맨드 스트림에서 수신된 애플리케이션 생성 타임스탬프 요청들 각각에 대한 복수의 빈당 타임스탬프 요청들을 생성할 수도 있다. 빈당 타임스탬프 요청들은 GPU 에 의해 서비스될 수도 있으며, 이는 빈당 타임스탬프 요청들 각각을 수신하는 것에 응답하여 각각의 빈당 타임스탬프 값을 생성할 수도 있다. 빈당 타임스탬프 값은 GPU 가 타일 기반 렌더링을 수행할 때 실행되는 커맨드 스트림에서 빈당 타임스탬프 요청을 조우한 시간을 나타낼 수도 있다. 애플리케이션 생성 타임스탬프 요청들에 응답하여 생성되는 타임스탬프들 및 타임스탬프 값들은 각각 애플리케이션 요청 타임스탬프들 및 애플리케이션 요청 타임스탬프 값들로 지칭될 수도 있다. 유사하게, 빈당 타임스탬프 요청들에 응답하여 생성되는 타임스탬프들 및 타임스탬프 값들은 각각 빈당 타임스탬프들 및 빈당 타임스탬프 값들로 지칭될 수도 있다.

[0018] 그래픽스 애플리케이션 (예를 들어, GPU 로 하여금 하나 이상의 그래픽스 프레임들을 렌더링하게 하는 명령들을 포함하는 호스트 중앙 프로세싱 유닛 (CPU) 상에서 실행되는 애플리케이션들) 은 종종 특정 그래픽스 프레임을 렌더링하기 위하여 다수의 드로우 콜 커맨드들을 이슈할 수도 있다. 예를 들어, GPU들은 통상 실행될 각각의 드로우 콜 커맨드에 대한 단일 세트의 렌더 상태 세팅들로 단일 타입의 프리미티브 (예를 들어, 포인트, 라인, 삼각형, 패치 등) 를 렌더링하도록 구성된다. 이러한 예들에서, 하나보다 더 많은 타입의 프리미티브가 프레임을 렌더링하는데 필요하다면 또는 하나보다 더 많은 타입의 렌더 상태가 프레임을 렌더링하는데 필요하다면, 그래픽스 애플리케이션은 단일 그래픽스 프레임을 렌더링하기 위해 다수의 드로우 콜 커맨드들을 이슈할 필요가 있을 수도 있다.

[0019] 다수의 드로우 콜 커맨드들이 그래픽스 프레임을 렌더링하는데 이용될 때 개개의 드로우 콜 커맨드들 또는 드로우 콜 커맨드들의 서브세트들의 실행에 대한 타이밍 통계들을 획득하기 위해, 그래픽스 애플리케이션은 GPU 에 의해 실행될 커맨드 스트림에서의 드로우 콜 커맨드들 사이에 타임스탬프 요청들을 배치할 수도 있다. 개개의 그래픽스 프레임을 렌더링하는데 이용되는 드로우 콜 커맨드들 사이에 배치되는 타임스탬프 요청들은 본 명세서에 인트라-프레임 타임스탬프 요청들로 지칭될 수도 있고, 이러한 요청들에 응답하여 생성되는 대응하는 타임스탬프들은 인트라-프레임 타임스탬프들로 지칭될 수도 있다.

[0020] 그래픽스 애플리케이션은 커맨드 스트림에 배치되는 각각의 타임스탬프 요청에 응답하여 타임스탬프를 수신할 수도 있다. 타임스탬프는 GPU 가 타임스탬프 요청을 실행한 시간을 특정하는 타임스탬프 값을 포함할 수도

있다. 그래픽스 커맨드 스트림들은 통상 커맨드들이 커맨드 스트림에 배치되는 순서로 GPU 에 의해 실행되기 때문에, 그래픽스 애플리케이션은, 타임스탬프 요청이 커맨드 스트림에서의 2 개의 인접한 드로우 콜들 사이에 배치될 때, 반환된 타임스탬프 값이 제 1 드로우 콜 커맨드의 실행과 제 2 드로우 콜 커맨드의 실행 사이에 일어나는 시간에 대응할 것이라고 예상할 수도 있다.

[0021] 상기 언급된 예상을 충족하는 타임스탬프 값들은 예를 들어, 그래픽스 애플리케이션이 다양한 타임스탬프 프로세싱 기법들을 수행하는 것을 허용할 수도 있다. 예를 들어, 이러한 타임스탬프 값들은 드로우 콜 커맨드 이전에 및 이에 후속하여 커맨드 스트림에 배치되는 타임스탬프 요청들에 응답하여 반환되는 타임스탬프 값들 간의 차이를 취함으로써 드로우 콜 커맨드에 대한 적절한 실행 시간을 결정하는데 이용될 수도 있다.

[0022] 타일 기반 렌더링은, 일부 예들에서, 렌더 타겟 (예를 들어, 프레임) 을 복수의 서브-영역들 (예를 들어, 빈들 또는 타일들) 로 서브분할하는 것, 및 렌더 타겟의 서브-영역들 각각에 대해 별개의 렌더링 패스 반복을 포함하는 렌더링 패스를 수행하는 것을 수반할 수도 있다. 별개의 렌더링 패스 반복들을 수행하기 위해, 타일 기반 렌더링 시스템은 렌더링될 그래픽스 프레임과 연관된 드로우 콜 커맨드들의 실행을 복수의 빈당 드로우 콜들로 서브분할하고 빈당 드로우 콜들을 빈별로 그룹화할 수도 있다. 빈당 드로우 콜들의 그룹들 각각은 별개의 렌더링 패스 반복의 부분으로서 실행될 수도 있다.

[0023] 렌더링될 그래픽스 프레임이 다수의 드로우 콜들을 포함한다면, 하나의 드로우 콜 커맨드와 연관된 빈당 드로우 콜들의 실행은 동일한 그래픽스 프레임에 대한 다른 드로우 콜 커맨드들과 연관된 빈당 드로우 콜들의 실행과 인터리빙될 수도 있다. 그러나, 상기 논의한 바와 같이, 일부 타입들의 타임스탬프 프로세싱 기법들은 드로우 콜 커맨드들이 연속적인 방식으로 및 그래픽스 커맨드들이 커맨드 스트림에 배치되는 순서로 실행되는 것을 가정할 수도 있다. 타일 기반 렌더링을 수행할 때 일어나는 드로우 콜 커맨드들의 인터리빙된 실행은 이러한 타임스탬프 프로세싱 기법들에 대해 유용한 인트라-프레임 타임스탬프들을 제공하기 어렵게 만들 수도 있다.

[0024] 본 개시물의 기법들은 드로우 콜 커맨드들이 타일 기반 렌더링 기법들의 수행으로 인해 인터리빙된 방식으로 실행되는 경우라도 인트라-프레임 타임스탬프들을 생성하는데 이용될 수도 있다. 일부 예들에서, 본 개시물에 따라 생성된 인트라-프레임 타임스탬프 값들은 (드로우 콜들이 실제로는 인터리빙된 방식으로 실행될 수도 있지만) 연속적인, 순차의 드로우 콜 프로세싱이 수행되었다면 획득될 타임스탬프 값들을 모방 또는 근사화할 수도 있는 것과 동시에 커맨드 스트림에서의 드로우 콜 커맨드들에 대해 일어났던 상대적 실행 시간량들을 나타내는 타임스탬프 값들을 제공할 수도 있다. 이렇게 하여, 타임스탬프 값들은 드로우 콜 커맨드들이 연속적인 방식으로 및 그래픽스 커맨드들이 커맨드 스트림에 배치되는 순서로 실행되는 것을 가정하는 타임스탬프 프로세싱 기법들에 이용될 수도 있다.

[0025] 도 1 은 본 개시물의 인트라-프레임 타임스탬프 생성 기법들을 구현하는데 이용될 수도 있는 예시적인 컴퓨팅 디바이스 (2) 를 예시하는 블록 다이어그램이다. 컴퓨팅 디바이스 (2) 는 개인용 컴퓨터, 데스크톱 컴퓨터, 랩톱 컴퓨터, 컴퓨터 워크스테이션, 비디오 게임 플랫폼 또는 콘솔, 무선 통신 디바이스 (이를 테면, 예를 들어, 모바일 전화기, 셀룰러 전화기, 위성 전화기, 및/또는 모바일 폰 핸드셋), 랜드라인 전화기, 인터넷 전화기, 핸드헬드 디바이스, 이를 테면 휴대용 비디오 게임 디바이스 또는 개인 휴대 정보 단말기 (PDA), 개인용 뮤직 플레이어, 비디오 플레이어, 디스플레이 디바이스, 텔레비전, 텔레비전 셋톱 박스, 서버, 중간 네트워크 디바이스, 메인프레임 컴퓨터 또는 그래픽 데이터를 프로세싱 및/또는 디스플레이하는 임의의 다른 타입의 디바이스를 포함할 수도 있다.

[0026] 도 1 의 예에 예시한 바와 같이, 컴퓨팅 디바이스 (2) 는 사용자 인터페이스 (4), CPU (6), 메모리 제어기 (8), 메모리 (10), 그래픽스 프로세싱 유닛 (GPU) (12), 디스플레이 인터페이스 (14), 디스플레이 (16) 및 버스 (18) 를 포함한다. 사용자 인터페이스 (4), CPU (6), 메모리 제어기 (8), GPU (12) 및 디스플레이 인터페이스 (14) 는 버스 (18) 를 이용하여 서로 통신할 수도 있다. 도 1 에 도시된 상이한 컴포넌트들 간의 특정 구성의 버스들 및 통신 인터페이스들은 단지 예시적일 뿐이며, 동일하거나 상이한 컴포넌트들을 가진 다른 구성들의 컴퓨팅 디바이스들 및/또는 다른 그래픽스 프로세싱 시스템들이 본 개시물의 기법들을 구현하는데 이용될 수도 있다는 것에 주목해야 한다.

[0027] CPU (6) 는 컴퓨팅 디바이스 (2) 의 동작을 제어하는 범용 또는 특수 목적 프로세서를 포함할 수도 있다. 사용자는 컴퓨팅 디바이스 (2) 에 입력을 제공하여 CPU (6) 로 하여금 하나 이상의 소프트웨어 애플리케이션들을 실행하게 할 수도 있다. CPU (6) 상에서 실행되는 소프트웨어 애플리케이션들은, 예를 들어, 그래픽스 애플리케이션, 워드 프로세서 애플리케이션, 이메일 애플리케이션, 스프레드 시트 애플리케이션, 미디어 플레이어 애플리케이션, 비디오 게임 애플리케이션, 그래픽 사용자 인터페이스 애플리케이션, 오퍼레이팅 시스템, 또

는 임의의 다른 타입의 프로그램을 포함할 수도 있다. 사용자는 하나 이상의 입력 디바이스들 (미도시), 이를 테면 키보드, 마우스, 마이크로폰, 터치 패드 또는 사용자 인터페이스 (4) 를 통해 컴퓨팅 디바이스 (2) 에 커플링되는 다른 입력 디바이스를 통해 컴퓨팅 디바이스 (2) 에 입력을 제공할 수도 있다.

[0028] CPU (6) 상에서 실행되는 소프트웨어 애플리케이션들은 디스플레이 (16) 상의 디스플레이를 위해 프레임 버퍼에 그래픽스 데이터를 렌더링할 것을 GPU (12) 에 명령하는 하나 이상의 그래픽스 렌더링 명령들을 포함할 수도 있다. 일부 예들에서, 그래픽스 렌더링 명령들은 그래픽스 애플리케이션 프로그래밍 인터페이스 (API), 이를 테면, 예를 들어, 오픈 그래픽스 라이브러리 (OpenGL®) API, 오픈 그래픽스 라이브러리 임베디드 시스템들 (OpenGL ES) API, Direct3D API, X3D API, RenderMan API, WebGL API, 또는 임의의 다른 공공 또는 사유 표준 그래픽스 API 를 따를 수도 있다. 그래픽스 렌더링 명령들을 프로세싱하기 위하여, CPU (6) 는 GPU (12) 로 하여금 그래픽스 데이터의 렌더링의 일부 또는 전부를 수행하게 하기 위해 GPU (12) 에 하나 이상의 그래픽스 렌더링 커맨드들을 이슈할 수도 있다. 일부 예들에서, 렌더링될 그래픽스 데이터는 그래픽스 프리미티브들의 리스트, 예를 들어, 포인트들, 라인들, 삼각형들, 사각형들, 삼각형 스트립들 등을 포함할 수도 있다.

[0029] 메모리 제어기 (8) 는 메모리 (10) 로 들어가고 메모리 (10) 에서 나오는 데이터의 전송을 용이하게 한다. 예를 들어, 메모리 제어기 (8) 는 메모리 판독 및 기입 커맨드들을 수신하고, 이러한 커맨드들을 컴퓨팅 디바이스 (2) 에서의 컴포넌트들에 대해 메모리 서비스들을 제공하기 위하여 메모리 (10) 에 대하여 서비스할 수도 있다. 메모리 제어기 (8) 는 메모리 (10) 에 통신적으로 커플링된다. 메모리 제어기 (8) 는 도 1 의 예시적인 컴퓨팅 디바이스 (2) 에 CPU (6) 및 메모리 (10) 양자와는 별개인 프로세싱 모듈인 것으로 예시되지만, 다른 예들에서, 메모리 제어기 (8) 의 기능성의 일부 또는 전부는 CPU (6) 및 메모리 (10) 중 하나 또는 양자 상에서 구현될 수도 있다.

[0030] 메모리 (10) 는 CPU (6) 에 의한 실행을 위해 액세스가능한 프로그램 모듈들 및/또는 명령들 및/또는 CPU (6) 상에서 실행되는 프로그램들에 의한 이용을 위한 데이터를 저장할 수도 있다. 예를 들어, 메모리 (10) 는 CPU (6) 상에서 실행되는 애플리케이션들과 연관된 프로그램 코드 및 그래픽스 데이터를 저장할 수도 있다. 메모리 (10) 는 컴퓨팅 디바이스 (2) 의 다른 컴포넌트들에 의한 이용을 위한 및/또는 이들에 의해 생성된 정보를 추가적으로 저장할 수도 있다. 예를 들어, 메모리 (10) 는 GPU (12) 에 대한 디바이스 메모리의 역할을 할 수도 있고 GPU (12) 에 의해 동작될 데이터는 물론 GPU (12) 에 의해 수행된 동작들로부터 발생하는 데이터를 저장할 수도 있다. 예를 들어, 메모리 (10) 는 텍스처 버퍼들, 깊이 버퍼들, 스텐실 버퍼들, 버텍스 버퍼들, 프레임 버퍼들, 렌더 타겟들 등의 임의의 조합을 저장할 수도 있다. 또한, 메모리 (10) 는 GPU (12) 에 의한 프로세싱을 위한 커맨드 스트림들을 저장할 수도 있다. 메모리 (10) 는 하나 이상의 휘발성 또는 비휘발성 메모리들 또는 저장 디바이스들, 이를 테면, 예를 들어, 랜덤 액세스 메모리 (RAM), 정적 RAM (SRAM), 동적 RAM (DRAM), 판독 전용 메모리 (ROM), 소거가능한 프로그램가능 ROM (EPROM), 전기적으로 소거가능한 프로그램가능 ROM (EEPROM), 플래시 메모리, 자기 데이터 매체 또는 광학 저장 매체를 포함할 수도 있다.

[0031] GPU (12) 는 CPU (6) 에 의해 GPU (12) 에 이슈되는 커맨드들을 실행하도록 구성될 수도 있다. GPU (12) 에 의해 실행된 커맨드들은 그래픽스 커맨드들, 드로우 콜 커맨드들, GPU 상태 프로그래밍 커맨드들, 타임스탬프 요청들, 메모리 전송 커맨드들, 범용 컴퓨팅 커맨드들, 커널 실행 커맨드들 등을 포함할 수도 있다.

[0032] 일부 예들에서, GPU (12) 는 디스플레이 (16) 에 하나 이상의 그래픽스 프리미티브들을 렌더링하기 위해 그래픽스 동작들을 수행하도록 구성될 수도 있다. 이러한 예들에서, CPU (6) 상에서 실행되는 소프트웨어 애플리케이션들 중 하나가 그래픽스 프로세싱을 요구할 때, CPU (6) 는 GPU (12) 에 그래픽스 데이터를 제공하고 하나 이상의 그래픽스 커맨드들을 GPU (12) 에 이슈할 수도 있다. 그래픽스 커맨드들은 예를 들어, 드로우 콜 커맨드들, GPU 상태 프로그래밍 커맨드들, 메모리 전송 커맨드들, 블리팅 (blitting) 커맨드들 등을 포함할 수도 있다. 그래픽스 데이터는 버텍스 버퍼들, 텍스처 데이터, 표면 데이터 등을 포함할 수도 있다. 일부 예들에서, CPU (6) 는 GPU (12) 에 의해 액세스될 수도 있는 메모리 (10) 에 커맨드들 및 그래픽스 데이터를 기입함으로써 커맨드들 및 그래픽스 데이터를 GPU (12) 에 제공할 수도 있다.

[0033] 추가 예들에서, GPU (12) 는 CPU (6) 상에서 실행되는 애플리케이션들에 대해 범용 컴퓨팅을 수행하도록 구성될 수도 있다. 이러한 예들에서, CPU (6) 상에서 실행되는 소프트웨어 애플리케이션들 중 하나가 GPU (12) 에 컴퓨테이션 태스크를 오프로딩하기로 결정하는 경우, CPU (6) 는 GPU (12) 에 범용 컴퓨팅 데이터를 제공하고, GPU (12) 에 하나 이상의 범용 컴퓨팅 커맨드들을 이슈할 수도 있다. 범용 컴퓨팅 커맨드들은 예를 들어, 커널 실행 커맨드들, 메모리 전송 커맨드들 등을 포함할 수도 있다. 일부 예들에서, CPU (6) 는 GPU (12) 에 의해 액세스될 수도 있는 메모리 (10) 에 커맨드들 및 범용 컴퓨팅 데이터를 기입함으로써 GPU (12) 에 커맨

드들 및 범용 컴퓨팅 데이터를 제공할 수도 있다.

