(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0027964
(43) 공개일자 2022년03월08일

(51) 국제특허분류(Int. Cl.)
G09G 5/36 (2006.01) *G06T 1/20* (2018.01)
G06T 15/20 (2011.01)
(52) CPC특허분류
G09G 5/363 (2013.01)
G06T 1/20 (2013.01)
(21) 출원번호 10-2022-7001688
(22) 출원일자(국제) 2020년05월11일
심사청구일자 없음
(85) 번역문제출일자 2022년01월17일
(86) 국제출원번호 PCT/IB2020/054450
(87) 국제공개번호 WO 2020/260966
국제공개일자 2020년12월30일
(30) 우선권주장
16/457,179 2019년06월28일 미국(US)

(71) 출원인
에이티아이 테크놀로지스 유엘씨
캐나다 온타리오 엘3티 7엑스6 마크햄 커머스 밸리 드라이브 이스트 1
(72) 발명자
챈 벤자민 쿤 팬
캐나다 온타리오 엘3티 7엑스6 마크햄 커머스 밸리 드라이브 이스트 1
앳킨슨 윌리엄 로이드
캐나다 온타리오 엘3티 7엑스6 마크햄 커머스 밸리 드라이브 이스트 1
(뒷면에 계속)
(74) 대리인
박장원

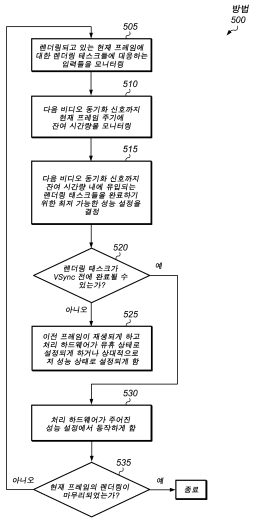
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **성능 보장 전력 관리를 갖는 실시간 GPU 렌더링**

(57) 요 약

성능 보장 전력 관리를 갖는 실시간 비디오 렌더링을 수행하기 위한 시스템들, 장치들 및 방법들이 개시된다. 시스템은 렌더링 태스크들을 수행하기 위한 적어도 소프트웨어 드라이버, 전력 관리 유닛, 및 복수의 처리 엘리먼트들을 포함한다. 시스템은 수행될 필요가 있는 렌더링 태스크들에 대응하는 입력들을 수신한다. 소프트웨어 드라이버는 수신된 입력들 및 대응하는 렌더링 태스크들의 수를 모니터링한다. 또한 소프트웨어 드라이버는 다음 비디오 동기화 신호까지 잔여 시간량을 모니터링한다. 소프트웨어 드라이버는 다음 비디오 동기화 신호 이전에 현재 프레임에 대한 렌더링 태스크를 완료하는 데 충분한 시간을 허용하면서 전력 소비를 최소화할 성능 설정을 결정한다. 그런 다음, 소프트웨어 드라이버는 전력 관리 유닛으로 하여금, 이들이 현재 프레임에 대한 렌더링 태스크들을 수행할 때, 이러한 성능 설정을 복수의 처리 엘리먼트들에 제공하게 한다.

대 표 도 - 도5



(52) CPC특허분류
G06T 15/205 (2013.01)
G09G 2310/08 (2013.01)
G09G 2330/021 (2013.01)
G09G 2354/00 (2013.01)
G09G 2360/08 (2013.01)
(72) 발명자
아이피 클라렌스
캐나다 온타리오 엘3티 7엑스6 마크햄 커머스 밸리 드라이브 이스트 1
콩 통 추엔
캐나다 온타리오 엘3티 7엑스6 마크햄 커머스 밸리 드라이브 이스트 1

명 세 서

청구범위

청구항 1

시스템에 있어서,

하나 이상의 처리 엘리먼트;

전력 관리 유닛; 및

제어 유닛을 포함하고, 상기 제어 유닛은,

렌더링되고 있는 현재 프레임에 대해 얼마나 많은 렌더링 태스크가 수행되기를 기다리고 있는지를 나타내는 입력을 모니터링하고;

다음 비디오 동기화 신호까지 잔여 시간량을 모니터링하고;

전력 소비 목표를 달성하면서 상기 다음 상기 비디오 동기화 신호까지 잔여 시간량 내에 하나 이상의 렌더링 태스크가 완료되도록 하기 위한 주어진 성능 설정을 선택하고; 및

상기 주어진 성능 설정의 표시를 전력 관리 유닛에 전달하여, 상기 하나 이상의 처리 엘리먼트가 상기 주어진 성능 설정에서 동작하게 하도록 구성된, 시스템.

청구항 2

제1항에 있어서, 상기 제어 유닛은, 상기 입력이 상기 다음 비디오 동기화의 신호가 감소될 때까지, 잔여 시간량 동안 더 많은 렌더링 태스크가 큐잉되었음을 나타낸다는 결정에 응답하여, 상기 주어진 성능 설정을 증가시키도록 구성된, 시스템.

청구항 3

제1항에 있어서, 상기 제어 유닛은, 상기 다음 비디오 동기화 신호까지 잔여 시간량 내에 상기 하나 이상의 렌더링 태스크가 완료될 수 있는 최저 가능한 성능 설정을 선택하도록 구성된, 시스템.

청구항 4

제1항에 있어서, 상기 제어 유닛은, 잔여 시간량과 렌더링 태스크의 수의 조합을 상기 주어진 성능 설정에 매핑하도록 구성된, 시스템

청구항 5

제4항에 있어서, 상기 제어 유닛은 상기 렌더링 태스크의 수와 상기 잔여 시간량의 조합을 상기 주어진 성능 설정에 맵핑하기 위한 테이블을 유지하도록 구성되고, 상기 테이블의 각 엔트리는 주어진 렌더링 태스크의 수와 주어진 잔여 시간량을 대응하는 성능 설정에 맵핑하는, 시스템.

청구항 6

제1항에 있어서, 상기 입력은 가상 현실 환경에서의 사용자 움직임을 포함하는, 시스템.

청구항 7

제1항에 있어서, 상기 제어 유닛은,

상기 렌더링 태스크의 수가 임계치보다 크다는 결정에 응답하여, 상기 하나 이상의 처리 엘리먼트가 상대적으로 고 성능 설정에서 동작하게 하고, 상기 임계치는 상기 다음 비디오 동기화 신호까지 잔여 시간량에 기초하고;

상기 렌더링 태스크의 수가 임계치 이하라는 결정에 응답하여, 상기 하나 이상의 처리 엘리먼트가 상대적으로 낮은 성능 설정에서 동작하게 하도록 추가로 구성된, 시스템.

청구항 8

방법에 있어서,

제어 유닛에 의해, 렌더링되는 현재 프레임에 대해 얼마나 많은 렌더링 태스크가 수행되기를 기다리고 있는지를 나타내는 입력을 모니터링하는 단계;

다음 비디오 동기화 신호까지 잔여 시간량을 모니터링하는 단계;

전력 소비 목표를 달성하면서 상기 다음 비디오 동기화 신호까지 잔여 시간 량 내에 하나 이상의 렌더링 태스크가 완료되도록 하기 위한 주어진 성능 설정을 선택하는 단계; 및

하나 이상의 처리 엘리먼트가 상기 주어진 성능 설정에서 동작하게 하기 위해 전력 관리 유닛에 상기 주어진 성능 설정의 표시 전달하는 단계를 포함하는, 방법.

청구항 9

제8항에 있어서, 상기 입력이 상기 다음 비디오 동기화의 신호가 감소될 때까지, 잔여 시간량 동안 더 많은 렌더링 태스크가 큐잉되었음을 나타낸다는 결정에 응답하여, 상기 주어진 성능 설정을 증가시키는 단계를 더 포함하는, 방법.

청구항 10

제8항에 있어서, 상기 다음 비디오 동기화 신호까지 잔여 시간량 내에 상기 하나 이상의 렌더링 태스크가 완료될 수 있는 최저 가능한 성능 설정을 선택하는 단계를 더 포함하는, 방법.

청구항 11

제8항에 있어서, 잔여 시간량과 렌더링 태스크 수의 조합을 상기 주어진 성능 설정에 매핑하는 단계를 더 포함하는, 방법.

