(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년12월28일
(11) 등록번호 10-1813429
(24) 등록일자 2017년12월21일

(51) 국제특허분류(Int. Cl.)
G06T 15/00 (2006.01) *G06T 1/20* (2006.01)
G06T 1/60 (2006.01) *G06T 15/80* (2011.01)
(52) CPC특허분류
G06T 15/005 (2013.01)
G06T 1/20 (2013.01)
(21) 출원번호 10-2016-7023022
(22) 출원일자(국제) 2015년01월26일
심사청구일자 2017년07월10일
(85) 번역문제출일자 2016년08월23일
(65) 공개번호 10-2016-0123311
(43) 공개일자 2016년10월25일
(86) 국제출원번호 PCT/US2015/012917
(87) 국제공개번호 WO 2015/126574
국제공개일자 2015년08월27일
(30) 우선권주장
14/182,976 2014년02월18일 미국(US)
(56) 선행기술조사문헌
WO2013112692 A1*
US20130155080 A1
US20060070079 A1
US20030050725 A1
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
메이 춘휘
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
고엘 비니트
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
김 동현
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(74) 대리인
특허법인코리아나

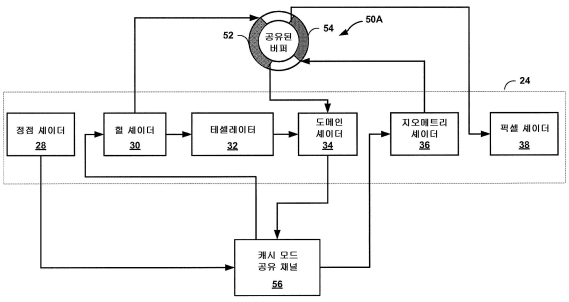
전체 청구항 수 : 총 18 항 심사관 : 조우연

(54) 발명의 명칭 **공유 데이터 채널들을 가지는 셰이더 파이프라인**

(57) 요 약

그래픽스 프로세싱 유닛 (GPU) 은 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들에 의해 공유되는 GPU 의 온칩 그래픽스 메모리에서 공유 데이터 채널을 할당할 수도 있다. GPU 에서의 셰이더 유닛들은 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들을 실행할 수도 있다. GPU 는 온칩 그래픽스 메모리에서의 공유 데이터 채널에, 셰이더 유닛들 상에서 실행하는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들의 각각에 의해 발생된 데이터를 저장할 수도 있다.

대 표 도



(52) CPC특허분류
G06T 1/60 (2013.01)
G06T 15/80 (2013.01)

명 세 서

청구범위

청구항 1

그래픽스 프로세싱의 방법으로서,

그래픽스 프로세싱 유닛 (GPU) 에 의해, 상기 GPU 의 온칩 그래픽스 메모리에서 제 1 공유 데이터 채널로서 제 1 링 버퍼를 할당하는 단계로서, 상기 제 1 링 버퍼는 그래픽스 프로세싱 파이프라인의 각각의 제 1 의 2개의 스테이지들에 의해 발생된 제 1 및 제 2 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 공유되는, 상기 제 1 링 버퍼를 할당하는 단계;

상기 GPU 에 의해, 상기 그래픽스 프로세싱 파이프라인의 각각의 제 2 의 2개의 스테이지들에 의해 발생된 제 3 및 제 4 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 공유되는 상기 GPU 의 상기 온칩 그래픽스 메모리에서 제 2 공유 데이터 채널로서 제 2 링 버퍼를 할당하는 단계;

상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 하나의 스테이지에 의해, 상기 제 1 링 버퍼에 저장되는 상기 제 1 데이터를 발생시키기 위해 상기 제 2 링 버퍼에 저장된 상기 제 3 데이터를 소비하는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들을 상기 GPU 에서의 셰이더 유닛들 상에서 실행하는 단계;