[0034] GPU (12) 는 일부 인스턴스들에서, CPU (6) 보다 더 효율적인 벡터 연산들의 프로세싱을 제공하는 고도 병렬 구조로 구성될 수도 있다. 예를 들어, GPU (12) 는 병렬 방식으로 다수의 버텍스들, 제어 포인트들, 픽셀들 및/또는 다른 데이터에 대해 동작하도록 구성되는 복수의 프로세싱 엘리먼트들을 포함할 수도 있다. GPU (12) 의 고도 병렬 본질은 일부 인스턴스들에서, CPU (6) 를 이용하여 이미지들을 렌더링하는 것보다 더 신속히 GPU (12) 가 디스플레이 (16) 상으로 그래픽스 이미지들 (예를 들어, GUI들 및 2 차원 (2D) 및/또는 3 차원 (3D) 그래픽스 장면들) 을 렌더링하는 것을 허용할 수도 있다. 또한, GPU (12) 의 고도 병렬 본질은 CPU (6) 보다 더 신속히 GPU (12) 가 범용 컴퓨팅 애플리케이션들에 대한 소정의 타입들의 벡터 및 매트릭스 동작들을 프로세싱하는 것을 허용할 수도 있다.

[0035] GPU (12) 는 일부 인스턴스들에서, 컴퓨팅 디바이스 (2) 의 마더보드에 통합될 수도 있다. 다른 인스턴스들에서, GPU (12) 는 컴퓨팅 디바이스 (2) 의 마더보드에서의 포트에 설치되는 그래픽스 카드 상에 존재할 수도 있고 또는 다르게는 컴퓨팅 디바이스 (2) 와 상호동작하도록 구성된 주변 디바이스 내에 통합될 수도 있다. 추가 인스턴스들에서, GPU (12) 는 시스템 온 칩 (SoC) 을 형성하는 CPU (6) 와 동일한 마이크로칩 상에 위치할 수도 있다. GPU (12) 는 하나 이상의 프로세서들, 이를 테면 하나 이상의 마이크로프로세서들, 주문형 집적 회로들 (ASIC들), 필드 프로그램가능 게이트 어레이들 (FPGA들), 디지털 신호 프로세서들 (DSP들), 또는 다른 등가의 집적 또는 이산 로직 회로를 포함할 수도 있다.

[0036] 일부 예들에서, GPU (12) 는 메모리 (10) 의 전부 또는 일부에 대해 캐싱 서비스들을 제공할 수도 있는 GPU 캐시를 포함할 수도 있다. 이러한 예들에서, GPU (12) 는 오프-칩 메모리 대신에, 로컬 스토리지를 이용하여 데이터를 로컬로 프로세싱하기 위해 캐시를 이용할 수도 있다. 이것은 각각의 판독 및 기입 커맨드 동안, 과중한 버스 트래픽을 경험할 수도 있는, 버스 (18) 를 통해 GPU (12) 가 메모리 (10) 에 액세스할 필요성을 감소시킴으로써 보다 효율적인 방식으로 GPU (12) 가 동작하는 것을 허용한다. 그러나, 일부 예들에서, GPU (12) 는 별개의 캐시를 포함하지 않고 그 대신 버스 (18) 를 통해 메모리 (10) 를 활용할 수도 있다. GPU 캐시는 하나 이상의 휘발성 또는 비휘발성 메모리들 또는 저장 디바이스들, 이를 테면, 예를 들어, 랜덤 액세스 메모리 (RAM), 정적 RAM (SRAM), 동적 RAM (DRAM) 등을 포함할 수도 있다.

[0037] CPU (6) 및/또는 GPU (12) 는 메모리 (10) 내에 할당되는 프레임 버퍼에 래스터화된 이미지 데이터를 저장할 수도 있다. 디스플레이 인터페이스 (14) 는 프레임 버퍼로부터 데이터를 취출하고 래스터화된 이미지 데이터에 의해 표현된 이미지를 디스플레이하도록 디스플레이 (16) 를 구성할 수도 있다. 일부 예들에서, 디스플레이 인터페이스 (14) 는 프레임 버퍼로부터 취출된 디지털 값들을 디스플레이 (16) 에 의해 소비가능한 아날로그 신호로 변환하도록 구성되는 디지털-아날로그 컨버터 (DAC) 를 포함할 수도 있다. 다른 예들에서, 디스플레이 인터페이스 (14) 는 프로세싱을 위해 디스플레이 (16) 에 직접 디지털 값들을 전달할 수도 있다.

[0038] 디스플레이 (16) 는 모니터, 텔레비전, 프로젝션 디바이스, 액정 디스플레이 (LCD), 플라즈마 디스플레이 패널, 발광 다이오드 (LED) 어레이, 음극선관 (CRT) 디스플레이, 전자 페이퍼, SED (surface-conduction electron-emitted display), 레이저 텔레비전 디스플레이, 나노크리스탈 디스플레이 또는 다른 타입의 디스플레이 유닛을 포함할 수도 있다. 디스플레이 (16) 는 컴퓨팅 디바이스 (2) 내에 통합될 수도 있다. 예를 들어, 디스플레이 (16) 는 모바일 전화기 핸드셋 또는 태블릿 컴퓨터의 스크린일 수도 있다. 대안적으로, 디스플레이 (16) 는 유선 또는 무선 통신 링크를 통해 컴퓨팅 디바이스 (2) 에 커플링된 스탠드-얼론 디바이스일 수도 있다. 예를 들어, 디스플레이 (16) 는 케이블 또는 무선 링크를 통해 개인용 컴퓨터에 접속된 컴퓨터 모니터 또는 평판 디스플레이일 수도 있다.

[0039] 버스 (18) 는 제 1, 제 2 및 제 3 세대 버스 구조들 및 프로토콜들, 공유 버스 구조들 및 프로토콜들, 점대점 버스 구조들 및 프로토콜들, 단방향 버스 구조들 및 프로토콜들, 및 양방향 버스 구조들 및 프로토콜들을 포함하는 버스 구조들 및 버스 프로토콜들의 임의의 조합을 이용하여 구현될 수도 있다. 버스 (18) 를 구현하는데 이용될 수도 있는 상이한 버스 구조들 및 프로토콜들의 예들은 예를 들어, HyperTransport 버스, InfiniBand 버스, 어드밴스드 그래픽스 포트 버스, 주변 컴포넌트 인터커넥트 (PCI) 버스, PCI 익스프레스 버스, 어드밴스드 마이크로제어기 버스 아키텍처 (AMBA) 어드밴스드 고성능 버스 (AHB), AMBA 어드밴스드 주변 버스 (APB), 및 AMBA AXI (Adavanced eXentisible Interface) 버스를 포함한다. 다른 타입들의 버스 구조들 및 프로토콜들이 또한 이용될 수도 있다.

[0040] 본 개시물에 따르면, 컴퓨팅 디바이스 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 는 본 개시물에 설명된 인트라-프레임 타임스탬프 값 생성 기법들 중 임의의 것을 수행하도록 구성될 수도 있다. 예를 들어, 컴퓨팅 디

바이스 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU (12) 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하도록 구성될 수도 있다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다. 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용하는 것은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에 의해 인트라-프레임 타임스탬프 요청들이 지원되는 것을 허용할 수도 있다.

[0041] 동작 동안, CPU (6) 상에서 실행되는 그래픽스 애플리케이션은 그래픽스 프레임을 렌더링하기 위해 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림) 을 생성할 수도 있다. 일부 경우들에서, 순서화된 시퀀스의 커맨드들은 복수의 드로우 콜 커맨드들 및 복수의 타임스탬프 요청들을 포함할 수도 있다. 타임스탬프 요청들 중 적어도 일부는 순서화된 시퀀스의 커맨드들에서의 상이한 드로우 콜 커맨드들 사이에 배치될 수도 있다.

[0042] 타일 기반 렌더링 기법들을 이용하여 그 시퀀스의 커맨드들을 실행하기 위해, CPU (6) 는 타임스탬프 요청들 각각에 대해, 각각의 타임스탬프 요청에 기초하여 복수의 빈당 타임스탬프 요청들을 생성할 수도 있다. CPU (6) 는 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치할 수도 있다.

커맨드 스트림들 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 실행될 수도 있다. 커맨드 스트림들은 빈당 커맨드 스트림들로 지칭될 수도 있다. 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하도록 구성될 수도 있다.

[0043] CPU (6) 는 GPU (12) 로 하여금 빈당 커맨드 스트림들을 실행하게 할 수도 있다. 빈당 커맨드 스트림들을 실행하는 동안, GPU (12) 는 GPU (12) 에 의해 수신된 빈당 커맨드 스트림들에서의 빈당 타임스탬프 요청들을 실행하는 것에 응답하여 빈당 타임스탬프 값들을 생성할 수도 있다. 일부 경우들에서, GPU (12) 는 빈당 커맨드 스트림들에 포함된 빈당 타임스탬프 요청들 각각에 대한 각각의 빈당 타임스탬프 값을 생성할 수도 있다.

빈당 타임스탬프 값들 각각은 각각의 빈당 타임스탬프 값과 연관된 빈당 타임스탬프 요청이 GPU (12) 에 의해 실행되었던 시간을 나타낼 수도 있다. 일부 예들에서, 빈당 타임스탬프 값들 각각은 GPU (12) 에 의해 생성된 각각의 빈당 타임스탬프에 포함될 수도 있다.

[0044] 일부 예들에서, GPU (12) 는 CPU (6) 에 빈당 타임스탬프 값들을 제공할 수도 있다. 빈당 타임스탬프 값들을 수신하는 것에 응답하여, CPU (6) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. CPU (6) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. CPU (6) 는 그래픽스 애플리케이션에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0045] 추가 예들에서, GPU (12) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프 값들을 생성하고, 애플리케이션 요청 타임스탬프 값들을 CPU (6) 에 제공할 수도 있다. GPU (12) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. CPU (6) 는 그래픽스 애플리케이션에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0046] 추가적인 예들에서, GPU (12) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 중간 값들을 생성하고, 중간 값들을 CPU (6) 에 제공할 수도 있다. CPU (6) 는 중간 값들에 기초하여 빈당 타임스탬프 값들을 생성할 수도 있다. CPU (6) 및 GPU (12) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. CPU (6) 는 그래픽스 애플리케이션에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0047] 일부 예들에서, CPU (6) 및/또는 GPU (12) 는 각각의 애플리케이션 요청 타임스탬프 값들이 적어도 2 개의 상이한 빈당 타임스탬프 값들의 함수가 되도록 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. 이러한 예들에서, 적어도 2 개의 상이한 빈당 타임스탬프 값들은 일부 예들에서, 상이한 렌더링 패스 반복들 동안 생성될 수도 있다. 상이한 렌더링 패스 반복들 동안 생성되는 빈당 타임스탬프 값들을 이용하는 것은 그래픽스 프레임의 렌더링 동안 상이한 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 적어도 어느 정도까지는 반영하는 애플리케이션 요청 타임스탬프 값들을 그래픽스 프로세싱 시스템이 생성하는 것을 허용할 수도 있다. 이렇게 하여, 본 개시물에 설명된 기법들은 타일 기반 렌더링 시스템이 비연속적인 인터리빙된 방식으로 드로우 콜 커맨드들을 실행하는 경우들에도, 타일 기반 렌더링 시스템이 인트라-프레임 타임스탬프들을 지원하는 것을 허용할 수도 있다.

[0048] 일부 예들에서, 컴퓨팅 디바이스 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 는 타일 기반 렌더링을 수행할 때 수신되는 인트라-프레임 타임스탬프 요청들에 응답하여 인트라-프레임 타임스탬프들을 생성하도록 구성될 수도 있다. 추가 예들에서, 컴퓨팅 디바이스 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 는 렌더링될 그래픽스 프레임에 대해 실행될 복수의 드로우 콜 커맨드들 및 렌더링될 그래픽스 프레임과 연관된 하나 이상의 타임스탬프 요청들을 포함하는 커맨드 스트림을 수신하도록 구성될 수도 있다. 이러한 예들에서, 컴퓨팅 디바이스 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 는 또한, 일부 예들에서는, GPU (12) 로 하여금, 타일 기반 렌더링 기법들을 이용하여 렌더링될 그래픽스 프레임에 대해 복수의 드로우 콜 커맨드들을 실행하게 하고, 하나 이상의 타임스탬프 요청들에 응답하여 하나 이상의 타임스탬프들을 생성하게 하도록 구성될 수도 있다. 일부 예들에서, 하나 이상의 타임스탬프들은 복수의 빈당 타임스탬프 값들에 기초하여 생성될 수도 있다.

[0049] 본 개시물에 설명된 기법들은 일부 예들에서, 예를 들어, CPU (6), GPU (12), 및 시스템 메모리 (10) 를 포함하는 도 1 에 예시된 컴퓨팅 디바이스 (2) 에서의 컴포넌트들 중 임의의 컴포넌트에서 구현될 수도 있다. 예를 들어, 인트라-프레임 타임스탬프들을 생성하기 위한 기법들은 CPU (6) 에서의 그래픽스 드라이버, GPU (12) 에서의 프로세싱 유닛, 또는 이들의 조합에 의해 수행될 수도 있다. 다른 예로서, 타임스탬프 요청은 CPU (6) 상에서 실행되는 소프트웨어 애플리케이션 (예를 들어, 그래픽스 애플리케이션 또는 사용자 애플리케이션) 에 의해 CPU (6) 상에서 실행되는 GPU 드라이버에 이슈될 수도 있고, 타임스탬프 요청에 응답하여, GPU 드라이버는 본 개시물의 기법들에 따라 생성된 타임스탬프를 반환할 수도 있다. 일부 예들에서, 타임스탬프 요청들 및 드로우 콜들은 메모리 (10) 에 (예를 들어, 하나 이상의 커맨드 큐들의 부분으로서) 저장될 수도 있다. 추가 예들에서, 타임스탬프 요청들에 응답하여 반환된 타임스탬프들은 메모리 (10) 에 저장될 수도 있다.

[0050] 도 2 는 도 1 에서의 컴퓨팅 디바이스 (2) 의 CPU (6), GPU (12) 및 메모리 (10) 를 더욱 상세히 예시하는 블록 다이어그램이다. 도 2 에 도시한 바와 같이, CPU (6) 는 GPU (12) 및 메모리 (10) 에 통신적으로 커플링되고, GPU (12) 는 CPU (6) 및 메모리 (10) 에 통신적으로 커플링된다. GPU (12) 는 일부 예들에서, CPU (6) 와 함께 마더보드 상에 통합될 수도 있다. 추가적인 예들에서, GPU (12) 는 CPU (6) 를 포함하는 마더보드의 포트에 설치되는 그래픽스 카드 상에 구현될 수도 있다. 추가 예들에서, GPU (12) 는 CPU (6) 와 상호동작하도록 구성되는 주변 디바이스 내에 통합될 수도 있다. 추가적인 예들에서, GPU (12) 는 시스템 온 칩 (SoC) 을 형성하는 CPU (6) 와 동일한 마이크로칩 상에 위치할 수도 있다.

[0051] CPU (6) 는 소프트웨어 애플리케이션 (24), 그래픽스 API (26), GPU 드라이버 (28) 및 오퍼레이팅 시스템 (30) 중 임의의 것을 실행하도록 구성되는 하나 이상의 프로세서들 (예를 들어, 마이크로프로세서들) 을 포함할 수도 있다. 일부 예들에서, CPU (6) 는 CPU (6) 의 하나 이상의 프로세서들로 하여금 본 개시물에 설명된 기법들 중 임의의 기법의 전부 또는 부분을 수행하게 하는 명령들을 실행하도록 구성될 수도 있다.

[0052] GPU (12) 는 커맨드 엔진 (32), 하나 이상의 프로세싱 유닛들 (34), 및 비닝 버퍼 (36) 를 포함한다. 하나 이상의 프로세싱 유닛들 (34) 은 3D 그래픽스 렌더링 파이프라인을 형성하도록 구성될 수도 있다. 일부 예들에서, 하나 이상의 프로세싱 유닛들 (34) 은 온-칩, 테셀레이션-가능 그래픽스 렌더링 파이프라인을 구현할 수도 있다. 커맨드 엔진 (32) 및 프로세싱 유닛들 (34) 은 이러한 컴포넌트들에 기인한 기능들을 수행하도록 구성되는 전용 하드웨어 유닛들, 펌웨어, 소프트웨어, 및 프로세서들의 임의의 조합을 포함할 수도 있다. 일부 예들에서, GPU (12) 는 GPU (12) 의 하나 이상의 프로세서들로 하여금 본 개시물에 설명된 기법들 중 임의의 것의 전부 또는 부분을 수행하게 하는 명령들을 실행하도록 구성될 수도 있다.

[0053] 메모리 (10) 는 하나 이상의 커맨드들 (38), 프리미티브 데이터 (40), 및 타임스탬프 데이터 (42) 를 저장할 수도 있다. 일부 예들에서, 메모리 (10) 는 또한, 실행될 때, 하나 이상의 프로세서들로 하여금, 본 개시물에 설명된 기법들 중 임의의 것의 전부 또는 부분을 수행하게 하는 명령들을 저장할 수도 있다.

[0054] 소프트웨어 애플리케이션 (24) 은 GPU (12) 를 이용하여, 하나 이상의 3D 그래픽스 장면들 및/또는 3D 그래픽스 오브젝트들을 디스플레이 상에 디스플레이될 이미지로 렌더링하는 그래픽스 애플리케이션일 수도 있다. 소프트웨어 애플리케이션 (24) 은 GPU (12) 로 하여금, 3D 그래픽스 프리미티브들의 세트를 래스터화 및 렌더링하게 하는 명령들을 포함할 수도 있다. 소프트웨어 애플리케이션 (24) 은 그래픽스 API (26) 를 통해 GPU 드라이버 (28) 에 명령들을 이슈할 수도 있다. 그래픽스 API (26) 는 소프트웨어 애플리케이션 (24) 으로부터 수신된 명령들을 GPU 드라이버 (28) 에 의해 소비가능한 포맷으로 트랜슬레이트하는 런타임 서비스일 수도 있다.

[0055] GPU 드라이버 (28) 는 그래픽스 API (26) 를 통해 소프트웨어 애플리케이션 (24) 으로부터 명령들을 수신하고, 명령들을 서비스하기 위해 GPU (12) 의 동작을 제어한다. 예를 들어, GPU 드라이버 (28) 는 하나 이상의 커

맨드들 (38) 을 만들어내고, 커맨드들 (38) 을 메모리 (10) 에 배치하며, 커맨드들 (38) 을 실행할 것을 GPU (12) 에 명령할 수도 있다. 일부 예들에서, GPU 드라이버 (28) 는 커맨드들 (38) 을 메모리 (10) 에 배치하고 오퍼레이팅 시스템 (30) 을 통해, 예를 들어, 하나 이상의 시스템 콜들을 통해 GPU (12) 와 통신할 수도 있다.

[0056] 오퍼레이팅 시스템 (30) 은 소프트웨어 애플리케이션 (24), 그래픽스 API (26), 및 GPU 드라이버 (28) 가 실행하는 소프트웨어 플랫폼을 제공할 수도 있다. 오퍼레이팅 시스템 (30) 은 CPU (6), 메모리 (10) 및 GPU (12) 사이에서 데이터를 통신 및 전송하는 하드웨어 상세들을 관리할 수도 있다.