청구항 12

제11항에 있어서, 상기 렌더링 태스크의 수 및 상기 잔여 시간량의 조합을 상기 주어진 성능 설정에 매핑하기 위한 테이블을 유지하는 단계를 더 포함하고, 상기 테이블 내의 각각의 엔트리는 주어진 렌더링 태스크의 수 및 주어진 잔여 시간량을 대응하는 성능 설정에 매핑하는, 방법.

청구항 13

제8항에 있어서, 상기 입력은 가상 현실 환경에서의 사용자 움직임을 포함하는, 방법.

청구항 14

제8항에 있어서, 다음을 더 포함하는, 방법.

상기 렌더링 태스크의 수가 임계치보다 크다는 결정에 응답하여, 상기 하나 이상의 처리 엘리먼트가 상대적으로 고 성능 설정에서 동작하게 하는 단계로서, 상기 임계치는 상기 다음 비디오 동기화 신호까지 잔여 시간량에 기초하는, 상기 고 성능 설정에서 동작하게 하는 단계;

상기 렌더링 태스크의 수가 임계치 이하라는 결정에 응답하여, 상기 하나 이상의 처리 엘리먼트가 상대적으로 낮은 성능 설정에서 동작하게 하는 단계를 더 포함하는, 방법.

청구항 15

장치에 있어서,

제1 프로세서;

제2 프로세서; 및

프로그램 명령어를 저장하는 메모리를 포함하고, 상기 프로그램 명령어는 상기 제1 프로세서에 의해,

렌더링되고 있는 현재 프레임에 대해 얼마나 많은 렌더링 태스크가 수행되기를 기다리고 있는지를 나타내는 입

력을 모니터링하고;

다음 비디오 동기화 신호까지 잔여 시간량을 모니터링하고;

전력 소비 목표를 달성하면서 상기 다음 상기 비디오 동기화 신호까지 잔여 시간량 내에 하나 이상의 렌더링 태스크가 완료되도록 하기 위한 주어진 성능 설정을 선택하고; 및

상기 제2 프로세서가 상기 주어진 성능 설정에서 동작하게 하도록 실행 가능한, 장치.

청구항 16

제15항에 있어서, 상기 프로그램 명령어는, 상기 입력이 상기 다음 비디오 동기화의 신호가 감소될 때까지, 잔여 시간량 동안 더 많은 렌더링 태스크가 큐잉되었음을 나타낸다는 결정에 응답하여, 상기 주어진 성능 설정을 증가시키도록 상기 제1 프로세서에 의해 실행 가능한, 장치.

청구항 17

제15항에 있어서, 상기 프로그램 명령어는 상기 하나 이상의 렌더링 태스크가 상기 다음 비디오 동기화 신호까지 잔여 시간량 내에 완료되게 하는 최저 가능한 성능 설정을 선택하도록 상기 제1 프로세서에 의해 실행가능한, 장치.

청구항 18

제15항에 있어서, 상기 프로그램 명령어는 잔여 시간량과 렌더링 태스크의 수의 조합을 상기 주어진 성능 설정에 매핑하도록 상기 제1 프로세서에 의해 실행 가능한, 장치.

청구항 19

제18항에 있어서, 상기 프로그램 명령어는 상기 렌더링 태스크의 수와 상기 잔여 시간량의 조합을 상기 주어진 성능 설정에 맵핑하기 위한 테이블을 유지하도록 상기 제1 프로세서에 의해 실행 가능하고, 상기 테이블의 각 엔트리는 주어진 렌더링 태스크의 수와 주어진 잔여 시간량을 대응하는 성능 설정에 맵핑하는, 장치.

청구항 20

제15항에 있어서, 상기 프로그램 명령어는 상기 제1 프로세서에 의해,

상기 렌더링 태스크의 수가 임계치보다 크다는 결정에 응답하여, 상기 제2 처리 엘리먼트가 상대적으로 고 성능 설정에서 동작하게 하고, 상기 임계치는 상기 다음 비디오 동기화 신호까지 잔여 시간량에 기초하고;

상기 렌더링 태스크의 수가 임계치 이하라는 결정에 응답하여, 상기 제2 처리 엘리먼트가 상대적으로 낮은 성능 설정에서 동작하게 하도록 실행 가능한, 장치.

발명의 설명

기 술 분 야

배 경 기 술

[0001] 다양한 애플리케이션들은 이미지들 또는 비디오 콘텐츠의 실시간 렌더링에 의존한다. 예를 들어, 클라우드 게임, 가상 현실, 및 게임 관람은 콘텐츠의 실시간 렌더링을 수반하는 애플리케이션들의 예들이다. 비디오 프레임들의 실시간 렌더링은 종종 많은 양의 전력을 소비하는 상당한 양의 처리 자원들을 사용한다. 실시간 렌더링 환경에서, 생성된 이미지 프레임 레이턴시들을 제어하기 위한 요건 및 손실된 프레임들을 회피하려는 욕구는 전력 관리에 대한 특별한 요구들을 제기한다. 한편, 레이턴시를 최소화하고 이미지의 렌더링이 정시에 종료되는 것을 보장하기 위해 가능한 가장 높은 클록 레이트(clock rate)로 실행되는 것이 바람직하다. 반면에, 처리 하드웨어가 과열하기 시작하거나 열 임계치에 가까워지면, 하드웨어는 그 클록 레이트를 감소시킬 것이고, 이는 그런 다음 손실된 프레임들을 초래한다. 이러한 문제는 전력 또는 열적으로 제약이 있는 플랫폼에서는 특히 어렵다.

[0002] 다양한 프레임 기반 실시간 애플리케이션들은 프레임당 다수의 작업들을 제출하고 이 프로세스를 일정한 또는 가변적인 프레임 레이트로 반복하는 게임 애플리케이션들 뿐만 아니라 다른 유형들의 렌더링 애플리케이션들을 포함한다. 프레임당 처리 단위 작업 부하(workload)(예를 들어, 작업 수, 작업당 시간, 작업당 자원)는 애플리케이션 런타임 동작에 따라 복잡성 및 계산 요구가 달라질 수 있다. 이러한 애플리케이션들에 대해, 처리 유닛은 프레임의 적시에 사용(예를 들어, 디스플레이 또는 송신)할 수 있도록 충분히 일찍 프레임 실행을 완료하거나, 처리 유닛은 프레임 실행을 완료하는 데 늦으며, 이는 프레임이 드롭되거나 나중에 소비되게 한다. 이러한 지연은 사용자 경험에 부정적인 영향을 미친다.

[0003] 이러한 관점에서, 성능 보장 전력 관리를 갖는 실시간 비디오 렌더링을 관리하기 위한 개선된 방법들이 요구된다.

도면의 간단한 설명

[0004] 본 출원에 설명된 방법들 및 메커니즘들의 이점들은 첨부 도면들과 함께 이하의 설명을 참조함으로써 더 잘 이해될 수 있다:

도 1은 컴퓨팅 시스템의 일 구현예의 블록도이다.

도 2는 컴퓨팅 시스템의 일 구현예의 블록도이다.

도 3은 큐(queue) 점유에 기초하여 렌더링되고 있는 프레임들에 대한 성능 설정을 선택하는 일 구현예의 타이밍 다이어그램이다.

도 4 는 일 구현예에 따른 다수의 유입(incoming) 태스크들 및 잔여 시간을 성능 설정에 맵핑하기 위한 테이블의 예이다.

도 5는 성능 보장 전력 관리를 갖는 실시간 비디오 렌더링을 수행하기 위한 방법의 일 구현예를 예시하는 개괄적인 흐름도이다.

도 6 은 애플리케이션 유형에 기초하여 하드웨어를 처리하기 위한 성능 설정을 제어하기 위한 방법의 일 구현예를 예시하는 개괄적인 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0005] 이하의 설명에서, 본 출원에 제시된 방법들 및 메커니즘들의 철저한 이해를 제공하기 위해 다수의 특정 세부사항들이 제시된다. 그러나, 당업자는 다양한 구현들이 이러한 특정 세부사항들 없이 실시될 수 있다는 것을 인식해야 한다. 일부 예들에서, 공지의 구조들, 컴포넌트들, 신호들, 컴퓨터 프로그램 명령어들, 및 기술들은 본 출원에 설명된 접근법들을 모호하게 하는 것을 피하기 위해 상세히 도시되지 않았다. 예시의 단순성 및 명확성을 위해, 도면들에 도시된 엘리먼트들이 반드시 축척에 맞게 그려지지 않았다는 것이 이해될 것이다. 예를 들어, 엘리먼트들 중 일부의 치수들은 다른 엘리먼트들에 비해 과장될 수 있다.