상기 제 1 링 버퍼와 상기 제 2 링 버퍼 사이의 교착상태 (deadlock) 를 방지하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 저장하기 위한 상기 제 1 링 버퍼에서의 자유 공간 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들 중 하나의 스테이지에 의해 발생된 상기 제 4 데이터를 저장하기 위한 상기 제 2 링 버퍼에서의 자유 공간을 상기 GPU 에 의해 예비하는 단계;

상기 GPU 에 의해 온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼에, 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터를 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터의 큐들로서 저장하는 단계; 및

상기 제 1 링 버퍼로부터 판독되는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 상기 링 버퍼로부터 삭제하여, 상기 GPU 가 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 하나의 스테이지에 의해 발생된 추가적인 데이터를 저장하게 하기 위해 상기 제 1 링 버퍼에서의 공간을 증가시키는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를, 상기 GPU 에 의해 온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼로부터 판독하는 단계

를 포함하는, 그래픽스 프로세싱의 방법.

청구항 2

제 1 항에 있어서,

데이터가 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 하나 이상의 스테이지들에 의해 소비될 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하고, 그리고 자유 공간이 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들에 의해 발생된 데이터를 저장하기 위해 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하도록, 상기 GPU 에 의해, 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼의 상태에 적어도 부분적으로 기초하여 상기 셰이더 유닛들 상에서의 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들의 실행을 스케쥴링하는 단계를 더 포함하는, 그래픽스 프로세싱의 방법.

청구항 3

제 1 항에 있어서,

상기 제 2 링 버퍼는 상기 제 2 링 버퍼에 저장된 데이터를 캐시하기 위해 캐시 모드에서 동작하며, 상기 제 1 링 버퍼는 선입선출 (FIFO) 모드에서 동작하는, 그래픽스 프로세싱의 방법.

청구항 4

제 1 항에 있어서,

상기 제 1 링 버퍼에 저장된 상기 제 1 및 제 2 데이터는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 정점 데이터를 포함하며, 상기 제 2 링 버퍼에 저장된 상기 제 3 및 제 4 데이터는 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 발생된 프리미티브들을 포함하는, 그래픽스 프로세싱의 방법.

청구항 5

제 4 항에 있어서,

상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들은 정점 셰이더 및 도메인 셰이더를 포함하는, 그래픽스 프로세싱의 방법.

청구항 6

제 4 항에 있어서,

상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들은 헐 셰이더 및 지오메트리 셰이더를 포함하는, 그래픽스 프로세싱의 방법.

청구항 7

그래픽스 데이터를 프로세싱하도록 구성된 장치로서,

메모리; 및

그래픽스 프로세싱 유닛 (GPU)

을 포함하고, 상기 GPU 는,

상기 GPU 의 온칩 그래픽스 메모리에서 제 1 공유 데이터 채널로서 제 1 링 버퍼를 할당하는 것으로서, 상기 제 1 링 버퍼는 그래픽스 프로세싱 파이프라인의 각각의 제 1 의 2개의 스테이지들에 의해 발생된 제 1 및 제 2 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 공유되는, 상기 제 1 링 버퍼를 할당하고;

상기 그래픽스 프로세싱 파이프라인의 각각의 제 2 의 2개의 스테이지들에 의해 발생된 제 3 및 제 4 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 공유되는 상기 GPU 의 상기 온칩 그래픽스 메모리에서 제 2 공유 데이터 채널로서 제 2 링 버퍼를 할당하고;

상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 하나의 스테이지에 의해, 상기 제 1 링 버퍼에 저장되는 상기 제 1 데이터를 발생시키기 위해 상기 제 2 링 버퍼에 저장된 상기 제 3 데이터를 소비하는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들을 상기 GPU 에서의 셰이더 유닛들 상에서 실행하고;

상기 제 1 링 버퍼와 상기 제 2 링 버퍼 사이의 교착상태를 방지하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 저장하기 위한 상기 제 1 링 버퍼에서의 자유 공간 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들 중 하나의 스테이지에 의해 발생된 상기 제 4 데이터를 저장하기 위한 상기 제 2 링 버퍼에서의 자유 공간을 예비하고;

온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼에, 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터를 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터의 큐들로서 저장하고; 그리고

상기 제 1 링 버퍼로부터 판독되는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 상기 링 버퍼로부터 삭제하여, 상기 GPU 가 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 하나의 스테이지에 의해 발생된 추가적인 데이터를 저장하게 하기 위해 상기 제 1 링 버퍼에서의 공간을 증가시키는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를, 온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼로부터 판독하도록

구성되는, 그래픽스 데이터를 프로세싱하도록 구성된 장치.