[0057] 커맨드들 (38) 은 하나 이상의 상태 커맨드들, 하나 이상의 드로우 콜 커맨드들 및/또는 하나 이상의 타임스탬프 요청들 (예를 들어, 하나 이상의 빈당 타임스탬프 요청들) 을 포함할 수도 있다. 상태 커맨드는 GPU (12) 에서의 상태 변수들 중 하나 이상, 이를 테면, 예를 들어, 프리미티브 타입을 변경할 것을 GPU (12) 에 명령할 수도 있다. 드로우 콜 커맨드는 메모리 (10) 에 저장된 하나 이상의 버텍스들의 그룹에 의해 정의된 (예를 들어, 버텍스 버퍼에 정의된) 지오메트리를 렌더링할 것을 GPU (12) 에 명령할 수도 있다. 하나 이상의 버텍스들의 그룹에 의해 정의된 지오메트리는 일부 예들에서, 렌더링될 복수의 프리미티브들 (예를 들어, 프리미티브 데이터 (40)) 에 대응할 수도 있다. 일반적으로, 드로우 콜 커맨드는 메모리 (10) 의 정의된 섹션 (예를 들어, 버퍼) 에 저장된 모든 버텍스들을 렌더링하기 위해 GPU (12) 를 인보크할 수도 있다. 즉, 일단 GPU (12) 가 드로우 콜 커맨드를 수신하면, 메모리 (10) 의 정의된 섹션 (예를 들어, 버퍼) 에서의 버텍스들에 의해 표현된 지오메트리 및 프리미티브들을 렌더링하기 위한 제어가 GPU (12) 에 전달된다.

[0058] 타임스탬프 요청은 타임스탬프 요청을 프로세싱하는 것에 응답하여 타임스탬프를 생성할 것을 GPU (12) 및/또는 GPU 드라이버 (28) 에 명령할 수도 있다. GPU (12) 및/또는 GPU 드라이버 (28) 는 타임스탬프 요청을 수신하는 것에 응답하여 타임스탬프를 반환할 수도 있다. 타임스탬프는 시간 값을 포함할 수도 있다. 빈당 타임스탬프 요청들의 경우, 시간 값은 GPU (12) 가 타임스탬프 요청을 프로세싱한 시간을 나타낼 수도 있다. 애플리케이션 요청 타임스탬프 요청들의 경우, 시간 값은 GPU (12) 에 의해 커맨드 스트림에서의 드로우 콜들을 수행하는데 걸리는 상대적 시간량들을 나타낼 수도 있다.

[0059] 일부 예들에서, 커맨드들 (38) 은 커맨드 스트림의 형태 (예를 들어, 커맨드 큐, 커맨드 버퍼 등) 로 저장될 수도 있다. 커맨드 스트림은 순서화된 시퀀스의 그래픽스 커맨드들을 특정할 수도 있다. 일부 예들에서, 순서화된 시퀀스의 커맨드들은 복수의 드로우 콜 커맨드들 및 복수의 타임스탬프 요청들을 포함할 수도 있다. 일부 예들에서, 타임스탬프 요청들 중 적어도 하나는 순서화된 시퀀스의 그래픽스 커맨드들에서의 드로우 콜 커맨드들 중 적어도 2 개의 드로우 콜 커맨드들 사이에 포지셔닝될 수도 있다. 추가 예들에서, 드로우 콜 커맨드들 중 적어도 하나는 순서화된 시퀀스의 그래픽스 커맨드들에서의 타임스탬프 요청들 중 적어도 2 개의 타임스탬프 요청들 사이에 포지셔닝될 수도 있다.

[0060] 커맨드 엔진 (32) 은 메모리 (10) 에 저장된 커맨드들 (38) 을 취출 및 실행하도록 구성된다. 커맨드 엔진 (32) 은 GPU (12) 의 렌더링 상태를 관리하고, 프로세싱 유닛들 (34) 이 그래픽스 렌더링 파이프라인을 구현하도록 프로세싱 유닛들 (34) 의 동작을 제어하고, 그래픽스 데이터로 하여금, 그래픽스 렌더링 파이프라인을 통해 렌더 타겟으로 렌더링되게 하며, 커맨드 스트림에서의 타임스탬프 요청 (예를 들어, 빈당 타임스탬프 요청) 을 조우하는 것에 응답하여 타임스탬프들을 반환할 수도 있다.

[0061] 상태 커맨드를 수신하는 것에 응답하여, 커맨드 엔진 (32) 은 GPU 에서의 하나 이상의 상태 레지스터들을 상태 커맨드에 기초하여 특정 값들로 설정하고, 및/또는 상태 커맨드에 기초하여 고정 함수 프로세싱 유닛들 (34) 중 하나 이상을 구성하도록 구성될 수도 있다. 드로우 콜 커맨드를 수신하는 것에 응답하여, 커맨드 엔진 (32) 은 프로세싱 유닛들 (34) 로 하여금, 메모리 (10) 에서 버텍스들에 의해 표현된 지오메트리 (예를 들어, 프리미티브 데이터 (40) 에 의해 표현된 지오메트리) 를 렌더링하게 하도록 구성될 수도 있다. 커맨드 엔진 (32) 은 또한, 셰이더 프로그램 바인딩 커맨드들을 수신하고, 셰이더 프로그램 바인딩 커맨드들에 기초하여 특정 셰이더 프로그램들을 프로그램가능 프로세싱 유닛들 (34) 중 하나 이상으로 로드할 수도 있다. 타임스탬프 요청 (예를 들어, 빈당 타임스탬프 요청) 을 수신하는 것에 응답하여, 커맨드 엔진 (32) 은 타임스탬프를 생성하고 타임스탬프를 CPU (6) (예를 들어, GPU 드라이버 (28)) 에 제공할 수도 있다.

[0062] 프로세싱 유닛들 (34) 은 각각이 프로그램가능 프로세싱 유닛 또는 고정 함수 프로세싱 유닛일 수도 있는 하나 이상의 프로세싱 유닛들을 포함할 수도 있다. 프로그램가능 프로세싱 유닛은 예를 들어, CPU (6) 로부터 GPU (12) 로 다운로드되는 하나 이상의 셰이더 프로그램들을 실행하도록 구성되는 프로그램가능 셰이더 유닛을 포함할 수도 있다. 셰이더 프로그램은, 일부 예들에서, 하이-레벨 셰이딩 언어, 이를 테면, 예를 들어,

OpenGL 셰이딩 언어 (GLSL), 하이 레벨 셰이딩 언어 (HLSL), Cg (C for Graphics) 셰이딩 언어 등으로 기입된 프로그램의 컴파일링된 버전일 수도 있다.

[0063] 프로그램가능 셰이더 유닛은 일부 예들에서, 병렬, 예를 들어, 단일 명령 다중 데이터 (SIMD) 파이프라인으로 동작하도록 구성되는 복수의 프로세싱 유닛들을 포함할 수도 있다. 프로그램가능 셰이더 유닛은 셰이더 프로그램 명령들을 저장하는 프로그램 메모리 및 실행 상태 레지스터, 예를 들어, 실행중인 프로그램 메모리에서의 현재 명령 또는 페칭될 다음 명령을 나타내는 프로그램 카운터 레지스터를 가질 수도 있다. 프로세싱 유닛들 (34) 에서의 프로그램가능 셰이더 유닛들은, 예를 들어, 버텍스 셰이더 유닛들, 픽셀 셰이더 유닛들, 지오메트리 셰이더 유닛들, 헐 셰이더 유닛들, 도메인 셰이더 유닛들, 컴퓨트 셰이더 유닛들, 및/또는 통합된 셰이더 유닛들을 포함할 수도 있다.

[0064] 고정 함수 프로세싱 유닛은 소정의 기능들을 수행하기 위해 하드-와이어링되는 하드웨어를 포함할 수도 있다. 고정 함수 하드웨어는 하나 이상의 제어 신호들을 통해, 예를 들어 상이한 기능들을 수행하도록 구성가능할 수도 있지만, 고정 함수 하드웨어는 통상은 사용자 컴파일링된 프로그램들을 수신가능한 프로그램 메모리를 포함하지 않는다. 일부 예들에서, 프로세싱 유닛들 (34) 에서의 고정 함수 프로세싱 유닛들은 예를 들어, 래스터 동작들, 이를 테면, 예를 들어, 깊이 테스팅, 시저스 테스팅 (scissors testing), 알파 블렌딩 등을 수행하는 프로세싱 유닛들을 포함할 수도 있다.

[0065] 비닝 버퍼 (36) 는 렌더 타겟의 서브-영역에 대한 래스터화된 데이터를 저장하도록 구성될 수도 있다. 비닝 버퍼 (36) 는 렌더링 패스의 수행 동안 실제 렌더 타겟의 특정 서브-영역들에 대한 임시의 렌더 타겟의 역할을 할 수도 있다. 비닝 버퍼 (36) 는 하나 이상의 휘발성 또는 비휘발성 메모리들 또는 저장 디바이스들, 이를 테면, 예를 들어, 랜덤 액세스 메모리 (RAM), 정적 RAM (SRAM), 동적 RAM (DRAM) 등을 포함할 수도 있다. 일부 예들에서, 비닝 버퍼 (36) 는 온-칩 버퍼일 수도 있다. 온-칩 버퍼는 마이크로칩, 집적 회로, 및/또는 마이크로칩, 집적 회로 및/또는 GPU 가 형성, 위치 및/또는 배치되는 다이와 동일한 다이 상에 형성, 위치, 및/또는 배치되는 버퍼를 지칭할 수도 있다.

[0066] 일부 예들에서, 프로세싱 유닛들 (34) 은 제 1 통신 인터페이스를 통해 비닝 버퍼 (36) 에 액세스하고, 제 1 통신 인터페이스와는 상이한 제 2 통신 인터페이스를 통해 렌더 타겟 (예를 들어, 메모리 (10) 에 저장된 프레임 버퍼) 에 액세스할 수도 있다. 이러한 예들에서, 제 1 통신 인터페이스는 일부 예들에서, 제 2 통신 인터페이스보다 더 높은 대역폭을 가질 수도 있다. 제 2 통신 인터페이스는 일부 예들에서는, 도 1 의 버스 (18) 및 도 1 의 메모리 제어기 (8) 및 메모리 (10) 사이의 커넥션에 대응할 수도 있다. 비닝 버퍼 (36) 가 온-칩 빈 버퍼인 경우, 제 1 통신 인터페이스는 GPU (12) 의 내부에 있는 통신 인터페이스일 수도 있다.

[0067] 본 명세서에 사용한 바와 같이, 대역폭은 통신 인터페이스가 2 개의 컴포넌트들, 예를 들어, 메모리 컴포넌트와 GPU (12) 사이에서 데이터를 전송가능한 레이트를 지칭할 수도 있다. 대역폭에 대한 단위들은 일부 예들에서, 시간 단위당 비트수, 예를 들어, 초당 기가비트 (Gb/s) 로서 주어질 수도 있다. 다수의 비트들의 버스 폭을 갖는 버스가 통신 인터페이스의 부분으로서 이용되는 경우, 대역폭은, 일부 예들에서는, 버스의 폭과, 데이터가 단일 비트 라인을 따라 전송되는 레이트를 곱한 결과물과 동일할 수도 있다. 예를 들어, 버스가 16 비트 폭이며, 버스의 각각의 비트 라인이 2Gb/s 의 레이트에서 데이터를 전송가능하다면, 버스의 대역폭은 32Gb/s 와 동일할 수도 있다. 다수의 버스들이 2 개의 컴포넌트들 간에 통신 인터페이스를 형성한다면, 그 통신 인터페이스의 대역폭은 다수의 버스들 각각의 대역폭, 예를 들어, 개개의 버스들 각각의 최소 대역폭의 함수일 수도 있다.

[0068] 비닝 버퍼 (36) 가 GPU (12) 와 동일한 칩 상에 구현되는 경우, GPU (12) 는 시스템 및 메모리 버스들 (예를 들어, 도 1 의 버스 (18) 및 도 1 의 메모리 제어기 (8) 와 메모리 (10) 사이의 커넥션) 을 통해 비닝 버퍼 (36) 에 반드시 액세스할 필요가 없고, 오히려 GPU (12) 와 동일한 칩 상에 구현된 내부 통신 인터페이스 (예를 들어, 버스) 를 통해 비닝 버퍼 (36) 에 액세스할 수도 있다. 이러한 인터페이스는 온-칩이기 때문에, 그것은 시스템 및 메모리 버스들보다 더 높은 대역폭에서 동작가능할 수도 있다. 상기 설명된 기법은 메모리 (10) 에 액세스하는데 이용된 통신 인터페이스의 대역폭을 초과하는 비닝 버퍼 (36) 에 대한 통신 인터페이스를 달성하는 하나의 방식이지만, 다른 기법들이 가능하며 본 개시물의 범위 내에 있다.

[0069] 비닝 버퍼 (36) 의 용량은 일부 예들에서, 소정의 타입들의 컴퓨팅 디바이스들, 예를 들어, 모바일 디바이스들 상에서 이용가능한 면적에 의해 제한될 수도 있다. 더욱이, 비닝 버퍼 (36) 가 GPU (12) 와 동일한 칩 상에 구현되는 경우, 동일한 칩 상에 비닝 버퍼 (36) 를 구현하기 위해 이용가능한 면적의 양은 칩 상에 구현되는 다른 기능성으로 인해 제한될 수도 있다. 일부 예들에서, 비닝 버퍼 (36) 는 비닝 버퍼 (36) 의 용량을 더욱

제한하는 렌더 타겟의 비트 밀도보다 낮은 비트 밀도를 가질 수도 있다. 이들 및/또는 다른 팩터들 때문에, 비닝 버퍼 (36) 의 용량은 일부 경우들에서, 렌더 타겟의 사이즈보다 적을 수도 있다. 그 결과, 비닝 버퍼 (36) 의 용량은 이러한 예들에서, 그래픽스 이미지 (예를 들어, 단일 프레임) 과 연관된 복수의 목적지 픽셀들 전부에 대한 픽셀 데이터를 저장하는데 필요한 최소 용량보다 적을 수도 있다. 메모리 컴포넌트의 용량은 메모리 컴포넌트에 저장될 수 있는 최대 양의 데이터 (예를 들어, 최대 수의 비트들) 를 나타낼 수도 있다. 렌더 타겟의 사이즈는 렌더 타겟에 할당된 메모리 범위에 저장된 데이터의 양 (예를 들어, 비트들의 수) 을 지칭할 수도 있다. 비트 밀도는 특정 면적의 양에 저장될 수 있는 비트들의 수를 나타낼 수도 있다.

[0070] 상기 논의한 바와 같이, 타일 기반 렌더링을 수행할 때, GPU (12) 는 렌더링 패스의 별개의 반복 동안 렌더 타겟의 각각의 서브-영역을 렌더링할 수도 있다. 예를 들어, (예를 들어, 그래픽스 이미지의 목적지 픽셀들의 특정 서브세트인) 렌더 타겟의 특정 서브-영역에 대한 단일 렌더링 패스 반복의 부분으로서, GPU (12) 는 렌더 타겟의 특정 서브-영역에 대하여 프리미티브들의 전부 또는 서브세트를 렌더링할 수도 있다. 비닝 버퍼 (36) 의 용량은 렌더 타겟의 서브-영역의 사이즈 이상이 되도록 구성될 수도 있다. 따라서, 단일 렌더링 패스 반복 동안, 렌더 타겟의 서브-영역들 중 각각의 서브-영역과 연관된 모든 목적지 픽셀 데이터는 메모리 (10) 에서의 프레임 버퍼에 반드시 액세스할 필요 없이 비닝 버퍼 (36) 에서 이용가능할 수도 있다. 그 결과, 단일 렌더링 패스 반복 동안, GPU (12) 는 비교적 낮은 대역폭 통신 인터페이스를 통해 메모리 (10) 로부터 이러한 데이터를 판독해야 하는 것보다는 비교적 높은 대역폭 통신 인터페이스를 통해 비닝 버퍼 (36) 로부터 목적지 픽셀 데이터를 판독가능할 수도 있다.

[0071] 타일 기반 렌더링을 수행하지 않는 일부 그래픽스 시스템들은 하드웨어 기반 온-칩 캐시를 이용함으로써 프레임 버퍼의 부분을 캐시하는 것이 가능할 수도 있지만, 이러한 캐시들은 주어진 픽셀에 대한 목적지 픽셀 값들이 필요할 때 이용가능할 것을 보장하지 않는다. 이것은, 다수의 목적지 픽셀들이 하드웨어 기반 캐시에서 동일한 어드레스에 맵핑할 수도 있기 때문이다. 타일 기반 렌더링이 이용되지 않는다면, 하드웨어 기반 캐시의 현재 상태는 렌더 타겟의 현재 프로세싱된 서브-영역과 연관된 목적지 픽셀 값들을 반드시 포함하지 않을 수도 있고, 오히려 렌더 타겟의 다른 서브-영역들에서의 이전에 프로세싱된 프리미티브들과 연관된 목적지 픽셀 값들을 포함할 수도 있다.

[0072] 다수의 목적지 픽셀들이 동일한 캐시 위치에 맵핑하는 하드웨어 기반 캐시와 대조하여, 주어진 렌더링 패스 반복에 대한 비닝 버퍼 (36) 에 저장된 목적지 픽셀들은, 일부 예들에서, 고유하게 어드레스가능할 수도 있다. 즉, 주어진 렌더링 패스 반복의 경우, 일대일 맵핑이 비닝 버퍼 (36) 에서의 어드레스가능한 저장 슬롯들과 그 렌더링 패스 반복에 대해 이용된 목적지 픽셀들 사이에서 정의될 수도 있다. 그 결과, 타일 기반 렌더링을 수행할 때, 주어진 비닝 패스에 대한 모든 목적지 픽셀 값들은, 일부 예들에서, 비교적 낮은 대역폭 통신 인터페이스를 통해 비닝 버퍼 (36) 로부터 이용가능할 수도 있다. 더욱이, 하드웨어 기반 캐시 시스템들과 달리, 비닝 버퍼 (36) 에서의 고유하게 어드레스가능한 데이터 때문에, 캐시 미스들이 일어나지 않고, 이로써 캐시 미스의 이벤트에서 대역폭-소비성 (bandwidth-expensive) 프레임 버퍼 액세스들을 리소팅 (resorting) 할 필요성을 덜어준다.