[0006] 이벤트 기반 피드 포워드(feed-forward) 제어 윈도우 구동 사용 케이스에 기초하여 성능 보장 전력 관리 관리기를 갖는 실시간 GPU 렌더링을 구현하기 위한 시스템들, 장치들 및 방법들이 본 출원에 개시된다. 일 구현예에서, 시스템은 렌더링 태스크(rendering task)들을 수행하기 위한 적어도 소프트웨어 드라이버, 전력 관리 유닛, 및 하나 이상의 처리 엘리먼트들을 포함한다. 시스템은 수행될 필요가 있는 렌더링 태스크들에 대응하는 입력들을 수신한다. 소프트웨어 드라이버는 수신된 입력의 수 및 대응하는 렌더링 태스크의 수를 모니터링한다. 또한 소프트웨어 드라이버는 다음 비디오 동기화 신호까지 잔여 시간량을 모니터링한다. 소프트웨어 드라이버는 다음 비디오 동기화 신호 전에 현재 프레임에 대한 렌더링 태스크를 완료하는 데 충분한 시간을 허용하면서 전력 소비를 최소화할 성능 설정을 결정한다. 그런 다음, 소프트웨어 드라이버는 전력 관리 유닛이 현재 비디오 프레임에 대한 렌더링 태스크들을 수행할 때 이러한 성능 설정을 복수의 처리 엘리먼트들에 제공하게 한다.

[0007] 이제 도 1을 참조하면, 컴퓨팅 시스템(100)의 일 구현예의 블록도가 도시된다. 일 구현예에서, 컴퓨팅 시스템(100)은 적어도 프로세서들(105A-N), 제어 유닛(110), 입력/출력(I/O) 인터페이스들(120), 버스(125), 메모리 컨트롤러(들)(130), 네트워크 인터페이스(135), 메모리 디바이스(들)(140), 전력 공급 장치(power supply)(145), 전력 관리 유닛(150), 디스플레이 컨트롤러(160) 및 디스플레이(165)를 포함한다. 다른 구현예들에서, 컴퓨팅 시스템(100)은 다른 컴포넌트들을 포함하고 및/또는 컴퓨팅 시스템(100)은 상이하게 배열된다. 프

로세서들(105A-N)은 시스템(100)에 포함되는 임의의 수의 프로세서들을 나타내며, 프로세서들의 수는 구현예마다 변한다.

[0008] 일 구현예에서, 프로세서(105A)는 중앙 처리 유닛(CPU)과 같은 범용 프로세서이다. 일 구현예에서, 프로세서(105N)는 고병렬 아키텍처를 갖는 데이터 병렬 프로세서이다. 데이터 병렬 프로세서는 그래픽 처리 유닛(GPU), 디지털 신호 프로세서(DSP), 필드 프로그램 가능한 게이트 어레이(FPGA), 주문형 집적 회로(ASIC) 등을 포함한다. 일 구현예에서, 프로세서(105N)는 디스플레이(165)로 구동되도록 디스플레이 컨트롤러(160)에 픽셀들을 제공하는 GPU이다. 일부 구현예들에서, 프로세서들(105A-N)은 다수의 데이터 병렬 프로세서들을 포함한다. 일 구현예에서, 제어 유닛(110)은 프로세서(105A) 상에서 실행되는 소프트웨어 드라이버이다. 다른 구현예들에서, 제어 유닛(110)은 프로세서들(105A-N)과 독립적이고 및/또는 프로세서들(105A-N) 내에 통합되는 제어 로직을 포함한다. 일반적으로 말하면, 제어 유닛(110)은 소프트웨어 및/또는 하드웨어의 임의의 적절한 조합이다.

[0009] 메모리 컨트롤러(들)(130)는 프로세서들(105A-N)에 의해 액세스가능한 임의의 수 및 유형의 메모리 컨트롤러들을 나타낸다. 메모리 컨트롤러(들)(130)는 임의의 수 및 유형의 메모리 디바이스(들)(140)에 결합된다. 메모리 디바이스(들)(140)는 임의의 수 및 유형의 메모리 디바이스들을 나타낸다. 예를 들어, 메모리 디바이스(들)(140) 내의 메모리의 유형은 DRAM(Dynamic Random Access Memory), SRAM(Static Random Access Memory), NAND 플래시 메모리, NOR 플래시 메모리, FeRAM 등을 포함한다.

[0010] I/O 인터페이스들(120)은 임의의 수 및 유형의 I/O 인터페이스들(예를 들어, 주변 컴포넌트 상호연결(PCI) 버스, PCI-X(PCI-Extended), PCIE(PCI Express) 버스, 기가비트 이더넷(GBE) 버스, 범용 직렬 버스(USB))을 나타낸다. 다양한 유형의 주변 디바이스(미도시)가 I/O 인터페이스(120)에 결합된다. 이러한 주변 디바이스는 디스플레이, 키보드, 마우스, 프린터, 스캐너, 미디어 레코딩 디바이스, 외부 저장 디바이스, 네트워크 인터페이스 카드 등을 포함한다(그러나 이에 제한되지 않는다). 네트워크 인터페이스(135)는 네트워크를 통해 네트워크 메시지들을 수신하고 발송하는데 사용된다. 버스(125)는 시스템(100)의 상이한 컴포넌트들을 함께 연결하기 위한 임의의 수의 링크들을 갖는 임의의 유형의 버스 또는 패브릭을 나타낸다.

[0011] 일 구현예에서, 큐(들)(142)는 메모리 디바이스들(들)(140)에 저장된다. 다른 구현예들에서, 큐(들)(142)는 시스템(100) 내의 다른 위치들에 저장된다. 큐(들)(142)는 시스템 (100)에서 할당되는 임의의 수 및 유형의 큐들을 나타낸다. 일 구현예에서, 큐(들)(142)는 렌더링되고 있는 프레임들에 대해 수행될 렌더링 태스크들을 저장한다. 일 구현예에서, 렌더링 태스크들은 네트워크 인터페이스(135)를 통해 수신된 입력들에 기초하여 큐(들)(142)에 인큐잉(enqueue)된다. 예를 들어, 일 시나리오에서, 입력들은 비디오 게임 애플리케이션의 사용자에 의해 생성되고 네트워크(미도시)를 통해 시스템(100)으로 발송된다. 다른 구현예에서, 입력들은 I/O 인터페이스들(120)에 연결된 주변 디바이스에 의해 생성된다.

[0012] 일 구현예에서, 전력 관리 유닛 (150)은 전력 공급 장치(145)로부터 시스템(100)의 컴포넌트들에 전력을 공급하고, 전력 관리 유닛(150)은 시스템(100) 내의 컴포넌트들의 다양한 전력 성능 상태들을 제어한다. 제어 유닛(110)으로부터 업데이트를 수신한 것에 응답하여, 전력 관리 유닛 (150)은 시스템(100) 내의 다른 컴포넌트들이 그것들의 현재 전력 성능 상태를 증가시키거나 감소시키게 한다. 다양한 구현예들에서, 전력 성능 상태를 변경하는 것은 디바이스의 현재 동작 주파수를 변경하는 것 및/또는 디바이스의 현재 전압 레벨을 변경하는 것을 포함한다. 프로세서들(105A-N)의 전력 성능 상태들이 감소될 때, 이는 프로세서들(105A-N)에 의해 실행되는 컴퓨팅 태스크들이 완료하는데 더 오래 걸리게 한다.

[0013] 일 구현예에서, 제어 유닛(110)은 명령을 전력 관리 유닛 (150)으로 발송하여, 현재 프레임에 대한 렌더링 태스크의 수가 주어진 임계치보다 더 크다는 결정에 응답하여, 프로세서(105N)가 상대적으로 고 전력 성능 상태에서 동작하게 한다. 일 구현예에서, 주어진 임계치는 다음 비디오 동기화 신호까지 잔여 시간량에 기초하여 조정된다. 예를 들어, 다음 비디오 동기화 신호까지 잔여 시간이 적을수록, 주어진 임계치가 더 낮게 프로그래밍된다.

[0014] 다양한 구현예들에서, 컴퓨팅 시스템(100)은 컴퓨터, 랩탑, 모바일 디바이스, 서버, 또는 다양한 다른 유형들의 컴퓨팅 시스템들 또는 디바이스들 중 임의의 것이다. 컴퓨팅 시스템(100)의 컴포넌트들의 수는 구현마다 달라진다는 것에 유의한다. 예를 들어, 다른 구현예들에서, 각각의 컴포넌트는 도 1에 도시된 수보다 더 많거나 더 적다. 또한, 다른 구현예들에서, 컴퓨팅 시스템(100)은 도 1에 도시되지 않은 다른 컴포넌트들을 포함하고 및/또는 컴퓨팅 시스템(100)에 도시된 컴포넌트들 중 하나 이상이 생략된다는 것에 유의한다. 추가적으로, 다른 구현예들에서, 컴퓨팅 시스템(100)은 도 1에 도시된 것 이외의 다른 방식들로 구성된다.

[0015] 이제 도 2를 참조하면, 컴퓨팅 시스템(200)의 다른 구현의 블록도가 도시된다. 일 구현예에서, 시스템(200)은

GPU(205), 시스템 메모리(225), 및 GPU(205)에 속한 로컬 메모리(230)를 포함한다. 시스템(200)은 또한 도면을 모호하게 하는 것을 피하기 위해 도시되지 않은 다른 컴포넌트들을 포함한다. GPU(205)는 적어도 명령 프로세서(235), 스케줄러(250), 컴퓨팅 유닛(255A-N), 메모리 컨트롤러(220), 전역 데이터 공유(270), 레벨 1(L1) 캐시(265), 및 레벨 2(L2) 캐시(260)를 포함한다. 컴퓨팅 유닛들(255A-N)은 또한 본 출원에서 "복수의 처리 엘리먼트들"로 지칭될 수 있음에 유의한다. 다른 구현예들에서, GPU(205)는 다른 컴포넌트들을 포함하고, 예시된 컴포넌트들 중 하나 이상을 생략하고, 단지 하나의 인스턴스만이 도 2에 도시되어 있더라도 컴포넌트의 다수의 인스턴스들을 갖고, 및/또는 다른 적절한 방식들로 조직된다. 일 구현예에서, GPU(205)의 회로는 (도 1의) 프로세서(105N)에 포함된다.

[0016] 다양한 구현예들에서, 컴퓨팅 시스템(200)은 다양한 유형들의 소프트웨어 애플리케이션들 중 임의의 것을 실행한다. 주어진 소프트웨어 애플리케이션을 실행하는 것의 일부로서, 컴퓨팅 시스템(200)의 호스트 CPU(미도시)는 GPU(205) 상에서 수행될 렌더링 태스크들을 개시한다. 명령 프로세서(235)는 호스트 CPU로부터 명령들을 수신하고 스케줄러(250)를 사용하여 대응하는 렌더링 태스크들을 컴퓨팅 유닛들(255A-N)에 발행한다. 컴퓨팅 유닛들(255A-N) 상에서 실행되는 렌더링 태스크들은 GPU(205) 내의 전역 데이터 공유(270), L1 캐시(265), 및 L2 캐시(260)에 데이터를 판독 및 기록한다. 도 2에 도시되지 않았지만, 일 구현예에서, 컴퓨팅 유닛들(255A-N)은 또한 각각의 컴퓨팅 유닛(255A-N) 내에 하나 이상의 캐시 및/또는 로컬 메모리를 포함한다. 다양한 구현예들에서, 컴퓨팅 유닛들(255A-N)은 실시간으로 디스플레이되거나, 스트리밍되거나, 소비될 프레임들을 렌더링하고 있는 임의의 수의 프레임 기반 애플리케이션들을 실행한다. 일 구현예에서, 큐(들)(232)는 로컬 메모리(230)에 저장된다. 다른 구현예들에서, 큐(들)(232)는 시스템(200) 내의 다른 위치들에 저장된다. 큐(들)(232)는 시스템(200)에서 할당되는 큐들의 임의의 수 및 유형을 나타낸다. 일 구현예에서, 큐(들)(232)는 GPU(205)에 의해 수행될 렌더링 태스크들을 저장한다.

[0017] 일 구현예에서, GPU (205)의 성능 설정은 큐(들) (232)에 저장된 현재 프레임에 대한 렌더링 태스크들의 수에 기초하여 뿐만 아니라 다음 비디오 동기화 신호까지 잔여 시간량에 기초하여 조정된다. 다양한 구현예들에서, GPU(205)의 성능 설정은 전력 소비 목표를 달성하면서 다음 비디오 동기화 신호 전에 렌더링 태스크들을 종료하도록 조정된다. 일 구현예에서, 성능 설정은 제어 유닛(미도시)에 의해 조정된다. 제어 유닛은 CPU(미도시) 상에서 실행되는 소프트웨어 드라이버일 수 있거나, 제어 유닛은 프로그램가능 로직 디바이스(예를 들어, FPGA) 내에서 구현되는 제어 로직 또는 전용 하드웨어(예를 들어, ASIC) 로서 구현되는 제어 로직을 포함할 수 있다. 일부 경우에, 제어 유닛은 소프트웨어와 하드웨어의 조합을 포함한다.

[0018] 일 구현예에서, GPU (205)의 성능 설정은 GPU (205)의 특정 전력 설정, 전력 상태, 또는 동작 지점에 대응한다. 일 구현예에서, 제어 유닛은 전력 소비를 선택된 전력 할당으로 제한하기 위해 GPU(205)의 주파수 및/또는 전압을 변경하기 위해 DVFS(dynamic voltage and frequency scaling)을 사용한다. 각각의 개별 주파수 및 전압 설정은 개별 성능 설정에 대응할 수 있다. 일 구현예에서, 제어 유닛에 의해 선택된 성능 설정은 PLL(phase-locked loop) 유닛(미도시)을 제어하고, 이는 대응하는 클록 신호를 생성하고, 대응하는 클록 신호를 GPU(205)에 분배한다. 일 구현예에서, 제어 유닛에 의해 선택된 성능 설정은 GPU(205)에 공급 전압을 제공하는 전압 레귤레이터(미도시)를 제어한다. 다른 구현예들에서, 특정 성능 설정에 도달하기 위해 제어 유닛으로부터 명령을 수신한 것에 응답하여 GPU(205)의 동작 지점 및/또는 전력 설정들을 변경하기 위한 다른 메커니즘들이 사용될 수 있다.

[0019] 이제 도 3을 참조하면, 큐 점유율(queue occupancy)에 기초하여 렌더링되고 있는 프레임들에 대한 성능 설정을 선택하는 일 구현예의 타이밍도가 도시된다. 비디오 시퀀스의 프레임들을 렌더링할 때, 일 구현예에서, 소프트웨어 드라이버는 큐 점유율에 적어도 부분적으로 기초하여 렌더링 하드웨어의 성능 설정을 변경한다. 큐 점유율은 렌더링되고 있는 현재 프레임에 대해 처리 하드웨어 (예를 들어, GPU)에 대해 인큐잉된 렌더링 태스크들의 수를 지칭한다.