[0073] 목적지 픽셀은 특정 픽셀 위치에 대한 렌더 타겟 (예를 들어, 프레임 버퍼나 대응하는 비닝 버퍼 중 어느 하나) 에 저장된 픽셀 데이터를 지칭할 수도 있다. 그에 반해, 소스 픽셀은 프로세싱 유닛들 (34) 에서의 래스터화 프로세싱 유닛에 의해 생성되었고 아직 렌더 타겟에 저장되고 및/또는 렌더 타겟과 병합되지 않은 픽셀 데이터를 지칭할 수도 있다. 목적지 픽셀은 상이한 프리미티브들과 연관된 다수의 소스 픽셀들로부터의 합성된 픽셀 데이터를 포함할 수도 있다.

[0074] 타일 기반 렌더링을 수행하기 위해, 소프트웨어 애플리케이션 (24) 은, 일부 예들에서, 프리미티브 데이터 (40) 를 렌더링될 하나 이상의 3D 그래픽스 프리미티브들의 세트를 지오메트릭적으로 정의하는 메모리 (10) 에 배치하고, 하나 이상의 드로우 콜 커맨드들을 그래픽스 API (26) 를 통해 GPU 드라이버 (28) 에 이슈할 수도 있다. 드로우 콜 커맨드들은 프리미티브 데이터 (40) 에 의해 정의된 프리미티브들로 하여금 GPU (12) 에 의해 렌더 타겟 (예를 들어, 메모리 (10) 에 저장된 프레임 버퍼) 으로 래스터화 및 렌더링되게 할 수도 있다.

[0075] 일부 예들에서, 소프트웨어 애플리케이션 (24) 은 특정 타입의 프리미티브를 렌더링하도록 GPU (12) 를 구성할 수도 있다. 예를 들어, 소프트웨어 애플리케이션 (24) 은 드로우 콜 동안 렌더링하기 위한 특정 타입의 프리미티브를 특정하는 상태 커맨드를 GPU (12) 에 이슈할 수도 있다. 추가적인 예들에서, 드로우 콜 커맨드들을 이슈하기 전에, 소프트웨어 애플리케이션 (24) 은 프리미티브를 렌더링하기 위해 하나 이상의 테셀레이션 기법들을 이용하도록 GPU (12) 를 구성할 수도 있다. 예를 들어, 소프트웨어 애플리케이션 (24) 은 테셀레

이션 기법들을 구현하는 하나 이상의 셰이더 프로그램들로 하여금, 드로우 콜 명령 동안 GPU (12) 의 하나 이상의 셰이더 유닛들 (예를 들어, 헐 셰이더 유닛 및/또는 도메인 셰이더 유닛) 상에서 실행되게 할 수도 있다.

[0076] 프리미티브 데이터 (40) 는 렌더링될 하나 이상의 프리미티브들을 나타내는 데이터를 포함할 수도 있다. 일부 경우들에서, 프리미티브 데이터 (40) 는 렌더링될 프리미티브들을 지오메트릭적으로 정의할 수도 있다. 프리미티브를 지오메트릭적으로 정의하는 것은 일 세트의 버텍스들 (또는 제어 포인트들) 및 대응하는 버텍스 속성들에 의해 프리미티브를 정의하는 것을 지칭할 수도 있다. 일부 예들에서, 프리미티브 데이터 (40) 는 복수의 버텍스들, 버텍스 리스트, 및/또는 버텍스 버퍼의 형태를 취할 수도 있다. 추가 예들에서, 프리미티브 데이터 (40) 는 인덱스 버퍼와 조합한 버텍스 버퍼의 형태를 취할 수도 있다. 이러한 예들에서, 버텍스 버퍼는 버텍스들을 정의할 수도 있고, 인덱스 버퍼는 어느 버텍스들이 프리미티브들 각각을 정의하는데 이용되는지를 특정할 수도 있다.

[0077] 프리미티브 데이터 (40) 에 포함된 버텍스들 각각은 하나 이상의 속성들, 이를 테면, 예를 들어, 포지션 좌표들, 기준 좌표들, 텍스처 좌표들 등을 포함할 수도 있다. 버텍스들은 개념적으로 지오메트릭 프리미티브 (예를 들어, 포인트, 라인, 삼각형 등) 의 버텍스들에, 및/또는 더 고차의 프리미티브 (예를 들어, 더 고차의 표면, 이를 테면 베지어 표면) 의 제어 포인트들에 대응할 수도 있다. 일부 경우들에서, 버텍스들 각각은 하나 이상의 버텍스들의 그룹들로 그룹화될 수도 있고, 버텍스들의 이들 그룹들 각각은 단일 프리미티브에 대응할 수도 있다.

[0078] 지오메트릭적으로 정의된 프리미티브의 형상은 일부 예들에서, 프리미티브 데이터 (40) 에 반드시 포함될 필요는 없는 추가적인 데이터에 의해 정의될 수도 있다. 추가적인 데이터는 하나 이상의 미리 결정된 프리미티브 타입들, 하나 이상의 수학적 함수들, 및/또는 하나 이상의 테셀레이션 기법들의 세트로부터 특정된 프리미티브 타입의 하나 이상을 포함할 수도 있다.

[0079] 일부 예들에서, 특정된 프리미티브 타입은 GPU (12) 에 렌더링 상태 변수로서 저장될 수도 있고 소프트웨어 애플리케이션 (24) 에 의해 구성가능할 수도 있다. 특정된 프리미티브 타입은 일부 경우들에서, 결과의 렌더링된 프리미티브들 (예를 들어, 포인트들, 라인들, 삼각형들 등) 의 형상 및/또는 프리미티브 데이터 (40) 에 포함된 버텍스들의 접속성 (예를 들어, 삼각형 스트립, 삼각형 팬 등) 을 정의할 수도 있다. 일부 예들에서, 상이한 프리미티브 타입들은 프로세싱 유닛들 (34) 에 의해 구현된 그래픽스 파이프라인이 프로세싱 가능한 프리미티브 토폴로지들의 세트에 대응할 수도 있다. 추가 예들에서, 상이한 프리미티브 타입들은 그래픽스 API (26) 에 의해 정의되고 소프트웨어 애플리케이션 (24) 에 의한 이용을 위해 이용가능한 프리미티브 토폴로지들의 세트에 대응할 수도 있다.

[0080] 하나 이상의 수학적 함수들 및/또는 하나 이상의 테셀레이션 기법들은 GPU (12) 의 하나 이상의 셰이더 유닛들 (예를 들어, 헐 셰이더 유닛 및/또는 도메인 셰이더 유닛) 상에서 실행하도록 구성되는 하나 이상의 셰이더 프로그램들에서 특정될 수도 있다. 수학적 함수들은 커브드 라인들 및/또는 커브 표면들을 갖는 프리미티브들을 정의하는데 이용될 수도 있다. 하나 이상의 테셀레이션 기법들은 입력 프리미티브의 형상 및/또는 곡률을 근사화하는 복수의 테셀레이트된 프리미티브들에 의해 프리미티브를 정의하는데 이용될 수도 있다.

[0081] 소프트웨어 애플리케이션 (24) 으로부터 드로우 콜 커맨드를 수신하는 것에 응답하여, GPU 드라이버 (28) 는 GPU (12) 로 하여금, 렌더링될 복수의 프리미티브들 (예를 들어, 프리미티브 데이터 (40)) 에 기초하여 타일 기반 렌더링을 수행하게 할 수도 있다. 예를 들어, GPU 드라이버 (28) 는 GPU (12) 로 하여금, 복수의 렌더링 패스 반복들을 포함하는 렌더링 패스 및 비닝 패스를 수행하게 할 수도 있다. 비닝 패스 동안, GPU (12) 는 프리미티브들 각각이 이미지 데이터 (예를 들어, 픽셀 데이터) 를 렌더 타겟의 복수의 서브-영역들 중 어느 것에 컨트리뷰트 (contribute) 하는지를 결정하고, 프리미티브들 각각이 이미지 데이터 (예를 들어, 픽셀 데이터) 를 렌더 타겟의 복수의 서브-영역들 중 어느 것에 컨트리뷰트하는지를 나타내는 비닝 데이터를 생성할 수도 있다. 일단 비닝 데이터가 생성되었다면, GPU (12) 는 혼합, 래스터화된 버전의 프리미티브들을 생성하기 위해 비닝 데이터 및 프리미티브 데이터 (40) 에 기초하여 복수의 렌더링 패스 반복들을 포함하는 렌더링 패스를 수행할 수도 있다.

[0082] 일부 예들에서, 비닝 패스를 수행하기 위하여, GPU (12) 에서의 래스터라이저는 래스터화될 프리미티브들에 대해 저해상도 z-버퍼링 및/또는 은면 제거 (back-face culling) 를 수행하도록 구성될 수도 있다. 이러한 예들에서, 비닝 데이터는 z-버퍼링 및/또는 은면 제거 후에 보이는 프리미티브들에 기초하여 생성될 수도 있다.

[0083] 일부 경우들에서, 렌더링된 프리미티브들은 복수의 픽셀들로서 저장될 수도 있다. 픽셀들 각각은 렌더 타겟

의 하나 이상의 공간 위치들과 연관될 수도 있고, 각각의 픽셀의 컬러를 나타내는 하나 이상의 속성들을 포함할 수도 있다. 일부 경우들에서, 픽셀들 각각은 픽셀의 투명도를 나타내는 하나 이상의 속성들을 더 포함할 수도 있다. 일부 예들에서, 픽셀 데이터는 각각의 픽셀에 대한 적색, 녹색, 청색, 및 알파 (RGBA) 속성들을 포함할 수도 있으며, 여기서 "RGB" 컴포넌트들은 컬러 값들에 대응하고 "A" 컴포넌트는 알파 값 (즉, 투명도 또는 블렌딩 값) 에 대응한다.

[0084] 본 개시물에 설명된 기법들은 예를 들어, 소프트웨어 애플리케이션 (24), 그래픽스 API (26), GPU 드라이버 (28), 커맨드 엔진 (32) 및 프로세싱 유닛들 (34) 을 포함하는 도 2 에 도시된 컴포넌트들 중 임의의 것에서 구현될 수도 있다. 예를 들어, GPU 드라이버 (28), 커맨드 엔진 (32), 및/또는 프로세싱 유닛들 (34) 은 본 개시물에 설명된 기법들 중 임의의 것에 따라 하나 이상의 타임스탬프들 (예를 들어, 인트라-프레임 타임스탬프들) 을 생성하도록 구성될 수도 있다.

[0085] 일부 예들에서, GPU 드라이버 (28), 커맨드 엔진 (32), 및/또는 프로세싱 유닛들 (34) 은 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU (12) 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성하도록 구성될 수도 있다. 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다. 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용하는 것은, 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에 의해 인트라-프레임 타임스탬프 요청들이 지원되는 것을 허용할 수도 있다.

[0086] 동작 동안, CPU (6) 상에서 실행되는 소프트웨어 애플리케이션 (24) (예를 들어, 그래픽스 애플리케이션) 은 그래픽스 프레임을 렌더링하기 위해 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림) 을 생성할 수도 있다. 일부 경우들에서, 순서화된 시퀀스의 커맨드들은 복수의 드로우 콜 커맨드들 및 복수의 타임스탬프 요청들을 포함할 수도 있다. 타임스탬프 요청들 중 적어도 일부는 순서화된 시퀀스의 커맨드들에서의 상이한 드로우 콜 커맨드들 사이에 배치될 수도 있다. 소프트웨어 애플리케이션 (24) 은 그래픽스 API (26) 를 통해 순서화된 시퀀스의 커맨드들을 GPU 드라이버 (28) 에 제공할 수도 있다.

[0087] 타일 기반 렌더링 기법들을 이용하여 그 시퀀스의 커맨드들을 실행하기 위해, GPU 드라이버 (28) 는, 타임스탬프 요청들 각각에 대해, 각각의 타임스탬프 요청에 기초하여 복수의 빈당 타임스탬프 요청들을 생성할 수도 있다. GPU 드라이버 (28) 는 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 (예를 들어, 메모리 (10) 에서의 커맨드들 (38)) 중 각각의 커맨드 스트림에 배치할 수도 있다. 커맨드 스트림들 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 실행될 수도 있다. 커맨드 스트림들은 빈당 커맨드 스트림들로 지칭될 수도 있다. 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하도록 구성될 수도 있다.

[0088] GPU 드라이버 (28) 는 GPU (12) 로 하여금 빈당 커맨드 스트림들을 실행하게 할 수도 있다. 빈당 커맨드 스트림들을 실행하는 동안, GPU (12) (예를 들어, 커맨드 엔진 (32) 및/또는 프로세싱 유닛들 (34)) 는 GPU (12) 에 의해 수신된 빈당 커맨드 스트림들에서의 빈당 타임스탬프 요청들을 실행하는 것에 응답하여 빈당 타임스탬프 값들을 생성할 수도 있다. 일부 경우들에서, GPU (12) 는 빈당 커맨드 스트림들에서의 빈당 타임스탬프 요청들 각각에 대한 각각의 빈당 타임스탬프 값을 생성할 수도 있다. 빈당 타임스탬프 값들 각각은 각각의 빈당 타임스탬프 값과 연관된 빈당 타임스탬프 요청이 GPU (12) 에 의해 실행되었던 시간을 나타낼 수도 있다. 일부 예들에서, 빈당 타임스탬프 값들 각각은 GPU (12) 에 의해 생성된 각각의 빈당 타임스탬프에 포함될 수도 있다.

[0089] 일부 예들에서, GPU (12) 는 빈당 타임스탬프 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. 예를 들어, GPU (12) 는 GPU 드라이버 (28) 에 의해 액세스될 수도 있는 메모리 (10) 의 타임스탬프 데이터 (42) 에 빈당 타임스탬프 값들을 배치할 수도 있다. 빈당 타임스탬프 값들을 수신하는 것에 응답하여, GPU 드라이버 (28) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. GPU 드라이버 (28) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. GPU 드라이버 (28) 는 소프트웨어 애플리케이션 (24) 에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0090] 추가 예들에서, GPU (12) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프 값들을 생성하고, 애플리케이션 요청 타임스탬프 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. GPU (12) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. 일부 예들에서, GPU (12) 는 GPU 드라이버 (28) 에 의해 액세스될 수도 있는 메모리 (10) 의 타임스

탬프 데이터 (42) 에 애플리케이션 요청 타임스탬프 값들을 배치할 수도 있다. GPU 드라이버 (28) 는 소프트웨어 애플리케이션 (24) 에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0091] 추가적인 예들에서, GPU (12) 는 빈당 타임스탬프 값들에 기초하여 하나 이상의 중간 값들을 생성하고, 중간 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. GPU 드라이버 (28) 는 중간 값들에 기초하여 빈당 타임스탬프 값들을 생성할 수도 있다. GPU 드라이버 (28) 및 GPU (12) 는 애플리케이션 요청 타임스탬프 값들을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. GPU 드라이버 (28) 는 소프트웨어 애플리케이션 (24) 에 애플리케이션 요청 타임스탬프 값들을 제공할 수도 있다.

[0092] 일부 예들에서, CPU (6) 및/또는 GPU (12) 는 각각의 애플리케이션 요청 타임스탬프 값이 적어도 2 개의 상이한 빈당 타임스탬프 값들의 함수가 되도록 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. 이러한 예들에서, 적어도 2 개의 상이한 빈당 타임스탬프 값들은 일부 예들에서, 상이한 렌더링 패스 반복들 동안 생성될 수도 있다. 상이한 렌더링 패스 반복들 동안 생성되는 빈당 타임스탬프 값들을 이용하는 것은 그래픽스 프레임의 렌더링 동안 상이한 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 적어도 어느 정도까지는 반영하는 애플리케이션 요청 타임스탬프 값들을 그래픽스 프로세싱 시스템이 생성하는 것을 허용할 수도 있다. 이렇게 하여, 본 개시물에 설명된 기법들은 타일 기반 렌더링 시스템이 비연속적인 인터리빙된 방식으로 드로우 콜 커맨드들을 실행하는 경우들에도, 타일 기반 렌더링 시스템이 인트라-프레임 타임스탬프들을 지원하는 것을 허용할 수도 있다.

[0093] 타일 기반 렌더링은, 일부 예들에서, 렌더 타겟을 복수의 서브-영역들 (예를 들어, 빈들 또는 타일들) 로 서브분할하는 것, 및 렌더 타겟의 서브-영역들 각각에 대해 별개의 렌더링 패스 반복을 포함하는 렌더링 패스를 수행하는 것을 수반할 수도 있다. 렌더링 패스 동안 프로세싱되어야 하는 프리미티브들의 수를 감소시키기 위해, 비닝 패스는, 일부 예들에서, 렌더링 패스 전에 수행될 수도 있다. 비닝 패스는 렌더링될 프리미티브들 각각이 픽셀 데이터를 렌더 타겟의 복수의 서브-영역들 중 어느 것에 컨트리뷰트하는지를 나타내는 비닝 데이터를 생성하는데 이용될 수도 있다. 비닝 데이터는 특정 렌더링 패스 반복들 동안 활성인 렌더 타겟의 서브-영역들에 컨트리뷰트하는 프리미티브들을 선택적으로 렌더링하기 위해 렌더링 패스 반복들 동안 이용되어, 렌더링 패스 동안 프로세싱되어야 하는 프리미티브들의 수를 감소시킬 수도 있다.

[0094] 렌더링은 그래픽스 장면에서의 3D 오브젝트들에 대응하는 3D 그래픽스 프리미티브들을 2D 래스터화된 이미지 데이터로 변환하는 프로세스를 지칭할 수도 있다. 렌더링은 통상, 장면에서의 그래픽스 프리미티브들 각각이 렌더링될 때 보통 업데이트되는 렌더 타겟 (예를 들어, 프레임 버퍼) 에 대하여 일어난다. 따라서, 렌더 타겟은 그래픽스 장면에 대한 최종 2D 래스터화된 이미지 데이터를 저장할 뿐만 아니라, 렌더 타겟은 그래픽스 장면이 렌더링될 때 중간 데이터를 저장할 수도 있다. 렌더 타겟에 저장된 2D 래스터화된 이미지 데이터는 픽셀들 각각이 컬러 데이터, 투명도 데이터, 및/또는 깊이 데이터를 포함하는 복수의 픽셀들을 포함할 수도 있다. 각각의 새로운 프리미티브가 렌더 타겟으로 렌더링됨에 따라, 새로운 프리미티브의 2D 래스터화된 이미지 데이터는 이전에 렌더링된 프리미티브들에 대한 렌더 타겟에 이미 저장되는 기존의 중간 데이터와 병합된다.