[0020] 렌더링되는 각 프레임의 시작 및 마무리에 해당하는 비디오 동기화 신호(또는 VSync)에 의해 경계가 정해지는 프레임 주기가 도시된다. 도 3에 도시된 제1 프레임 주기에서, 초기 성능 설정(325)은 렌더링되고 있는 프레임에 대한 처리 하드웨어에 대해 설정된다. 초기 성능 설정(325)은 일 구현예에서 디폴트 설정일 수 있다. 다른 구현예에서, 초기 성능 설정 (325)은 애플리케이션의 유형, 애플리케이션에 의해 생성된 힌트들, 렌더링되고 있는 현재 프레임의 복잡성의 추정에 기초하여, 및/또는 다른 인자들에 기초하여 프로그래밍가능하다. 일 구현예에서, 성능 설정 제어를 담당하는 소프트웨어 드라이버는 렌더링 태스크 큐(들)의 큐 점유율을 모니터링한다. 소프트웨어 드라이버는 프레임 주기마다 여러 번 큐 점유율을 모니터링하며, 모니터링 빈도는 구현예에 따라 고정되거나 프로그램 가능할 수 있다. 도 3에 도시된 바와 같이, 제1 점유율 샘플(305)은 특정 수의 렌더링 태스

크들이 인큐되었음을 지정한다. 이 샘플(305)에 기초하여, 소프트웨어 드라이버는 현재 성능 설정(325)을 유지한다.

[0021] 다음 큐 점유율 샘플(310)은 이전 샘플(305)로부터의 감소이다. 이는 처리 하드웨어에 의해 하나 이상의 렌더링 태스크가 완료되어 렌더링 태스크의 수가 감소하였음을 나타낸다. 따라서, 샘플(310)로부터 샘플(305)로의 큐 점유율의 감소를 검출하는 것에 응답하여, 소프트웨어 드라이버는 성능 설정 (330)을 감소시켜 처리 하드웨어의 전력 소비를 감소시킨다. 이러한 경향은 다음 2개의 샘플(315 및 320)에 대해 계속되며, 소프트웨어 드라이버는 성능 설정(335 및 340)에 대한 전력을 각각 감소시킨다. 처리 하드웨어에 현재 프레임에 대해 완료할 렌더링 태스크가 더 적기 때문에 이러한 성능 설정 감소가 허용된다. 비디오 동기화 신호가 발생할 때, 현재 프레임은 디스플레이로 발송되거나, 네트워크를 통해 하나 이상의 클라이언트로 발송되거나, 다른 위치로 발송된다.

[0022] 다음 프레임 주기 동안, 제1 점유율 샘플(345)은 이 프레임에 대한 렌더링 태스크들이 상대적으로 더 적다는 것을 나타낸다. 따라서, 성능 설정(340)은 프레임 주기의 시작에서 처리 하드웨어에 대해 비교적 낮은 레벨로 유지될 수 있다. 다음 점유율 샘플(350)은 렌더링 태스크들의 수가 감소되었음을 나타내며, 더 낮은 성능 설정(370)을 허용한다. 그러나, 후속 점유율 샘플(355)은 큐 점유율이 증가되었음을 나타낸다. 이는 게임 시나리오에서의 플레이어 입력, 가상 현실 환경에서의 사용자 움직임, 또는 다른 유형의 애플리케이션들에서 생성된 다른 입력 또는 이벤트에 의해 야기될 수 있는 다수의 렌더링 태스크들을 수신한 것에 기인할 수 있다.

[0023] 소프트웨어 드라이버가 점유율 샘플(355)에 대한 점유율의 증가, 및 다음 비디오 동기화 신호까지 잔여 감소 시간을 검출할 때, 소프트웨어 드라이버는 성능 설정 (375)에 대한 처리 하드웨어에 제공되는 전력을 증가시킴으로써 응답한다. 일 구현예에서, 성능 설정(375)은 처리 하드웨어에 대한 최대 성능 설정이다. 다음 2개의 점유율 샘플들(360 및 365)은 현재 프레임에 대한 렌더링 태스크들의 수가 감소되었음을 나타낸다. 그러나, 이들 렌더링 태스크들을 완료하는데 이용가능한 시간이 또한 감소되었으며, 이는 소프트웨어 드라이버가 처리 하드웨어에 대한 비교적 고 성능 설정(375)을 유지할 것이라는 것을 의미한다.

[0024] 타이밍도(300)에 도시된 예들은 렌더링 태스크 큐(들)의 큐 점유율에 기초하여 성능 설정을 조정하는 소프트웨어 드라이버에 대한 하나의 특정 구현예를 나타낸다. 다른 구현예들에서, 소프트웨어 드라이버는 큐 점유율의 변화들에 기초하여 다른 유형들의 조정들을 행할 수 있다. 구현예에 따라 성능 설정에 대한 업데이트가 이루어지는 세분성이 달라질 수 있다는 것이 이해되어야 한다. 또한, 소프트웨어 드라이버가 큐 점유율을 체크하는 빈도도 구현예에 따라 달라질 수 있다.

[0025] 이제 도 4를 참조하면, 다수의 유입 태스크들 및 잔여 시간을 성능 설정에 매핑하기 위한 테이블(400)의 일 구현이 도시된다. 일 구현예에서, 제어 로직 또는 소프트웨어 드라이버는 대응하는 성능 설정을 검색하기 위해 테이블(400)의 열들(405 및 410)의 룩업을 수행한다. 검색된 성능 설정은 특정 동작 지점에서 동작하도록 복수의 처리 엘리먼트들(예를 들어, 도 2의 GPU(205))을 프로그래밍하는데 사용된다. 일 구현예에서, 열(405)는 다수의 유입 태스크(예를 들어, 렌더링 태스크)에 대한 상이한 가능한 값을 포함한다. 다른 구현예들에서, 열(405)는 현재 프레임을 렌더링하기 위해 수행될 필요가 있는 작업의 양을 나타내는 다른 값들을 포함한다. 예를 들어, 다른 구현예에서, 열(405)는 큐 점유율의 관점에서 측정된다. 다른 구현예들에서, 열(405)는 수신된 힌트들의 수, 검출된 이벤트들의 수, 또는 다른 방식으로 측정된다. 일 구현예에서, 열(410)은 다음 비디오 동기화 신호까지 잔여 상이한 시간량에 대한 엔트리들을 포함한다.

[0026] 일 구현예에서, 소프트웨어 드라이버는 다수의 렌더링 태스크들 및 다음 비디오 동기화 신호까지 잔여 시간량을 사용하여 테이블(400)의 룩업을 수행한다. 룩업에서 적중이 발생하는 경우 일치하는 엔트리에서 성능 설정이 검색된다. 룩업이 미스(miss)를 초래하면, 소프트웨어 드라이버는 2개의 가장 가까운 엔트리들에 기초하여 성능 설정 값을 보간할 수 있다. 특정 성능 설정을 검색 및/또는 계산한 후, 소프트웨어 드라이버는 렌더링 하드웨어가 특정 성능 설정에서 동작하도록 한다. 일 구현예에서, 소프트웨어 드라이버는 프레임 주기 동안 렌더링 태스크들의 수 및/또는 잔여 시간 변화량으로서 성능 설정을 업데이트하기 위해 테이블(400)에 대한 프레임당 다수의 룩업들을 수행한다.

[0027] 일 구현예에서, 시스템 상에서 실행될 수 있는 각각의 상이한 애플리케이션에 대한 별개의 테이블(400)이 존재한다. 예를 들어, 클라우드 게임 환경의 경우, 제1 테이블(400A)이 시스템에 의해 저장된다. 가상 현실 애플리케이션의 경우, 시스템에 의해 제2 테이블(400B)이 저장된다. 임의의 수의 다른 테이블들(400C-N)이 또한 상이한 애플리케이션들에 대해 시스템에 의해 저장될 수 있다. 각각의 애플리케이션은 애플리케이션을 실행할 때 수행될 가능성이 있는 렌더링 태스크들에 대해 상이한 특성들 및 복잡성을 가질 수 있다. 따라서, 각각의 애플리케이션은 렌더링 태스크들의 수 및 잔여 시간에 기초하여 사용되어야 하는 상이한 성능 설정들을 수용하기 위해

별개의 테이블(400)을 갖는다.

[0028] 일 구현예에서, 각각의 테이블(400)은 소프트웨어에 의해 프로그래밍된다. 테이블(400)은 테스트 데이터에 기초하여 프로그래밍될 수 있고/있거나 테이블(400)은 애플리케이션의 거동을 모니터링하는 것에 기초하여 실시간 트레이닝에 기초하여 프로그래밍될 수 있다. 예를 들어, 일 구현예에서, 테이블(400)은 주어진 애플리케이션에 대한 디폴트 값들을 갖는 소프트웨어에 의해 프로그래밍된다. 그 후, 런타임 동안, 소프트웨어는 테이블(400)에 대한 디폴트 값들을 생성하는 데 사용된 테스트 시나리오들과 비교하여 임의의 변화들이 런타임 환경에서 관찰되었는지를 알기 위해 주어진 애플리케이션을 모니터링할 수 있다. 렌더링 태스크가 예측된 것보다 더 오래 걸리거나, 렌더링 태스크가 예측된 것보다 더 빨리 종료되면, 성능 설정 열(415)에 저장된 값은 주어진 애플리케이션의 거동을 더 정확하게 반영하도록 업데이트될 수 있다. 다른 구현예에서, 성능 설정을 선택하기 위해 테이블(400)을 사용하는 대신, 소프트웨어 드라이버는 유입 태스크들의 수 및 잔여 시간에 기초하여 성능 설정을 계산하기 위한 공식을 사용한다. 다른 구현들에서, 소프트웨어 드라이버는 성능 설정을 선택하기 위한 다른 적절한 기술들을 사용한다.

[0029] 이제 도 5를 참조하면, 성능 보장 전력 관리(performance guaranteed power management)를 갖는 실시간 비디오 렌더링을 수행하기 위한 방법(500)의 일 구현예가 도시된다. 논의를 위해, 이 구현의 단계들 및 도 6 의 단계들이 순차적인 순서로 도시된다. 그러나, 설명된 방법들의 다양한 구현예들에서, 설명된 엘리먼트들 중 하나 이상은 도시된 것과 상이한 순서로 동시에 수행되거나, 또는 완전히 생략된다는 것에 유의한다. 다른 추가적인 엘리먼트들이 또한 원하는 대로 수행된다. 본 출원에 설명된 다양한 시스템들 또는 장치들 중 임의의 것은 방법(500)을 구현하도록 구성된다.

[0030] 소프트웨어 드라이버는 렌더링되고 있는 현재 프레임에 대한 렌더링 태스크들에 대응하는 입력들을 모니터링한다(블록 505). 일 구현예에서, 입력들은 네트워크 상의 사용자와 연관된 이벤트들이다. 예를 들어, 사용자는 클라우드 게임 시나리오에서 비디오 게임을 플레이하고 있다. 다른 구현예에서, 입력들은 가상 현실 환경에서의 사용자 움직임들이다. 다른 구현예들에서, 다른 유형들의 시나리오들에 대한 다른 유형들의 입력들이 블록 505에서 수신된다. 또한, 소프트웨어 드라이버는 다음 비디오 동기화 신호까지 현재 프레임 주기에 잔여 시간량을 모니터링한다(블록 510). 다음으로, 소프트웨어 드라이버는 다음 비디오 동기화 신호까지 잔여 시간량 내에 유입되는 렌더링 태스크들을 완료하기 위한 최저 가능한 성능 설정을 결정한다(블록 515). 일 구현예에서, 처리 하드웨어 (예를 들어, GPU)에 대한 성능 설정은 대응하는 전압 및 주파수 값들을 포함한다.

[0031] 만약 소프트웨어 드라이버가 최대 성능 설정에서 조차도 다음 비디오 동기화 신호까지 잔여 시간량에 유입되는 렌더링 태스크를 완료할 수 없다고 결정한다면(조건부 블록 (520) "아니오" 레그), 소프트웨어 드라이버는 이전 프레임이 재생되게 하고 처리 하드웨어가 유휴 상태(idle)로 설정되게 하거나 상대적으로 저 성능 상태(예를 들어, 최저 성능 설정)로 설정되게 한다(블록 525). 대안적으로, 소프트웨어 드라이버는 현재 프레임이 이전 프레임을 재생하기보다는 블록(525)에서 지연되게 할 수 있다. 유입되는 렌더링 태스크가 다음 Vsync까지 잔여 시간에 완료될 수 있도록 하는 성능 설정이 있는 경우(조건부 블록 (520), "예" 레그), 소프트웨어 드라이버는 처리 하드웨어가 주어진 성능 설정에서 동작하게 한다(블록 530). 일 구현예에서, 주어진 성능 설정은 다음 비디오 동기화 신호까지 잔여 시간량에서 유입 렌더링 태스크들을 완료하기 위한 최저 가능한 성능 설정이다. 다른 구현예에서, 주어진 성능 설정은 다음 비디오 동기화 신호까지 잔여 시간량에서 유입 렌더링 태스크들을 완료하기 위한 에러의 마진을 제공하기 위해 최저 가능한 성능 설정보다 높은 하나의 설정이다. 다른 구현예들에서, 에러의 마진은 최저 가능한 성능 설정보다 더 높은 둘 이상의 설정들로 증가될 수 있다.

[0032] 현재 프레임의 렌더링이 종료되지 않은 경우(조건부 블록(540), "아니오" 레그), 일정 시간 경과 후 또는 일부 이벤트(즉, 큐 점유율의 변화)가 검출된 후, 방법(500)은 블록 (505)으로 복귀한다. 성능 설정이 너무 자주 변경되는 것을 방지하기 위해 일부 히스테리시스가 루프에 추가될 수 있음에 유의한다. 현재 프레임의 렌더링이 완료되면(조건부 블록(540), "예" 레그), 방법(500)이 종료된다. 방법 500 은 렌더링되고 있는 비디오 시퀀스의 각각의 비디오 프레임에 대해 수행될 수 있다는 것에 유의한다.

[0033] 이제 도 6을 참조하면, 애플리케이션 유형에 기초하여 하드웨어를 처리하기 위한 성능 설정을 제어하기 위한 방법(600)의 일 구현예가 도시된다. 제어 유닛은 어느 애플리케이션이 시스템에서 현재 실행 중인지를 결정한다(블록 605). 그런 다음, 제어 유닛은 해당 애플리케이션에 대응되는 성능 설정 룩업 테이블(도 4의 400)을 로딩한다(블록 610). 다음으로, 제어 유닛은 테이블을 사용하여 큐 점유율 및 다음 비디오 동기화 신호까지 잔여 시간량에 기초하여 처리 하드웨어에 대한 성능 설정을 선택한다(블록 615). 제어 유닛이 시스템에 의해 실행되는 상이한 애플리케이션을 검출하면(조건부 블록(620), "예" 레그), 방법(600)은 블록(610)으로 복귀한다. 그렇지

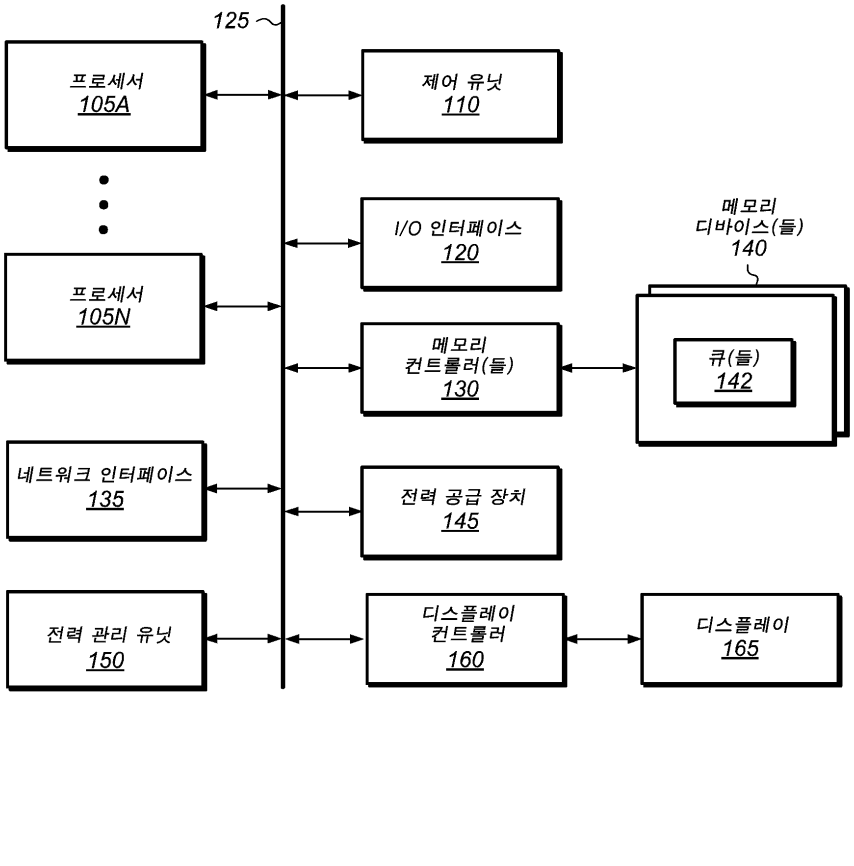
않고, 주어진 애플리케이션이 시스템에 의해 계속 실행되는 경우(조건부 블록(620), "아니오" 레그), 방법(600)은 블록(615)으로 복귀한다. 일부 경우들에서, 단일 애플리케이션은 다수의 상이한 성능 설정 룩업 테이블들을 가질 수 있다는 것에 유의한다. 예를 들어, 비디오 게임 애플리케이션은 상이한 양의 렌더링 복잡성을 갖는 상이한 장면들을 가질 수 있다. 애플리케이션의 제1 장면에 대해, 제어 유닛은 제1 테이블을 로딩할 수 있고, 제2 장면에 대해, 제어 유닛은 제2 테이블을 로딩하는 것 등이다.