[0095] 렌더 타겟에서의 데이터를 병합하기 위해, 중간 데이터는 통상 새로운 데이터를 렌더 타겟에 기입하기 전에 렌더 타겟으로부터 판독되어야 한다. 따라서, 렌더링은 렌더 타겟을 포함하는 메모리에 대하여 다수의 판독 및 기입 동작들의 수행을 수반하여, 높은 메모리 대역폭 사용량을 초래할 수도 있다. 높은 메모리 대역폭 사용량 때문에, 렌더 타겟에 대해 전용된 높은 대역폭, 온-칩 메모리를 이용하는 것이 바람직하다. 그러나, 면적 제한된 애플리케이션들, 이를 테면, 예를 들어, 모바일 애플리케이션들에는, 렌더 타겟에서의 픽셀들 각각에 대한 데이터 전부를 동시에 보유가능한 높은 대역폭, 온-칩 메모리를 구현할 정도로 충분한 이용가능한 공간이 없을 수도 있다.

[0096] 타일 기반 렌더링은 렌더 타겟을 복수의 서브-영역들 (예를 들어, 타일들 또는 빈들) 로 서브분할하는 것, 및 렌더 타겟의 서브-영역들 각각에 대해 별개의 렌더링 패스 반복을 포함하는 렌더링 패스를 수행하는 것에 의해 상기 언급된 이슈들을 다룰 수도 있다. 서브-영역들 각각은 렌더 타겟에서의 픽셀들의 서브세트 (예를 들어, 픽셀들의 16x16 타일) 에 대응할 수도 있다. 렌더 타겟의 서브-영역들은 대안적으로는 타일들 또는 빈들로 지칭될 수도 있다. 렌더링 패스 반복들 각각 동안, 대응하는 서브-영역과 연관된 이미지 데이터 전부가 렌더링될 수도 있으며, 이는 픽셀 데이터를 서브-영역에 컨트리뷰트하는 프리미티브들 각각을 렌더링하는 것을 포함할 수도 있다. 렌더 타겟의 단일 서브-영역에 대한 데이터를 저장할 정도로 큰 높은 대역폭, 온-칩 메모리는 렌더링 패스 반복들 각각에 대한 로컬 렌더 타겟으로서 이용될 수도 있고, 렌더링 패스 반복이 완료한 후, 렌더링 패스 반복에 대한 로컬 렌더 타겟의 콘텐츠들은 낮은 대역폭, 오프-칩 시스템 메모리에 저장된

일반 렌더 타겟으로 전송될 수도 있다. 타일 단위 기반으로 별개의 렌더링 패스 반복들을 수행함으로써, 타일 기반 렌더링 스킴들은 큰 온-칩 메모리들을 허용하지 않는 면적 제한된 애플리케이션들에서도 래스터화된 이미지 데이터를 병합하기 위해 높은 대역폭, 온-칩 메모리가 이용되는 것을 허용가능할 수도 있다.

[0097] 타일 기반 렌더링을 수행하기 위한 하나의 접근법은 렌더 타겟의 서브-영역들 각각에 대해 렌더링 패스 반복을 수행하고, 렌더링 패스 반복들 각각 동안, 현재 렌더링 중인 특정 서브-영역에 대한 출력을 제한하기 위해 상이한 시저스 세팅들을 이용하는 동안 장면에서의 프리미티브들 전부를 렌더링하는 것이다. 그러나, 이러한 접근법은 프리미티브가 렌더링된 서브-영역에서 실제로 보이는지 여부와 관계없이 프리미티브들 각각이 렌더링 패스 반복들 각각에서 렌더링되기 때문에 비효율적일 수도 있다.

[0098] 타일 기반 렌더링의 효율을 개선하기 위하여, 비닝 패스는 일부 예들에서, 렌더링 패스의 수행 전에 수행될 수도 있다. 비닝 패스는 프리미티브들에 대한 비닝 데이터를 결정하는데 이용될 수도 있다. 렌더링될 프리미티브들 각각에 대해, 비닝 데이터는 프리미티브들 각각이 픽셀 데이터를 렌더 타겟의 서브-영역들 중 어느 것에 컨트리뷰트하는지에 대하여 나타낼 수도 있다.

[0099] 도 3 은 복수의 서브-영역들로 서브분할되는 예시적인 렌더 타겟 (50) 을 예시하는 개념적 다이어그램이다. 도 3 은 또한, 서브분할된 렌더 타겟 (50) 상에 디스플레이된 복수의 프리미티브들을 예시한다. 일부 경우들에서, 렌더 타겟 (50) 은 프레임 버퍼에 대응할 수도 있다. 도 3 에 도시한 바와 같이, 렌더 타겟 (50) 은 1 내지 20 으로 넘버링되는 복수의 비중첩 서브-영역들 (대안적으로는 빈들 또는 타일들로 지칭됨) 로 분할된다. 서브-영역들 각각은 렌더 타겟 (50) 에서의 픽셀들의 서브세트 (예를 들어, 픽셀들의 16x16 타일 등) 에 대응할 수도 있다. 도 3 에 도시된 예시적인 프리미티브에 대해, 비닝 데이터는 프리미티브 A 가 픽셀 데이터를 타일들 (1, 2, 6, 7, 8 및 12) 에 컨트리뷰트하고, 프리미티브 B 가 픽셀 데이터를 타일들 (7, 8, 12 및 13) 에 컨트리뷰트하며, 등등인 것을 나타낼 수도 있다.

[0100] 일부 예들에서, 비닝 데이터는 렌더 타겟에 렌더링될 프리미티브들 각각의 래스터화된 버전들의 합성에 기초하여 생성될 수도 있다. 일부 경우들에서, 보존 z-테스팅 및/또는 다른 제거 기법들이 프리미티브들 각각의 래스터화된 버전들을 생성하는데 이용될 수도 있다. 보존 z-테스팅 및/또는 다른 제거 기법들은 특정 타일에 컨트리뷰트한다고 하는 프리미티브들의 리스트에 포함되는 것으로부터, 가려진 프리미티브들 (즉, 다른 프리미티브들 뒤에 위치하는 프리미티브들) 을 제거할 수도 있다.

[0101] 특정 서브-영역 (예를 들어, 타일 또는 빈) 에 대한 렌더링 패스 반복 동안, 비닝 데이터는 실제로 이미지 데이터 (예를 들어, 픽셀 데이터) 를 서브-영역에 컨트리뷰트하는 렌더링될 프리미티브들을 선택하고, 이미지 데이터를 서브-영역에 컨트리뷰트하지 않는 렌더링 프리미티브들을 바이패스하는데 이용될 수도 있다. 이렇게 하여, 주어진 렌더링 패스 반복 동안 프로세싱되어야 하는 프리미티브들의 수는 일부 경우들에서 감소될 수도 있다.

[0102] 일부 예들에서, GPU (12) 는 비닝 패스로부터 생성된 비닝 데이터에 기초하여 렌더 타겟의 서브-영역들 각각에 대해 렌더링 패스 반복을 수행할 수도 있다. 예를 들어, 복수의 렌더링 패스 반복들 각각에 대해, GPU (12) 는 비닝 데이터에 기초하여 각각의 렌더링 패스 반복 동안 하나 이상의 드로우 콜들과 연관된 복수의 프리미티브들을 렌더링할지 여부를 결정할 수도 있다. 비닝 데이터가 프리미티브가 픽셀 데이터를 각각의 렌더링 패스 반복과 연관된 서브-영역에 컨트리뷰트한다는 것을 나타낸다면, GPU (12) 는 각각의 렌더링 패스 반복과 연관된 서브-영역으로 렌더링 패스 반복 동안 프리미티브를 렌더링할 수도 있다. 한편, 비닝이 프리미티브가 픽셀 데이터를 각각의 렌더링 패스 반복과 연관된 서브-영역에 컨트리뷰트하지 않는다는 것을 나타낸다면, GPU (12) 는 각각의 렌더링 패스 반복과 연관된 서브-영역으로 프리미티브를 렌더링하지 않을 수도 있다.

[0103] 도 3 에 도시된 서브-영역들은 실질적으로 동일한 사이즈 및 형상이지만, 다른 예들에서, 서브-영역들은 상이한 사이즈들 및/또는 상이한 형상들을 가질 수도 있다. 또한, 서브-영역들의 사이즈 및 형상은 GPU 의 제조 시에 또는 렌더링 시에 실질적으로 고정될 필요는 없고, 일부 예들에서는, GPU (12) 의 동작 동안 동적으로 조정될 수도 있다.

[0104] 도 4 는 본 개시물에 따른 그래픽스 애플리케이션에 의해 이슈된 예시적인 커맨드 스트림 (52) 을 예시하는 개념적 다이어그램이다. 일부 예들에서, 커맨드 스트림 (52) 은 도 2 에 도시된 소프트웨어 애플리케이션 (24) 에 의해 이슈될 수도 있다. 추가 예들에서, 커맨드 스트림 (52) 은 도 2 에 도시된 GPU 드라이버 (28) 에 의해 수신 및/또는 프로세싱될 수도 있다.

[0105] 커맨드 스트림 (52) 은 순서화된 시퀀스의 커맨드들 (즉, TSR1, DRAW1, TSR2, DRAW2, TSR3, DRAW3, TSR4) 을

포함한다. 도 4 에서, TSR 은 "타임스탬프 요청" 을 의미하고 DRAW 는 "드로우 콜 커맨드" 를 의미한다. 이로써, 도 4 에 예시된 커맨드들의 시퀀스는 제 1 타임스탬프 요청 (TSR1), 다음에 제 1 드로우 콜 커맨드 (DRAW1), 다음에 제 2 타임스탬프 요청 (TSR2), 다음에 제 2 드로우 콜 커맨드 (DRAW2), 다음에 제 3 타임스탬프 요청 (TSR3), 다음에 제 3 드로우 콜 커맨드 (DRAW3), 다음에 제 4 타임스탬프 요청 (TSR4), 다음에 제 4 드로우 콜 커맨드 (DRAW4) 를 포함한다.

[0106] 커맨드 스트림 (52) 에서의 커맨드들 각각은 단일 그래픽스 프레임의 렌더링과 연관될 수도 있다. 드로우 콜 커맨드들 각각은 동일한 그래픽스 프레임의 부분으로서 렌더링될 하나 이상의 프리미티브들을 특정할 수도 있다. 일부 예들에서, 상이한 드로우 콜 커맨드들은 그래픽스 프레임에서의 프리미티브들을 렌더링하는데 이용되는 상이한 프리미티브 타임들과 및/또는 그래픽스 프레임에서의 프리미티브들을 렌더링하는데 이용되는 상이한 렌더링 상태들과 연관될 수도 있다.

[0107] 도 4 에 도시한 바와 같이, 타임스탬프 요청은 드로우 콜 커맨드들 각각 사이에 배치된다. 도 4 에 추가 도시한 바와 같이, 드로우 콜 커맨드는 타임스탬프 요청들 각각 사이에 배치된다. TSR2 및 TSR3 양자는 이러한 타임스탬프 요청들이 2 개의 상이한 드로우 콜 커맨드들 사이에 각각 포지셔닝되기 때문에 인트라-프레임 타임스탬프 요청들에 대응할 수도 있다.

[0108] 도 5 는 본 개시물에 따른 렌더링 패스를 수행하기 위한 예시적인 실행 타임라인 (54) 을 예시하는 개념적 다이어그램이다. 실행 타임라인 (54) 은 왼쪽으로부터 오른쪽으로 시간이 증가한다.

[0109] 예시적인 실행 타임라인 (54) 은 렌더 타겟이 4 개의 빈들 또는 서브-영역들 (즉, A, B, C 및 D) 로 서브분할되는 경우에 렌더링 패스의 부분으로서 수행되는 렌더링 패스 반복들의 시퀀스를 나타낸다. 렌더링 패스 반복들 각각은 렌더 타겟의 특정 빈에 대하여 수행된다. 예를 들어, "렌더링 패스 반복 A" 는 렌더 타겟의 제 1 빈 (즉, 빈 "A") 에 대하여 수행되고, "렌더링 패스 반복 B" 는 렌더 타겟의 제 2 빈 (즉, 빈 "B") 에 대하여 수행되며, 등등이다.

[0110] 일부 예들에서, 도 1 에 도시된 컴퓨팅 시스템 (2) 은 실행 타임라인 (54) 에서 나타내진 렌더링 패스 반복들을 수행할 수도 있다. 추가 예들에서, GPU (12) 는 실행 타임라인 (54) 에서 나타내진 렌더링 패스 반복들을 수행할 수도 있고, CPU (6) 는 GPU (12) 로 하여금 실행 타임라인 (54) 에서 나타내진 렌더링 패스 반복들을 수행하게 할 수도 있다.

[0111] 도 5 에 도시한 바와 같이, 컴퓨팅 시스템 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 은 렌더 타겟에 포함된 빈들 각각에 대해 렌더링 패스 반복을 수행할 수도 있다. 예를 들어, 컴퓨팅 시스템 (2) (예를 들어, CPU (6) 및/또는 GPU (12)) 은 제 1 빈에 대하여 제 1 렌더링 패스 반복 (렌더링 패스 반복 A) 을 수행하고, 다음에 제 2 빈에 대하여 제 2 렌더링 패스 반복 (렌더링 패스 반복 B) 을 수행하고, 다음에 제 3 빈에 대하여 제 3 렌더링 패스 반복 (렌더링 패스 반복 C) 을 수행하며, 다음에 제 4 빈에 대하여 제 4 렌더링 패스 반복 (렌더링 패스 반복 D) 을 수행할 수도 있다.

[0112] 도 6 은 본 개시물에 따른 복수의 렌더링 패스 반복들을 수행하기 위한 예시적인 커맨드 스트림들 (56, 58, 60, 62) 을 예시하는 개념적 다이어그램이다. 일부 예들에서, 커맨드 스트림들 (56, 58, 60, 62) 은 도 2 에 도시된 GPU 드라이버 (28) 에 의해 이슈 및/또는 생성될 수도 있다. 추가 예들에서, 커맨드 스트림들 (56, 58, 60, 62) 은 도 2 에 도시된 GPU (12) 및/또는 커맨드 엔진 (32) 에 의해 수신 및/또는 프로세싱될 수도 있다.

[0113] 도 6 에서, TSR 은 "타임스탬프 요청" 을 의미하고, DRAW 는 "빈당 드로우 콜" 을 의미한다. TSRGPU 타임스탬프 요청들은 GPU 드라이버 (28) 에 의해 생성되고 각각의 렌더링 패스 반복들의 처음에 각각의 커맨드 스트림들 (56, 58, 60, 62) 에 배치되는 타임스탬프 요청들에 대응한다. 일부 예들에서, 도 6 에서의 TSR들 및 TSRGPU들은 대안적으로는 이러한 타임스탬프 요청들을 도 4 의 커맨드 스트림 (52) 에 포함된 타임스탬프 요청들과 구별하기 위해 빈당 타임스탬프 요청들로 지칭될 수도 있다. 빈당 타임스탬프 요청들에 응답하여 생성되는 타임스탬프들 및 타임스탬프 값들은 각각 빈당 타임스탬프들 및 빈당 타임스탬프 값들로 지칭될 수도 있다.

[0114] 유사하게, 도 4 의 커맨드 스트림 (52) 에 포함된 타임스탬프 요청들은 이러한 타임스탬프 요청들을 GPU 드라이버 (28) 에 의해 생성되는 빈당 타임스탬프 요청들과 구별하기 위해 애플리케이션 생성 타임스탬프 요청들로 지칭될 수도 있다. 애플리케이션 생성 타임스탬프 요청들에 응답하여 생성되는 타임스탬프들 및 타임스탬프 값들은 각각 애플리케이션 요청 타임스탬프들 및 애플리케이션 요청 타임스탬프 값들로 지칭될 수도 있다.

[0115] 일부 예들에서, 빈당 타임스탬프 요청들은 커맨드 스트림 (52) 에서의 소프트웨어 애플리케이션 (24) 으로부터 애플리케이션 생성 타임스탬프 요청들을 수신하는 것에 응답하여 GPU 드라이버 (28) 에 의해 생성될 수도 있다.

예를 들어, GPU 드라이버 (28) 는 커맨드 스트림 (52) 에서의 애플리케이션 생성 타임스탬프 요청 (TSR1) 을 조우하는 것에 응답하여 빈당 타임스탬프 요청들 (TSR1A, TSR1B, TSR1C 및 TSR1D) 을 생성할 수도 있다. 다른 예로서, GPU 드라이버 (28) 는 커맨드 스트림 (52) 에서의 애플리케이션 생성 타임스탬프 요청 (TSR3) 을 조우하는 것에 응답하여 빈당 타임스탬프 요청들 (TSR3A, TSR3B, TSR3C 및 TSR3D) 을 생성할 수도 있다.

[0116] 일부 예들에서, TSRGPU 타임스탬프 요청들은 커맨드 스트림 (52) 에서의 애플리케이션 생성 타임스탬프 요청들을 조우하는 것에 기초하여 또는 그것에 응답하여 생성되지 않을 수도 있다. 그 대신, 이러한 예들에서, GPU 드라이버 (28) 는 렌더링 패스 동안 수행될 각각의 렌더링 패스 인스턴스에 대한 TSRGPU 타임스탬프 요청들을 자동으로 생성할 수도 있다.

[0117] 빈당 타임스탬프 요청들 각각 다음에 오는 숫자는 빈당 타임스탬프 요청과 연관되는 애플리케이션 생성 타임스탬프 요청을 나타낸다. 빈당 타임스탬프 요청은 빈당 타임스탬프 요청이 애플리케이션 생성 커맨드 스트림 (예를 들어, 커맨드 스트림 (52)) 에서의 애플리케이션 생성 타임스탬프 요청을 조우하는 것에 기초하여 또는 그것에 응답하여 GPU 드라이버 (28) 에 의해 생성된다면 애플리케이션 생성 타임스탬프 요청과 연관될 수도 있다. 예를 들어, 빈당 타임스탬프 요청들 (TSR1A, TSR1B, TSR1C 및 TSR1D) 은 커맨드 스트림 (52) 에서의 애플리케이션 생성 타임스탬프 요청 (TSR1) 과 연관된다.

[0118] 빈당 타임스탬프 요청들 각각 다음에 오는 문자는 각각의 빈당 타임스탬프 요청과 연관되는 실행 타임라인 (54) 에서의 렌더링 패스 반복을 나타낸다. 타임스탬프 요청은 타임스탬프 요청이 렌더링 패스 반복 동안 이슈 및/또는 서비스된다면 렌더링 패스 반복과 연관될 수도 있다. 예를 들어, TSR1A 는 애플리케이션 생성 타임스탬프 요청 (TSR1) 과 연관되는 빈당 타임스탬프 요청 및 실행 타임라인 (54) 에서의 제 1 렌더링 패스 반복 (즉, 렌더링 패스 반복 A) 을 나타낸다.