[0034] 다양한 구현예들에서, 소프트웨어 애플리케이션의 프로그램 명령어들은 본 출원에 설명된 방법들 및/또는 메커니즘들을 구현하기 위해 사용된다. 예를 들어, 범용 또는 특수 목적 프로세서에 의해 실행 가능한 프로그램 명령어들이 고려된다. 다양한 구현예들에서, 이러한 프로그램 명령어들은 하이 레벨 프로그래밍 언어로 표현될 수 있다. 다른 구현예들에서, 프로그램 명령어들은 하이 레벨 프로그래밍 언어로부터 이진, 중간, 또는 다른 형태로 컴파일될 수 있다. 대안적으로, 하드웨어의 동작 또는 설계를 기술하는 프로그램 명령어들이 기록될 수 있다. 이러한 프로그램 명령어들은 C와 같은 하이 레벨 프로그래밍 언어로 표현될 수 있다. 대안적으로, 베릴로그(Verilog)와 같은 하드웨어 설계 언어(HDL)를 사용할 수 있다.다양한 구현예들에서, 프로그램 명령어들은 다양한 비 일시적 컴퓨터 판독가능 저장 매체들 중 임의의 매체 상에 저장된다. 저장 매체는 프로그램 실행을 위해 프로그램 명령어들을 컴퓨팅 시스템에 제공하기 위해 사용 동안 컴퓨팅 시스템에 의해 액세스가능하다. 일반적으로 말하면, 이러한 컴퓨팅 시스템은 적어도 하나 이상의 메모리 및 프로그램 명령어를 실행하도록 구성된 하나 이상의 프로세서를 포함한다.

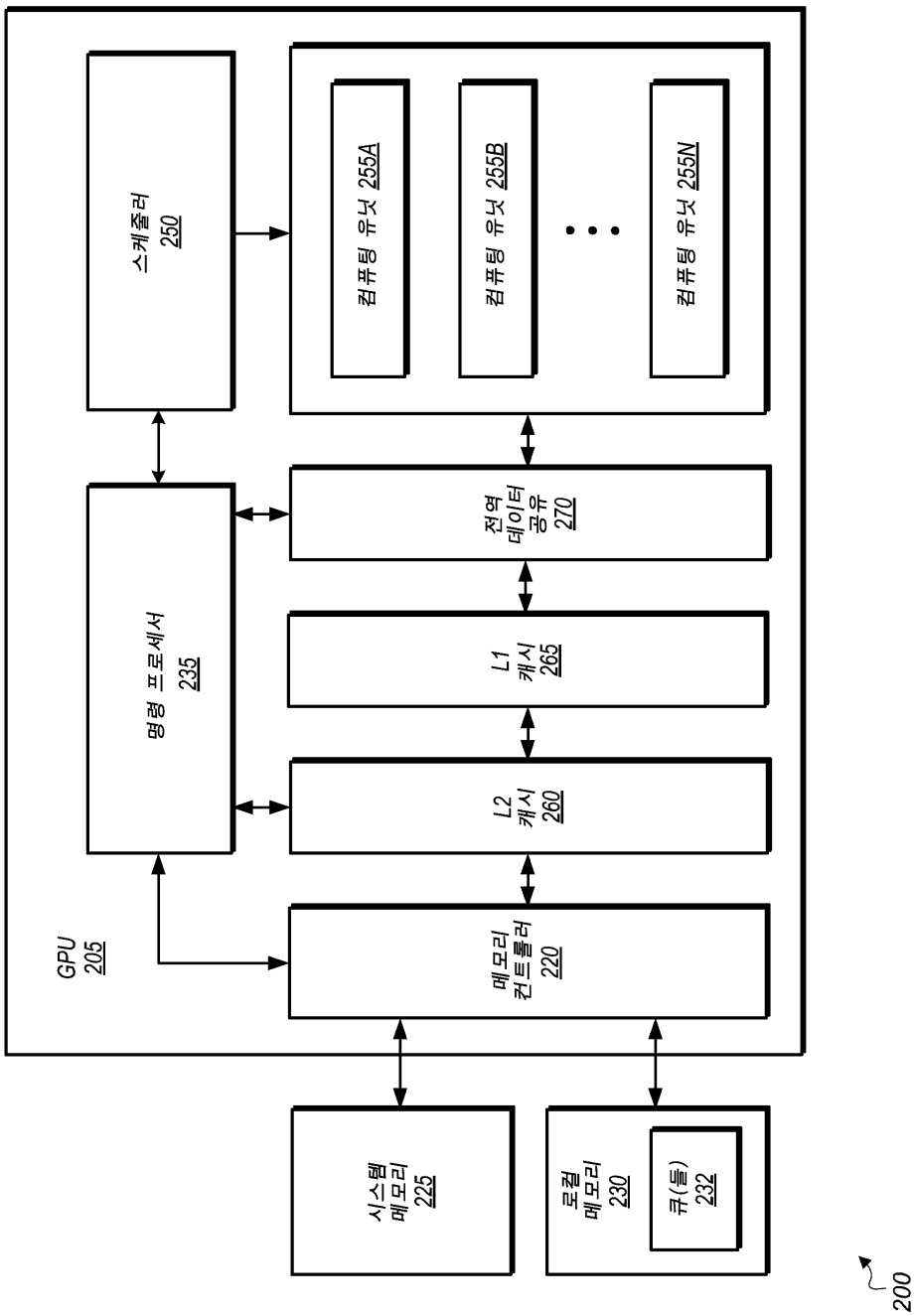
[0035] 상술한 구현들은 단지 구현들의 비제한적인 예들일 뿐이라는 것이 강조되어야 한다. 상기 개시가 충분히 인식되면, 다수의 변형 및 수정이 당업자에게 명백해질 것이다. 이하의 청구항들은 이러한 모든 변형 및 수정을 포함하는 것으로 해석되어야 한다.

도면

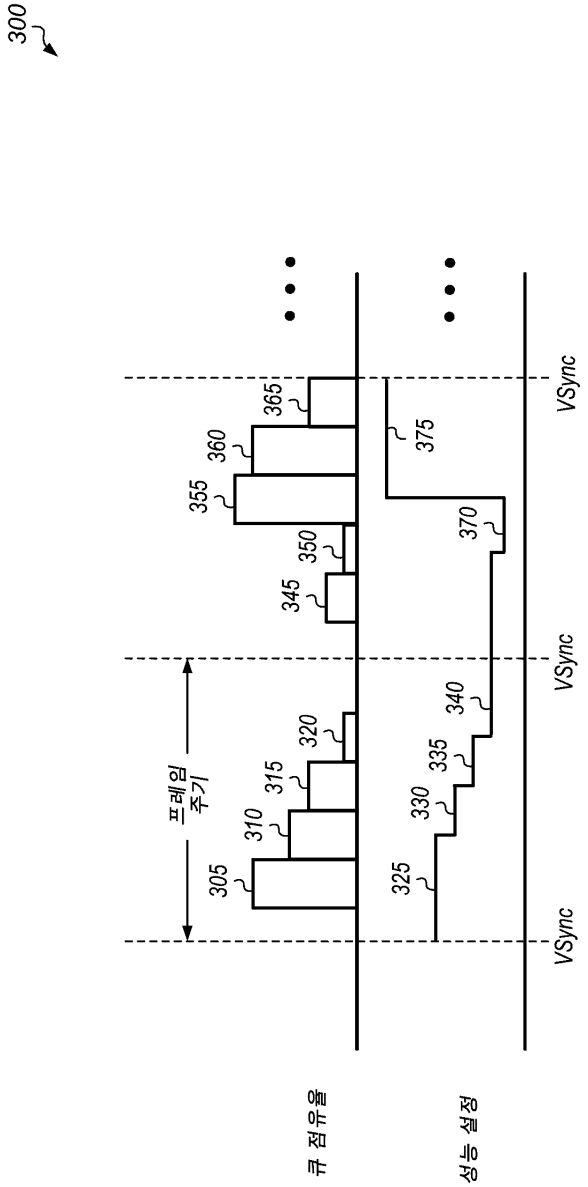
도면1



도면2


GPU
205
스케줄러
250
컴퓨팅 유닛 255A
컴퓨팅 유닛 255B
컴퓨팅 유닛 255N
명령 프로세서
235
전역
데이터
공유
270
L1
캐시
265
L2
캐시
260
메모리
컨트롤러
220
시스템
메모리
225
로컬
메모리
230
큐(들)
232
200

도면3

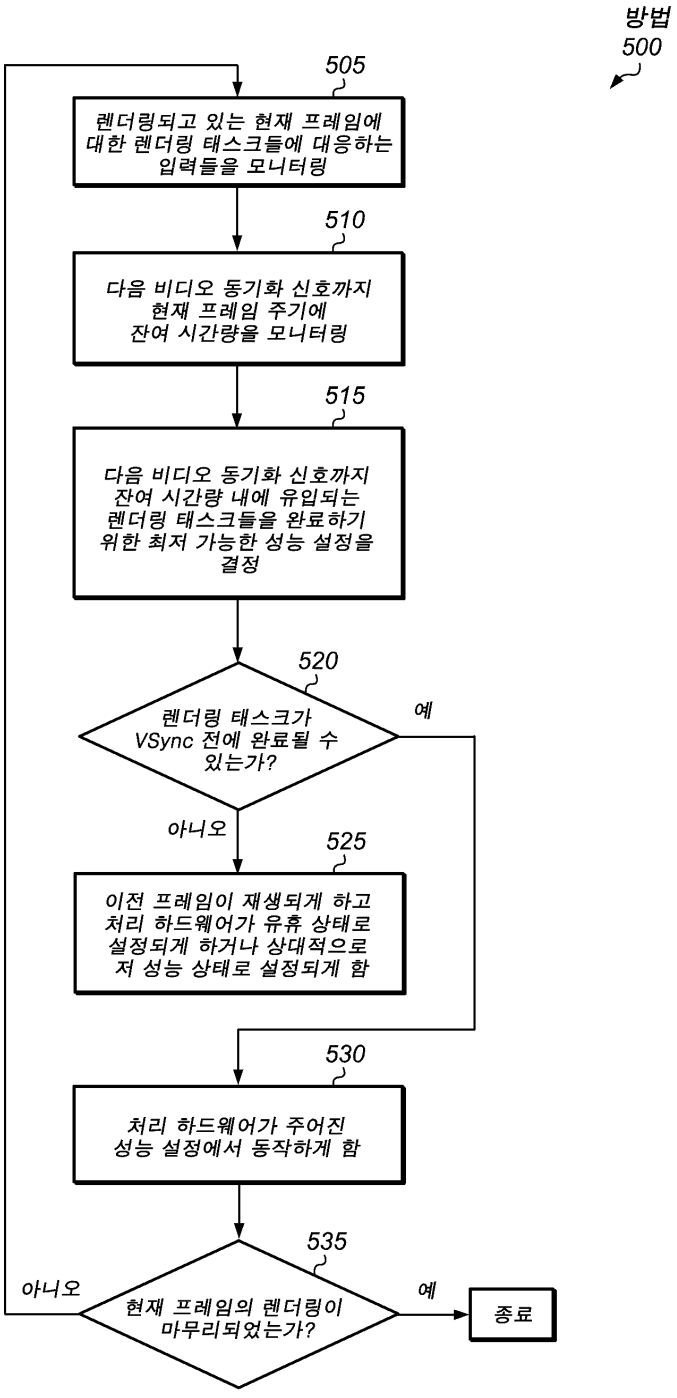

300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
VSync
VSync
VSync
프레임 주기
큐 점유율
성능 설정

도면4

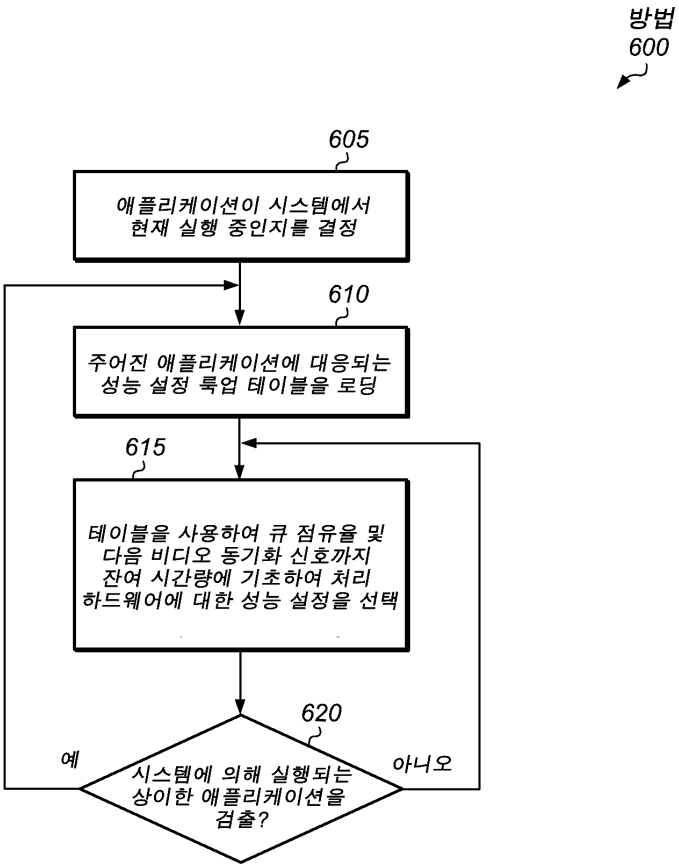
405 렌더링 태스크의 수	410 VSync 까지 잔여 시간	415 성능 설정
1	10 ms	3
2	10 ms	4
3	10 ms	4.5
4	10 ms	5
5	10 ms	6
1	9 ms	3.5
2	9 ms	4.5
⋮	⋮	⋮

400

도면5


방법
500
505
렌더링되고 있는 현재 프레임에 대한 렌더링 태스크들에 대응하는 입력들을 모니터링
510
다음 비디오 동기화 신호까지 현재 프레임 주기에 잔여 시간량을 모니터링
515
다음 비디오 동기화 신호까지 잔여 시간량 내에 유입되는 렌더링 태스크들을 완료하기 위한 최저 가능한 성능 설정을 결정
520
렌더링 태스크가 VSync 전에 완료될 수 있는가?
예
아니오
525
이전 프레임이 재생되게 하고 처리 하드웨어가 유휴 상태로 설정되게 하거나 상대적으로 저 성능 상태로 설정되게 함
530
처리 하드웨어가 주어진 성능 설정에서 동작하게 함
535
현재 프레임의 렌더링이 마무리되었는가?
아니오
예
종료

도면6


방법
600
605
애플리케이션이 시스템에서
현재 실행 중인지를 결정
610
주어진 애플리케이션에 대응되는
성능 설정 룩업 테이블을 로딩
615
테이블을 사용하여 큐 점유율 및
다음 비디오 동기화 신호까지
잔여 시간량에 기초하여 처리
하드웨어에 대한 성능 설정을 선택
620
예
시스템에 의해 실행되는
상이한 애플리케이션을
검출?
아니오