[0119] 빈당 드로우 콜들 각각 다음에 오는 숫자는 각각의 빈당 드로우 콜과 연관되는 커맨드 스트림 (52) 에서의 드로우 콜 커맨드를 나타낸다. 빈당 드로우 콜들 각각 다음에 오는 문자는 각각의 빈당 드로우 콜과 연관되는 실행 타임라인 (54) 에서의 렌더링 패스 반복을 나타낸다.

[0120] 예를 들어, "DRAW1A" 는 커맨드 스트림 (52) 에서의 제 1 드로우 콜 커맨드 (즉, DRAW1) 및 실행 타임라인 (54) 에서의 제 1 렌더링 패스 반복 (즉, 렌더링 패스 반복 A) 과 연관되는 빈당 드로우 콜을 나타낸다. 다른 예로서, "DRAW2C" 는 커맨드 스트림 (52) 에서의 제 2 드로우 콜 커맨드 (즉, DRAW2) 및 실행 타임라인 (54) 에서의 제 3 렌더링 패스 반복 (즉, 렌더링 패스 반복 C) 과 연관되는 빈당 드로우 콜을 나타낸다.

[0121] 빈당 드로우 콜은 빈당 드로우 콜이 드로우 콜 커맨드를 실행하기 위하여 드로우 콜 커맨드에 기초하여 생성된다면 드로우 콜 커맨드와 연관될 수도 있다. 빈당 드로우 콜은 빈당 드로우 콜이 렌더링 패스 반복 동안 실행된다면 렌더링 패스 반복과 연관될 수도 있다.

[0122] 도 4 와 유사하게, 도 6 의 커맨드 스트림들 (56, 58, 60, 62) 각각은 순서화된 시퀀스의 커맨드들을 포함한다.

예를 들어, 커맨드 스트림 (56) 은 제 1 빈당 타임스탬프 요청 (TSRGPUA), 다음에 제 2 빈당 타임스탬프 요청 (TSR1A), 다음에 제 1 빈당 드로우 콜 (DRAW1A), 다음에 제 3 빈당 타임스탬프 요청 (TSR2A), 다음에 제 2 빈당 드로우 콜 (DRAW2A), 다음에 제 4 빈당 타임스탬프 요청 (TSR3A), 다음에 제 3 빈당 드로우 콜 (DRAW3A), 다음에 제 5 빈당 타임스탬프 요청 (TSR4A) 을 포함한다.

[0123] 커맨드 스트림들 (56, 58, 60, 62) 각각은 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 실행된다. 예를 들어, 커맨드 스트림 (56) 은 "렌더링 패스 반복 A" 동안 실행되고, 커맨드 스트림 (58) 은 "렌더링 패스 반복 B" 동안 실행되고, 커맨드 스트림 (60) 은 "렌더링 패스 반복 C" 동안 실행되며, 커맨드 스트림 (62) 은 "렌더링 패스 반복 D" 동안 실행된다.

[0124] 렌더링 패스 반복들 각각은 복수의 빈들 (예를 들어, 렌더 타겟의 서브-영역들) 중 각각의 빈을 렌더링하도록 구성된다. 예를 들어, 도 6 에 도시한 바와 같이, "렌더링 패스 반복 A" 는 렌더 타겟의 "빈 A" 를 렌더링하도록 구성되고, "렌더링 패스 반복 B" 는 렌더 타겟의 "빈 B" 를 렌더링하도록 구성되고, "렌더링 패스 반복 C" 는 렌더 타겟의 "빈 C" 를 렌더링하도록 구성되며, "렌더링 패스 반복 D" 는 렌더 타겟의 "빈 D" 를 렌더링하도록 구성된다.

[0125] 도 4 에 도시된 커맨드 스트림 (52) 을 수신하는 것에 응답하여, GPU 드라이버 (28) 는 커맨드 스트림 (52) 에

포함된 드로우 콜 커맨드들 각각에 대한 복수의 빈당 드로우 콜들 및 커맨드 스트림 (52) 에 포함된 타임스탬프 요청들 각각에 대한 복수의 빈당 타임스탬프 요청들을 생성할 수도 있다. 예를 들어, 커맨드 스트림 (52) 에서의 DRAW1 을 조우하는 것에 응답하여, GPU 드라이버 (28) 는 DRAW1A, DRAW1B, DRAW1C, 및 DRAW1D 를 생성할 수도 있다. 다른 예로서, 커맨드 스트림 (52) 에서의 TSR2 를 조우하는 것에 응답하여, GPU 드라이버 (28) 는 TSR2A, TSR2B, TSR2C, 및 TSR2D 를 생성할 수도 있다.

[0126] GPU 드라이버 (28) 는 또한 GPU 드라이버 (28) 에 의해 생성된 빈당 드로우 콜들 및 빈당 타임스탬프 요청들에 기초하여 복수의 렌더링 패스 반복 특정 커맨드 스트림들 (56, 58, 60, 62) 을 생성할 수도 있다. 예를 들어, GPU 드라이버 (28) 는 빈당 드로우 콜들 및 빈당 타임스탬프 요청들을 빈별로 그룹화하고, 빈당 드로우 콜들 및 빈당 타임스탬프 요청들의 각각의 그룹을 커맨드 스트림들 (56, 58, 60, 62) 중 별개의 커맨드 스트림에 배치할 수도 있다.

[0127] 일부 예들에서, GPU 드라이버 (28) 는 또한 도 4 의 커맨드 스트림 (52) 에서의 애플리케이션 생성 타임스탬프 요청과 연관되지 않는 커맨드 스트림들 (56, 58, 60, 62) 각각의 처음에 타임스탬프 요청들을 배치할 수도 있다. 이들 타임스탬프 요청들은 일부 예들에서, 애플리케이션 생성 타임스탬프 요청들에 대한 타임스탬프 값들을 생성할 때 참조 타임스탬프 요청들로서 기능할 수도 있다.

[0128] GPU 드라이버 (28) 는 GPU (12) 로 하여금, 커맨드 스트림들 (56, 58, 60, 62) 에 기초하여 복수의 렌더링 패스 반복들을 수행하게 할 수도 있다. 예를 들어, GPU 드라이버 (28) 는 GPU (12) 로 하여금, 별개의 렌더링 패스 반복 동안 커맨드 스트림들 (56, 58, 60, 62) 각각을 실행하게 할 수도 있다.

[0129] 일부 예들에서, 커맨드 스트림들 (56, 58, 60, 62) 을 실행하는 동안, GPU (12) 는 커맨드 스트림들 (56, 58, 60, 62) 에서의 빈당 타임스탬프 요청들 각각을 조우하는 것에 응답하여 CPU (6) (예를 들어, GPU 드라이버 (28)) 에 빈당 타임스탬프를 이슈할 수도 있다. GPU (12) 에 의해 이슈된 빈당 타임스탬프는 GPU (12) 가 커맨드 스트림들 (56, 58, 60, 62) 각각을 실행할 때 빈당 타임스탬프 요청을 조우한 시간에 대응하는 타임스탬프 값을 포함할 수도 있다. 이러한 예들에서, GPU (12) 로부터 빈당 타임스탬프들 및/또는 빈당 타임스탬프 값들을 수신하는 것에 응답하여, GPU 드라이버 (28) 는 GPU (12) 로부터 수신된 빈당 타임스탬프들 및/또는 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프들을 생성하고, 애플리케이션 요청 타임스탬프들을 소프트웨어 애플리케이션 (24) 에 제공할 수도 있다.

[0130] 추가 예들에서, 커맨드 스트림들 (56, 58, 60, 62) 을 실행하는 동안, GPU (12) 는 빈당 타임스탬프 요청들에 대응하는 빈당 타임스탬프 값들을 내부적으로 추적할 수도 있다. 이러한 예들에서, GPU (12) 는 일부 예들에서, GPU (12) 에 의해 내부적으로 추적한 빈당 타임스탬프 값들에 기초하여 하나 이상의 애플리케이션 요청 타임스탬프들 및/또는 하나 이상의 애플리케이션 요청 타임스탬프 값들을 생성하고, 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 수신하는 것에 응답하여, GPU 드라이버 (28) 는 애플리케이션 생성 타임스탬프를 생성하고 및/또는 그것을 소프트웨어 애플리케이션 (24) 에 제공할 수도 있다.

[0131] GPU (12) 가 빈당 타임스탬프 값들을 내부적으로 추적하는 추가적인 예들에서, GPU (12) 는 일부 예들에서, 하나 이상의 중간 시간 값들을 생성하고, 하나 이상의 중간 시간 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. 중간 시간 값들을 수신하는 것에 응답하여, GPU 드라이버 (28) 는 애플리케이션 생성 타임스탬프를 생성하고, 애플리케이션 생성 타임스탬프를 소프트웨어 애플리케이션 (24) 에 제공할 수도 있다.

[0132] 일부 예들에서, 도 4 에 도시된 커맨드 스트림 (52) 에 대해, 커맨드 스트림 (52) 에 포함된 타임스탬프 요청들 각각에 대한 GPU 드라이버 (28) 에 의해 반환된 타임스탬프들에 포함된 타임스탬프 값들은 표 1 에 열거된 수식들에 기초하여 생성될 수도 있다 :

표 1

TSV1 = TSVGPUA + (TSV1A – TSVGPUA) + (TSV1B – TSVGPUB) + (TSV1C – TSVGPUC) + (TSV1D – TSVGPUD)	(1)
TSV2 = TSVGPUA + (TSV2A – TSVGPUA) + (TSV2B – TSVGPUB) + (TSV2C – TSVGPUC) + (TSV2D – TSVGPUD)	(2)
TSV3 = TSVGPUA + (TSV3A – TSVGPUA) + (TSV3B – TSVGPUB) + (TSV3C – TSVGPUC) + (TSV3D – TSVGPUD)	(3)
TSV4 = TSVGPUA + (TSV4A – TSVGPUA) + (TSV4B – TSVGPUB) + (TSV4C – TSVGPUC) + (TSV4D – TSVGPUD)	(4)

타임스탬프 값들을 생성하기 위한 예시적인 수식들

[0133]

[0134] 표 1 에서, TSVx 는 타임스탬프 요청 (TSRx) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSVGPUx 는 타임스탬프 요청 (TSRGPUx) 에 응답하여 생성되는 타임스탬프 값이다. 예를 들어, TSV1 은 타임스탬프 요청 (TSR1) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSV3B 는 타임스탬프 요청 (TSR3B) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이며, 등등이다. 다른 예로서, TSVGPUA 는 타임스탬프 요청 (TSRGPUA) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSVGPUB 는 타임스탬프 요청 (TSRGPUB) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이며, 등등이다.

[0135] 추가 예들에서, 커맨드 스트림 (52) 에 포함된 타임스탬프 요청들 각각에 대한 GPU 드라이버 (28) 에 의해 반환된 타임스탬프들에 포함된 타임스탬프 값들은 다음의 일반 수식에 기초하여 생성될 수도 있으며 :

$$TSVx = TSVGPUA + (TSVxA - TSVGPUA) + (TSVxB - TSVGPUB) + (TSVxC - TSVGPUC) + (TSVxD - TSVGPUD) \quad (5)$$

[0136]

[0137] 여기서, TSVx 는 타임스탬프 요청 (TSRx) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSVxA 는 타임스탬프 요청 (TSRxA) 에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이며, 등등이다. 수식 (5) 에서, x 는 임의의 정수일 수도 있다.

[0138] 일부 예들에서, TSRxA, TSRxB, TSRxC, 및 TSRxD 는 커맨드 스트림 (52) 에서의 TSRx 를 수신하는 것에 응답하여 생성되는 타임스탬프 요청들일 수도 있고, TSRGPUA, TSRGPUB, TSRGPUC, 및 TSRGPUD 는 각각의 렌더링 패스 반복들의 처음에 GPU 드라이버 (28) 에 의해 생성되는 타임스탬프 요청들일 수도 있다.

[0139] 추가적인 예들에서, CPU (6) 및/또는 GPU (12) 는 다음의 수식에 기초하여 하나 이상의 타임스탬프 값들을 생성할 수도 있으며 :

$$Value = TSVGPU(0) + \sum_{y=0}^{N-1} \left(TSV(y) - TSVGPU(y)\right) \quad (6)$$

[0140]

[0141] 여기서 Value 는 그래픽스 애플리케이션으로부터 수신된 타임스탬프 요청에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSV(y) 는 제 y 렌더링 패스 반복 동안 생성되는 빈당 타임스탬프 값이고, TSVGPU(y) 는 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 생성되는 참조 타임스탬프 값이며, N 은 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수이다. 일부 예들에서, 수식 (6) 은 표 1 의 수식들의 일반화된 형태일 수도 있다.

[0142] 일부 경우들에서, TSV(y) 빈당 타임스탬프 값들 각각은 복수의 빈당 타임스탬프 요청들 중 각각의 빈당 타임스탬프 요청에 대응할 수도 있으며, 여기서 복수의 빈당 타임스탬프 요청들 각각은 Value 에 대응하는 그래픽스 애플리케이션으로부터 수신된 타임스탬프 요청에 대응하고 그것에 응답하여 생성된다.

[0143] 추가 예들에서, CPU (6) 및/또는 GPU (12) 는 다음의 수식에 기초하여 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에 포함된 복수의 타임스탬프 요청들에 대한 타임스탬프 값들을 생성할 수도 있으며 :

$$Value(x) = TSVGPU(0) + \sum_{y=0}^{N-1}\left(TSV(x,y) - TSVGPU(y)\right) \qquad (7)$$

[0144]

[0145] 여기서 Value(x) 는 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 제 x 타임스탬프 요청에 응답하여 생성되는 타임스탬프에 대한 타임스탬프 값이고, TSV(x,y) 는 제 y 렌더링 패스 반복 동안 생성되고 커맨드 스트림에서의 제 x 타임스탬프 요청에 대응하고 그것에 응답하여 생성되는 빈당 타임스탬프 값이고, TSVGPU(y) 는 제 y 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 생성되는 참조 타임스탬프 값이며, N 은 그래픽스 프레임을 렌더링하는데 이용된 렌더링 패스 반복들의 수이다. 일부 경우들에서, TSV(x,y) 빈당 타임스탬프 값들 각각은 복수의 빈당 타임스탬프 요청들 중 각각의 빈당 타임스탬프 요청에 대응할 수도 있으며, 여기서 복수의 빈당 타임스탬프 요청들 각각은 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 제 x 타임스탬프 요청에 대응하고 그것에 응답하여 생성된다. 일부 예들에서, 수식 (7) 은 표 1 의 수식들의 일반화된 형태일 수도 있다.

[0146] 일부 예들에서, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에 포함된 타임스탬프 요청들 각각에 대해, GPU 드라이버 (28) 는 빈당 타임스탬프 요청을 빈당 커맨드 스트림들 각각에 배치할 수도 있다. 이러한 예들에서, 빈당 커맨드 스트림들 각각에서의 빈당 타임스탬프 요청들의 순서는 일부 예들에서, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 대응하는 타임스탬프 요청들의 순서와 동일할 수도 있다.

[0147] 추가 예들에서, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림은 드로우 콜 커맨드들 및 타임스탬프 요청들을 포함할 수도 있다. 일부 경우들에서, 타임스탬프 요청들 중 적어도 하나는 커맨드 스트림에서의 드로우 콜 커맨드들 중 적어도 2 개의 드로우 콜 커맨드들 사이에 포지셔닝될 수도 있다. 이러한 예들에서, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 타임스탬프 요청들 각각에 대해, GPU 드라이버 (28) 는 빈당 타임스탬프 요청을 빈당 커맨드 스트림들 각각에 배치할 수도 있다. 또한, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 드로우 콜 커맨드들 각각에 대해, GPU 드라이버 (28) 는 빈당 드로우 콜을 빈당 커맨드 스트림들 각각에 배치할 수도 있다. 이러한 예들에서, 빈당 커맨드 스트림들 각각에서의 빈당 타임스탬프 요청들 및 빈당 드로우 콜들의 순서는 일부 예들에서, 그래픽스 애플리케이션으로부터 수신된 커맨드 스트림에서의 대응하는 타임스탬프 요청들 및 드로우 콜 커맨드들의 순서와 동일할 수도 있다.

[0148] 일반적으로, GPU 드라이버 (28) 및 GPU (12) 중 하나 또는 양자는 빈당 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 애플리케이션 요청 타임스탬프 값들을 생성할 수도 있다. 일부 예들에서, GPU 드라이버 (28) 는 GPU (12) 로부터 빈당 타임스탬프 값들을 수신하고, 빈당 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 생성할 수도 있다. 이러한 예들에서, GPU 드라이버 (28) 는 일부 예들에서, 수식 (1) 내지 (7) 의 하나 이상에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 생성할 수도 있다.

[0149] 추가 예들에서, GPU (12) 는 빈당 타임스탬프 값들을 생성하고, 빈당 타임스탬프 값들에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 생성하며, 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. 이러한 예들에서, GPU (12) 는 일부 예들에서, 수식 (1) 내지 수식 (7) 의 하나 이상에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 생성할 수도 있다.

[0150] 추가적인 예들에서, GPU (12) 는 빈당 타임스탬프 값들을 생성하고, 빈당 타임스탬프 값들에 기초하여 하나 이상의 중간 값들을 생성하며, 중간 값들을 GPU 드라이버 (28) 에 제공할 수도 있다. 이러한 예들에서, GPU 드라이버 (28) 는 GPU (12) 로부터 중간 값들을 수신하고, 중간 값들에 기초하여 애플리케이션 요청 타임스탬프들 및/또는 타임스탬프 값들을 생성할 수도 있다.

[0151] 일부 경우들에서, 중간 값들은 수식 (1) 내지 수식 (7) 에서의 하나 이상의 항들에 대응할 수도 있다. 추가적인 경우들에서, 중간 값들은 수식 (1) 내지 수식 (7) 에서 특정된 입력 변수들의 임의의 조합에 대응할 수도 있다.

[0152] 도 4 내지 도 6 에 도시한 바와 같이, CPU (6) 및/또는 GPU (12) 는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU (12) 에 의해 생성되는 복수의 빈당 타임스탬프 값들 (예를 들어, TSR2A, TSR2B, TSR2C,

및 TSR2D 에 대응하는 빈당 타임스탬프 값들) 에 기초하여 시점을 나타내는 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 을 생성할 수도 있다. 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들 (예를 들어, TSR2A, TSR2B, TSR2C, 및 TSR2D 에 대응하는 빈당 타임스탬프 값들) 의 함수일 수도 있다.

[0153] 일부 예들에서, 복수의 빈당 타임스탬프 값들 (예를 들어, TSR2A, TSR2B, TSR2C, 및 TSR2D 에 대응하는 빈당 타임스탬프 값들) 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 생성될 수도 있다. 예를 들어, TSR2A 에 대한 빈당 타임스탬프 값은 렌더링 패스 반복 A 동안 GPU (12) 에 의해 생성되고, TSR2B 에 대한 빈당 타임스탬프 값은 렌더링 패스 반복 B 동안 GPU (12) 에 의해 생성되고, TSR2C 에 대한 빈당 타임스탬프 값은 렌더링 패스 반복 C 동안 GPU (12) 에 의해 생성되며, TSR2D 에 대한 빈당 타임스탬프 값은 렌더링 패스 반복 D 동안 GPU (12) 에 의해 생성된다.

[0154] 렌더링 패스 반복들 각각은 일부 예들에서, 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하도록 구성될 수도 있다. 예를 들어, 렌더링 패스 반복 A 는 렌더 타겟의 빈 A 를 렌더링하도록 구성되고, 렌더링 패스 반복 B 는 렌더 타겟의 빈 B 를 렌더링하도록 구성되고, 렌더링 패스 반복 C 는 렌더 타겟의 빈 C 를 렌더링하도록 구성되며, 렌더링 패스 반복 D 는 렌더 타겟의 빈 D 를 렌더링하도록 구성된다.

[0155] 일부 예들에서, 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 을 생성하는데 이용되는 적어도 2 개의 빈당 타임스탬프 값들은 제 1 빈당 타임스탬프 값 (예를 들어, TSR2A 에 대응하는 타임스탬프 값) 및 제 2 빈당 타임스탬프 값 (예를 들어, TSR2B 에 대응하는 타임스탬프 값) 을 포함할 수도 있다. 제 1 빈당 타임스탬프 값 (예를 들어, TSR2A 에 대응하는 타임스탬프 값) 은 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 A) 동안 GPU (12) 에 의해 생성될 수도 있다. 제 2 빈당 타임스탬프 값 (예를 들어, TSR2B 에 대응하는 타임스탬프 값) 은 복수의 렌더링 패스 반복들 중 제 2 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 B) 동안 GPU (12) 에 의해 생성될 수도 있다. 제 2 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 B) 은 제 1 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 A) 과는 상이할 수도 있다.

[0156] 추가 예들에서, 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 을 생성하는데 이용되는 적어도 2 개의 빈당 타임스탬프 값들은 적어도 2 개의 참조 타임스탬프 값들 (예를 들어, TSRGPUA, TSRGPUB, TSRGPUC, TSRGPUD 에 대응하는 타임스탬프 값들) 을 더 포함할 수도 있다. 적어도 2 개의 참조 타임스탬프 값들 각각은 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 및 렌더링 패스 반복들 중 각각의 렌더링 패스 반복에 대한 임의의 프리미티브들을 렌더링하기 전에 생성될 수도 있다. 예를 들어, TSRGPUA 에 대응하는 타임스탬프 값은 렌더링 패스 반복 A 동안 및 렌더링 패스 반복 A 에 대한 임의의 프리미티브들을 렌더링하기 전에 (예를 들어, DRAW1A, DRAW2A, 및 DRAW3A 를 실행하기 전에) 생성된다.

[0157] 추가적인 예들에서, CPU (6) 및/또는 GPU (12) 는 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들 (커맨드 스트림 (52)) 에서의 적어도 2 개의 드로우 콜 커맨드들 (예를 들어, DRAW1, DRAW2) 사이에 포지셔닝되는 타임스탬프 요청 (예를 들어, TSR2) 에 응답하여 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 을 생성할 수도 있다. 일부 예들에서, TSR2 에 대한 타임스탬프 값은 그래픽스 프레임의 렌더링 동안 순서화된 시퀀스의 커맨드들에서의 드로우 콜 커맨드들 (예를 들어, DRAW1, DRAW2) 이 실행되는데 걸리는 상대적 시간량들에 기초하여 생성될 수도 있다.

[0158] 일부 예들에서, GPU (12) 는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 복수의 렌더링 패스 반복들을 수행할 수도 있다. 렌더링 패스 반복들 각각은 적어도 2 개의 빈당 드로우 콜들을 수행하도록 구성될 수도 있다. 예를 들어, 렌더링 패스 반복 A 는 DRAW1A 및 DRAW2A 를 수행하도록 구성될 수도 있다. 적어도 2 개의 빈당 드로우 콜들 각각은 적어도 2 개의 드로우 콜 커맨드들 중 각각의 드로우 콜 커맨드와 연관될 수도 있다. 예를 들어, DRAW1A 는 커맨드 스트림 (52) 에서의 DRAW1 과 연관될 수도 있고, DRAW2A 는 커맨드 스트림 (52) 에서의 DRAW2 와 연관될 수도 있다.

[0159] 이러한 예들에서, 타임스탬프 값 (예를 들어, TSR2 에 대응하는 타임스탬프 값) 을 생성하는데 이용되는 적어도 2 개의 빈당 타임스탬프 값들은 제 1 빈당 타임스탬프 값 (예를 들어 TSR2A 에 대응하는 타임스탬프 값) 및 제 2 빈당 타임스탬프 값 (예를 들어, TSR2B 에 대응하는 타임스탬프 값) 을 포함할 수도 있다. 제 1 빈당 타임스탬프 값 (예를 들어, TSR2A 에 대응하는 타임스탬프 값) 은 복수의 렌더링 패스 반복들 중 제 1 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 A) 동안 적어도 2 개의 빈당 드로우 콜들 (예를 들어, DRAW1A, DRAW2A) 의 수행 사이에 일어나는 시점을 나타낼 수도 있다. 제 2 빈당 타임스탬프 값 (예를 들어, TSR2B 에 대응하는 타임스탬프 값) 은 복수의 렌더링 패스 반복들 중 제 2 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 B)

동안 적어도 2 개의 빈당 드로우 콜들 (예를 들어, DRAW1B, DRAW2B) 의 수행 사이에 일어나는 시점을 나타낼 수도 있다. 제 2 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 B) 은 제 1 렌더링 패스 반복 (예를 들어, 렌더링 패스 반복 A) 과는 상이할 수도 있다.

[0160] 추가 예들에서, 복수의 빈당 타임스탬프 값들 (예를 들어, TSR2A. TSR2B, TSR2C, 및 TSR2D 에 대응하는 빈당 타임스탬프 값들) 각각은 복수의 빈당 타임스탬프 요청들 (예를 들어, 각각 TSR2A, TSR2B, TSR2C, 및 TSR2D) 중 각각의 빈당 타임스탬프 요청에 응답하여 생성될 수도 있다. 이러한 예들에서, 빈당 타임스탬프 요청들 (예를 들어, TSR2A, TSR2B, TSR2C, 및 TSR2D) 각각은 복수의 커맨드 스트림들 (예를 들어, 각각 커맨드 스트림들 (56, 58, 60, 62)) 중 각각의 커맨드 스트림에 배치될 수도 있다. 커맨드 스트림들 (예를 들어, 각각 커맨드 스트림들 (56, 58, 60, 62)) 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 (예를 들어, 각각 렌더링 패스 반복들 (A, B, C, 및 D)) 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 실행될 수도 있다. 렌더링 패스 반복들 (예를 들어, 렌더링 패스 반복들 (A, B, C, 및 D)) 각각은 렌더 타겟의 복수의 서브-영역들 (예를 들어, 각각 빈 (A, B, C, 및 D)) 중 각각의 서브-영역을 렌더링하도록 구성될 수도 있다.

[0161] 일부 예들에서, CPU (6) 및/또는 GPU (12) 는 복수의 빈당 타임스탬프 값들 (예를 들어, TSR1A 내지 TSR1D, TSR2A 내지 TSR2D, TSR3A 내지 TSR3D, TSR4A 내지 TSR4D) 에 기초하여 복수의 타임스탬프 값들 (예를 들어, TSR1, TSR2, TSR3, TSR4 에 대응하는 타임스탬프 값들) 을 생성할 수도 있다. 타임스탬프 값들 (예를 들어, TSR1, TSR2, TSR3, TSR4 에 대응하는 타임스탬프 값들) 각각은 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림 (52)) 에 포함된 복수의 타임스탬프 요청들 (예를 들어, 각각 TSR1, TSR2, TSR3, TSR4) 중 각각의 타임스탬프 요청에 대응할 수도 있다. 타임스탬프 요청들 각각은 그래픽스 애플리케이션에 의해 요청될 수도 있다.

[0162] 이러한 예들에서, 적어도 2 개의 타임스탬프 요청들 (예를 들어, TSR2 및 TSR3) 은 일부 예들에서, 그래픽스 프레임에 대해 실행될 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림 (52)) 에서의 연이은 드로우 콜 커맨드들의 각각의 쌍들 (예를 들어, 각각 DRAW1/DRAW2, DRAW2/DRAW3) 사이에 포지셔닝될 수도 있다. 이러한 예들에서, CPU (6) 및/또는 GPU (12) 는 순서화된 시퀀스의 커맨드들에서의 타임스탬프 요청들에 대해 반환된 타임스탬프 값들이 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림 (52)) 의 시작에서 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림 (52)) 의 끝까지 값이 단조 증가하도록 빈당 타임스탬프 값들에 기초하여 복수의 타임스탬프 값들 (예를 들어, TSR1, TSR2, TSR3, TSR4 에 대응하는 타임스탬프 값들) 을 생성할 수도 있다. 예를 들어, TSR1 은 TSR2 이하일 수도 있고, TSR2 는 TSR3 이하일 수도 있으며, TSR3 은 TSR4 이하일 수도 있다.

[0163] 이러한 예들에서, 복수의 타임스탬프 값들 (예를 들어, TSR1, TSR2, TSR3, TSR4 에 대응하는 타임스탬프 값들) 은 일부 예들에서, 그래픽스 프레임의 렌더링 동안 순서화된 시퀀스의 커맨드들 (예를 들어, 커맨드 스트림 (52)) 에서의 드로우 콜 커맨드들 (예를 들어, DRAW1, DRAW2, DRAW3) 이 실행되는데 걸리는 상대적 시간량들을 나타낼 수도 있다. 예를 들어, 그래픽스 프레임의 렌더링 동안 DRAW1 이 실행되는데 걸리는 시간량이 그래픽스 프레임의 렌더링 동안 DRAW2 가 실행되는데 걸리는 시간량보다 크다면, TSR2 와 TSR1 사이의 차이는 이러한 예들에서는, TSR3 과 TSR2 사이의 차이보다 클 수도 있다.

[0164] 일부 경우들에서, 그래픽스 프레임의 렌더링 동안 DRAW1 이 실행되는데 걸리는 시간량은 그래픽스 프레임의 렌더링 동안 DRAW1 과 연관된 빈당 드로우 콜 커맨드들 각각이 실행되는데 걸리는 총 시간량에 대응할 수도 있고, 그래픽스 프레임의 렌더링 동안 DRAW2 가 실행되는데 걸리는 시간량은 그래픽스 프레임의 렌더링 동안 DRAW2 와 연관된 빈당 드로우 콜 커맨드들 각각이 실행되는데 걸리는 총 시간량에 대응할 수도 있다. 예를 들어, 그래픽스 프레임의 렌더링 동안 DRAW1 이 실행되는데 걸리는 시간량은 DRAW1A, DRAW1B, DRAW1C, 및 DRAW1D 가 실행되는데 걸리는 시간량들의 합과 동일할 수도 있고, 그래픽스 프레임의 렌더링 동안 DRAW2 가 실행되는데 걸리는 시간량은 DRAW2A, DRAW2B, DRAW2C, 및 DRAW2D 가 실행되는데 걸리는 시간량들의 합과 동일할 수도 있다.

[0165] 일부 예들에서, GPU 드라이버 (28) 는 그래픽스 애플리케이션 (예를 들어, 소프트웨어 애플리케이션 (24)) 으로부터 타임스탬프 요청 (예를 들어, TSR2) 을 수신할 수도 있다. GPU 드라이버 (28) 는 타임스탬프 요청 (예를 들어, TSR2) 에 기초하여 복수의 빈당 타임스탬프 요청들 (예를 들어, TSR2A, TSR2B, TSR2C, TSR2D) 을 생성할 수도 있다. GPU 드라이버 (28) 는 빈당 타임스탬프 요청들 (예를 들어, TSR2A, TSR2B, TSR2C, TSR2D) 각각을 복수의 커맨드 스트림들 (예를 들어, 각각 커맨드 스트림들 (56, 58, 60, 62)) 중 각각의 커맨드 스트림에 배치할 수도 있다. 커맨드 스트림들 (예를 들어, 커맨드 스트림들 (56, 58, 60, 62)) 각각은 타일 기반

렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 (예를 들어, 각각 렌더링 패스 반복들 (A, B, C, 및 D)) 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 실행될 수도 있다. 렌더링 패스 반복들 (예를 들어, 렌더링 패스 반복들 (A, B, C, D)) 각각은 렌더 타겟의 복수의 서브-영역들 (예를 들어, 각각 빈들 (A, B, C, 및 D)) 중 각각의 서브-영역을 렌더링하도록 구성될 수도 있다.

[0166] GPU 드라이버 (28) 는 GPU (12) 로 하여금 커맨드 스트림들 (예를 들어, 커맨드 스트림들 (56, 58, 60, 62)) 을 실행하게 할 수도 있다. GPU 드라이버 (28) 및/또는 GPU (12) 는 커맨드 스트림들에 배치된 빈당 타임스탬프 요청들 (예를 들어, TSR2A, TSR2B, TSR2C, TSR2D) 에 응답하여 GPU (12) 에 의해 생성된 빈당 타임스탬프 값들에 기초하여 타임스탬프 값 (예를 들어, TSR2) 을 생성할 수도 있다.

[0167] 도 4 내지 도 6 은 그래픽스 프레임을 렌더링하기 위한 3 개의 드로우 콜 커맨드들을 포함하는 예시적인 커맨드 스트림 및 렌더 타겟을 4 개의 상이한 서브-영역들 또는 빈들로 서브분할하는 예시적인 타일 기반 렌더링 시스템을 도시한다. 그러나, 본 개시물에 설명된 기법들은 렌더링될 각각의 그래픽스 프레임에 대한 동일하거나 상이한 수의 드로우 콜 커맨드들을 포함하는 커맨드 스트림들로 및 그래픽스 프레임을 동일하거나 상이한 수의 서브-영역들 또는 빈들로 서브분할하는 타일 기반 렌더링 시스템들로 구현될 수도 있다는 것이 이해되어야 한다.

[0168] 일부 예들에서, 비닝 패스는 도 5 에 예시된 렌더링 패스 반복들 전에 수행될 수도 있다. 이러한 예들에서, 비닝 패스는 렌더링 패스 반복들 (예를 들어, "렌더링 패스 반복 A") 과 유사한 방식으로 처리될 수도 있다. 이러한 예들에서, GPU 드라이버 (28) 및/또는 GPU (12) 는 비닝 패스 동안 요청되는 하나 이상의 비닝 패스당 타임스탬프들에 기초하여 및 렌더링 패스의 하나 이상의 반복들 동안 요청되는 하나 이상의 빈당 타임스탬프 요청들에 기초하여 하나 이상의 타임스탬프들을 생성할 수도 있다.

[0169] 일부 예들에서, GPU 드라이버 (28) 및/또는 GPU (12) 는 렌더링될 그래픽스 프레임에 대한 타임스탬프 값들 중 적어도 하나의 타임스탬프 값이 렌더링될 그래픽스 프레임에 대한 타임스탬프 값들 중 적어도 하나의 다른 타임스탬프 값과는 상이하도록 렌더링될 그래픽스 프레임에 대한 복수의 타임스탬프 요청들에 응답하여 복수의 타임스탬프 값들을 생성할 수도 있다. 추가 예들에서, GPU 드라이버 (28) 및/또는 GPU (12) 는 타임스탬프 값들이 렌더링될 그래픽스 프레임에 대한 커맨드 스트림의 시작에서 렌더링될 그래픽스 프레임에 대한 커맨드 스트림의 끝까지 단조 증가하도록 렌더링될 그래픽스 프레임에 대한 복수의 타임스탬프 요청들에 응답하여 복수의 타임스탬프 값들을 생성할 수도 있다.

[0170] 도 7 은 본 개시물에 따른 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프들을 지원하기 위한 예시적인 기법을 예시하는 플로우 다이어그램이다. CPU (6) 및/또는 GPU (12) 는 타임스탬프 요청을 수신한다 (70). 일부 예들에서, 타임스탬프 요청은 CPU (6) 상에서 실행되는 그래픽스 애플리케이션으로부터 수신될 수도 있다. CPU (6) 및/또는 GPU (12) 는 그래픽스 프레임에 대해 타일 기반 렌더링을 수행하는 동안 GPU (12) 에 의해 생성되는 복수의 빈당 타임스탬프 값들에 기초하여 시점을 나타내는 타임스탬프 값을 생성한다 (72). 타임스탬프 값은 복수의 빈당 타임스탬프 값들 중 적어도 2 개의 빈당 타임스탬프 값들의 함수일 수도 있다.

[0171] CPU (6) 및/또는 GPU (12) 는 타임스탬프 값을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. 일부 예들에서, CPU (6) 및/또는 GPU (12) 는 수식 (1) 내지 수식 (7) 의 하나 이상에 기초하여 타임스탬프 값을 생성할 수도 있다. 타임스탬프 값들을 생성하기 위해 빈당 타임스탬프 값들을 이용하는 것은 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에 의해 인트라-프레임 타임스탬프 요청들이 지원되는 것을 허용할 수도 있다.

[0172] 도 8 은 본 개시물에 따른 타일 기반 렌더링을 수행하는 그래픽스 프로세싱 시스템에서 인트라-프레임 타임스탬프들을 지원하기 위한 다른 예시적인 기법들을 예시하는 플로우 다이어그램이다. 일부 예들에서, 도 8 에 예시된 기법들 중 일부 또는 전부는 도 7 에 예시된 기법들 중 일부 또는 전부를 수행하는데 이용될 수도 있다.

[0173] CPU (6) 는 타임스탬프 요청을 수신한다 (74). 일부 예들에서, 타임스탬프 요청은 CPU (6) 상에서 실행되는 그래픽스 애플리케이션으로부터 수신될 수도 있다. CPU (6) 는 타임스탬프 요청에 기초하여 복수의 빈당 타임스탬프 요청들을 생성한다 (76). CPU (6) 는 빈당 타임스탬프 요청들 각각을 복수의 커맨드 스트림들 중 각각의 커맨드 스트림에 배치한다 (78). 커맨드 스트림들 각각은 타일 기반 렌더링을 수행하는 동안 일어나는 복수의 렌더링 패스 반복들 중 각각의 렌더링 패스 반복 동안 GPU (12) 에 의해 실행될 수도 있다. 렌더링 패스 반복들 각각은 렌더 타겟의 복수의 서브-영역들 중 각각의 서브-영역을 렌더링하도록 구성될 수도

있다.

[0174] CPU (6) 는 GPU (12) 로 하여금 커맨드 스트림들을 실행하게 한다 (80). CPU (6) 및/또는 GPU (12) 는 커맨드 스트림들에 배치된 빈당 타임스탬프 요청들에 응답하여 GPU (12) 에 의해 생성된 빈당 타임스탬프 값들에 기초하여 타임스탬프 값을 생성한다 (82). CPU (6) 및/또는 GPU (12) 는 타임스탬프 값을 생성하기 위해 본 개시물에 설명된 기법들 중 임의의 것을 이용할 수도 있다. 일부 예들에서, CPU (6) 및/또는 GPU (12) 는 수식 (1) 내지 수식 (7) 의 하나 이상에 기초하여 타임스탬프 값을 생성할 수도 있다.

[0175] 상이한 렌더링 패스 반복들 동안 생성되는 빈당 타임스탬프 값들을 이용하는 것은 그래픽스 프레임의 렌더링 동안 상이한 드로우 콜 커맨드들이 실행되는데 걸리는 상대적 시간량들을 적어도 어느 정도까지는 반영하는 애플리케이션 요청 타임스탬프 값들을 그래픽스 프로세싱 시스템이 생성하는 것을 허용할 수도 있다. 이렇게 하여, 본 개시물에서 설명된 기법들은 타일 기반 렌더링 시스템이 비연속적 인터리빙된 방식으로 드로우 콜 커맨드들을 실행하는 경우들에도, 타일 기반 렌더링 시스템이 인트라-프레임 타임스탬프들을 지원하는 것을 허용할 수도 있다.

[0176] 그들 자신이 성능 페널티를 부과하지 않는 정확한 인트라-프레임 타임스탬프들은 타일 기반 렌더링 아키텍처들에 관해 획득하기 어려울 수도 있다. 이것은 비닝 및 다이렉트 렌더링 사이에 동적으로 스위칭할 수 있는 드라이버들에 관해 훨씬 더 어려울 수도 있다. 일부 예들에서, 본 개시물의 기법들은 비닝 및 다이렉트 렌더링 양자를 위해 작용할 수도 있는 합리적으로 정확한, 대표적인 인트라-프레임 타임스탬프를 구현할 수도 있다.

[0177] 일부 예들에서, 각각의 타임스탬프 요청은 빈당 타임스탬프로 변경될 수도 있다. 주어진 렌더 타겟에 대한 렌더링이 프로세싱된 후, 빈-시작으로부터 각각의 빈에 대한 타임스탬프까지의 평균 시간이 타임스탬프로서 생성 및 이용될 수도 있다. 이것은 다이렉트 렌더링 컴포넌트들에 의해 생성된 것들과 등가인 타임스탬프를 제공할 수도 있다. 본 개시물의 기법들은 타일 기반 렌더링 GPU 가 인트라-프레임 타임스탬프들을 지원하는 것을 허용할 수도 있다.

[0178] 본 개시물에 설명된 기법들은 적어도 부분적으로, 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 임의의 조합에서 구현될 수도 있다. 예를 들어, 설명된 기법들의 다양한 양태들은 하나 이상의 마이크로프로세서들, 디지털 신호 프로세서들 (DSP들), 주문형 집적 회로들 (ASIC들), 필드 프로그램가능 게이트 어레이들 (FPGA들), 또는 임의의 다른 등가의 집적 또는 이산 로직 회로는 물론 이러한 컴포넌트들의 임의의 조합을 포함하는 하나 이상의 프로세서들 내에 구현될 수도 있다. 용어 "프로세서" 또는 "프로세싱 회로" 는 일반적으로 전술한 로직 회로의 임의의 것을, 단독으로 또는 다른 로직 회로, 또는 프로세싱을 수행하는 이산 하드웨어와 같은 임의의 다른 등가의 회로와 조합하여 나타낼 수도 있다.

[0179] 이러한 하드웨어, 소프트웨어, 및 펌웨어는 본 개시물에 설명된 다양한 동작들 및 기능들을 지원하도록 동일한 디바이스 내에 또는 별개의 디바이스들 내에 구현될 수도 있다. 또한, 설명된 유닛들, 모듈들 또는 컴포넌트들 중 임의의 것은 별개이지만 상호동작가능한 로직 디바이스들로서 함께 또는 별도로 구현될 수도 있다. 상이한 피처들의 모듈들 또는 유닛들로서의 도시는 상이한 기능적 양태들을 하이라이트하도록 의도되지 않고 이러한 모듈들 또는 유닛들이 별개의 하드웨어 또는 소프트웨어 컴포넌트들에 의해 실현되어야 한다는 것을 반드시 의미하지 않는다. 오히려, 하나 이상의 모듈들 또는 유닛들과 연관된 기능성은 별개의 하드웨어, 펌웨어, 및/또는 소프트웨어 컴포넌트에 의해 수행되거나, 또는 공통 또는 별개의 하드웨어 또는 소프트웨어 컴포넌트들 내에 통합될 수도 있다.

[0180] 본 개시물에 설명된 기법들은 또한, 명령들을 저장하는 컴퓨터 판독가능 저장 매체와 같은 컴퓨터 판독가능 매체에 저장, 수록 또는 인코딩될 수도 있다. 컴퓨터 판독가능 매체에 수록 또는 인코딩된 명령들은 하나 이상의 프로세서들로 하여금, 예를 들어, 명령들이 하나 이상의 프로세서들에 의해 실행될 때 본 명세서에 설명된 기법들을 수행하게 할 수도 있다. 일부 예들에서, 컴퓨터 판독가능 매체는 비일시적 컴퓨터 판독가능 저장 매체일 수도 있다. 컴퓨터 판독가능 저장 매체들은 랜덤 액세스 메모리 (RAM), 판독 전용 메모리 (ROM), 프로그램가능 판독 전용 메모리 (PROM), 소거가능한 프로그램가능 판독 전용 메모리 (EPROM), 전자적으로 소거가능한 프로그램가능 판독 전용 메모리 (EEPROM), 플래시 메모리, 하드 디스크, CD-ROM, 플로피 디스크, 카세트, 자기 매체들, 광학 매체들, 또는 유형의 다른 컴퓨터 판독가능 저장 매체들을 포함할 수도 있다.

[0181] 컴퓨터 판독가능 매체들은 상기 열거된 것들과 같은 유형의 저장 매체에 대응하는 컴퓨터 판독가능 저장 매체들을 포함할 수도 있다. 컴퓨터 판독가능 매체들은 또한, 예를 들어, 통신 프로토콜에 따라 일 장소로부터 타

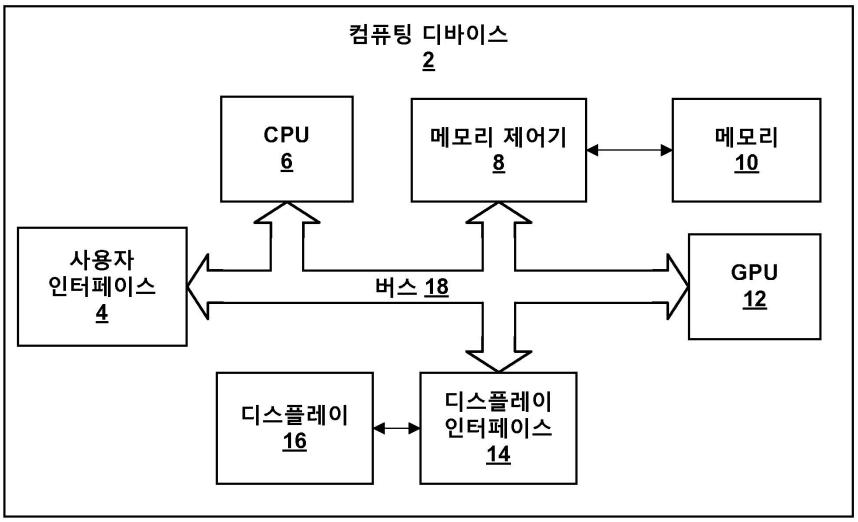
장소로의 컴퓨터 프로그램의 전송을 용이하게 하는 임의의 매체를 포함하는 통신 매체들을 포함할 수도 있다.

이 방식으로, 어구 "컴퓨터 판독가능 매체들" 은 일반적으로 (1) 비일시적인 유형의 컴퓨터 판독가능 저장 매체들, 및 (2) 일시적 신호 또는 반송파와 같은 비유형의 컴퓨터 판독가능 통신 매체에 대응할 수도 있다.

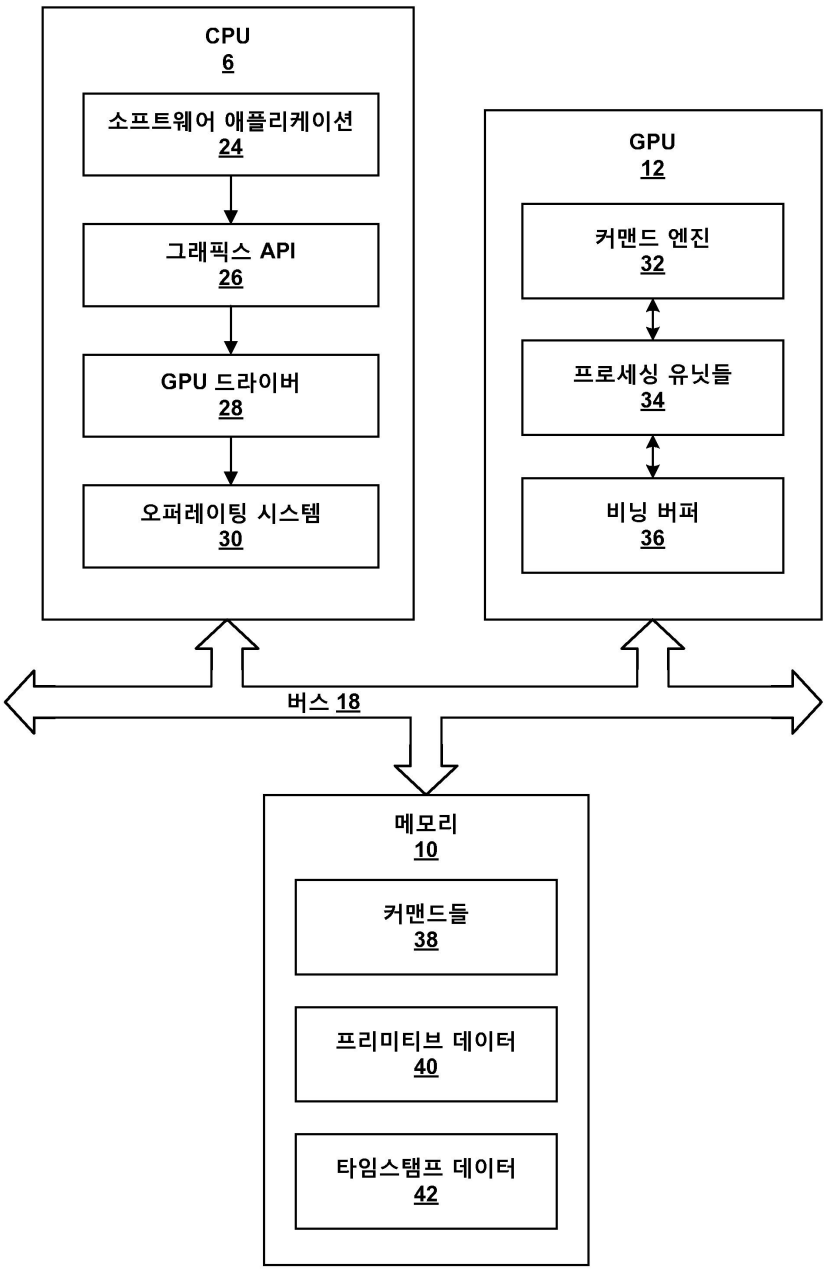
[0182] 다양한 양태들 및 예들이 설명되었다. 그러나, 본 개시물의 구조 또는 기법들에는 다음의 청구항들의 범위로부터 벗어남 없이 변경들이 행해질 수 있다.

도면

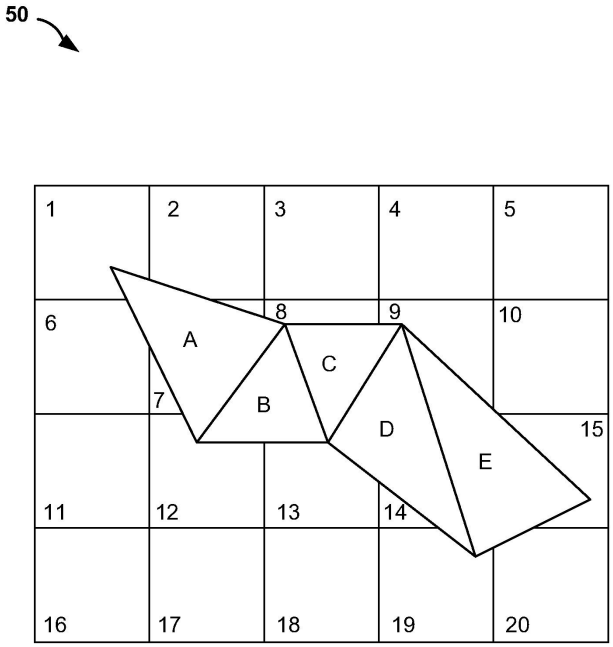
도면1



도면2


CPU
6
소프트웨어 애플리케이션
24
그래픽스 API
26
GPU 드라이버
28
오퍼레이팅 시스템
30
GPU
12
커맨드 엔진
32
프로세싱 유닛들
34
비닝 버퍼
36
버스 18
메모리
10
커맨드들
38
프리미티브 데이터
40
타임스탬프 데이터
42

도면3

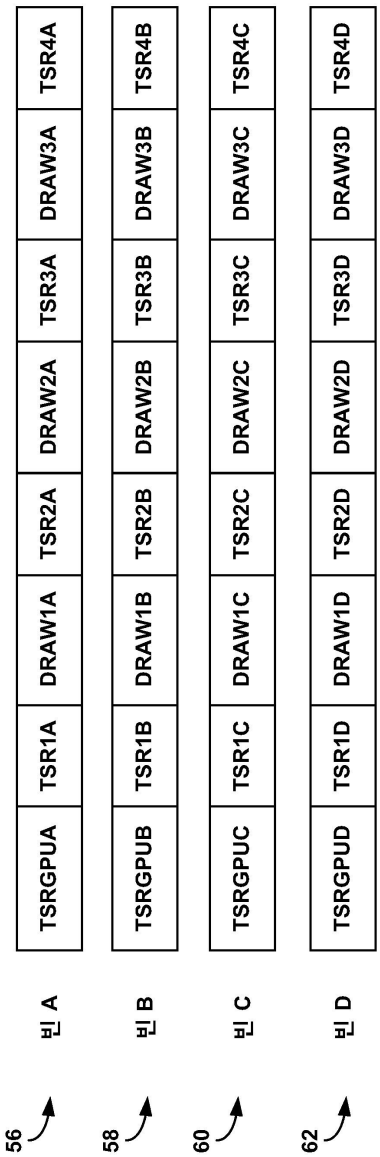

50
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
A
B
C
D
E

도면4

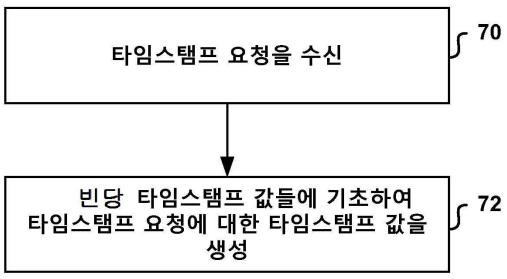

TSR4
DRAW3
TSR3
DRAW2
TSR2
DRAW1
TSR1
52

도면5

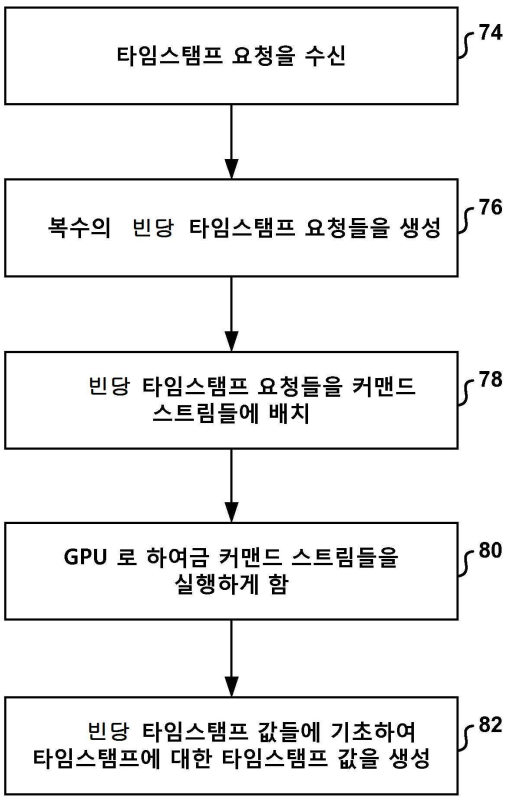
도면6


TSRGPUA
TSR1A
DRAW1A
TSR2A
DRAW2A
TSR3A
DRAW3A
TSR4A
TSRGPUB
TSR1B
DRAW1B
TSR2B
DRAW2B
TSR3B
DRAW3B
TSR4B
TSRGPUC
TSR1C
DRAW1C
TSR2C
DRAW2C
TSR3C
DRAW3C
TSR4C
TSRGPUD
TSR1D
DRAW1D
TSR2D
DRAW2D
TSR3D
DRAW3D
TSR4D
빈 A
빈 B
빈 C
빈 D
56
58
60
62

도면7


타임스탬프 요청을 수신
70
빈당 타임스탬프 값들에 기초하여 타임스탬프 요청에 대한 타임스탬프 값을 생성
72

도면8


타임스탬프 요청을 수신
74
복수의 빈당 타임스탬프 요청들을 생성
76
빈당 타임스탬프 요청들을 커맨드 스트림들에 배치
78
GPU 로 하여금 커맨드 스트림들을 실행하게 함
80
빈당 타임스탬프 값들에 기초하여 타임스탬프에 대한 타임스탬프 값을 생성
82