청구항 8

제 7 항에 있어서,

상기 GPU 는 또한,

데이터가 상기 그래픽스 프로세싱 파이프라인의 하나 이상의 스테이지들에 의해 소비될 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하고, 그리고 자유 공간이 셰이더 프로세서들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들에 의해 발생된 데이터를 저장하기 위해 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하도록, 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼의 상태에 적어도 부분적으로 기초하여 상기 셰이더 프로세서들 상에서의 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들의 실행을 스케쥴링하도록

구성되는, 그래픽스 데이터를 프로세싱하도록 구성된 장치.

청구항 9

제 7 항에 있어서,

상기 제 2 링 버퍼는 상기 제 2 링 버퍼에 저장된 데이터를 캐시하기 위해 캐시 모드에서 동작하며, 상기 제 1 링 버퍼는 선입선출 (FIFO) 모드에서 동작하는, 그래픽스 데이터를 프로세싱하도록 구성된 장치.

청구항 10

제 7 항에 있어서,

상기 제 1 링 버퍼에 저장된 상기 데이터는 셰이더 프로세서들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 정점 데이터를 포함하며, 상기 제 2 링 버퍼에 저장된 상기 제 3 및 제 4 데이터는 상기 셰이더 프로세서들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 발생된 프리미티브들을 포함하는, 그래픽스 데이터를 프로세싱하도록 구성된 장치.

청구항 11

그래픽스 프로세싱 유닛 (GPU) 의 온칩 그래픽스 메모리에서 제 1 공유 데이터 채널로서 제 1 링 버퍼를 할당하는 수단으로서, 상기 제 1 링 버퍼는 그래픽스 프로세싱 파이프라인의 각각의 제 1 의 2개의 스테이지들에 의해 발생된 제 1 및 제 2 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 공유되는, 상기 제 1 링 버퍼를 할당하는 수단;

상기 그래픽스 프로세싱 파이프라인의 각각의 제 2 의 2개의 스테이지들에 의해 발생된 제 3 및 제 4 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 공유되는 상기 GPU 의 상기 온칩 그래픽스 메모리에서 제 2 공유 데이터 채널로서 제 2 링 버퍼를 할당하는 수단;

상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 하나의 스테이지에 의해, 상기 제 1 링 버퍼에 저장되는 상기 제 1 데이터를 발생시키기 위해 상기 제 2 링 버퍼에 저장된 상기 제 3 데이터를 소비

하는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들을 실행하는 수단;

상기 제 1 링 버퍼와 상기 제 2 링 버퍼 사이의 교착상태를 방지하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 저장하기 위한 상기 제 1 링 버퍼에서의 자유 공간 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들 중 하나의 스테이지에 의해 발생된 상기 제 4 데이터를 저장하기 위한 상기 제 2 링 버퍼에서의 자유 공간을 예비하는 수단;

온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼에, 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터를 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터의 큐들로서 저장하는 수단; 및

상기 제 1 링 버퍼로부터 판독되는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 상기 링 버퍼로부터 삭제하여, 상기 GPU 가 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 하나의 스테이지에 의해 발생된 추가적인 데이터를 저장하게 하기 위해 상기 제 1 링 버퍼에서의 공간을 증가시키는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를, 온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼로부터 판독하는 수단

을 포함하는, 장치.

청구항 12

제 11 항에 있어서,

데이터가 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 하나 이상의 스테이지들에 의해 소비될 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하고, 그리고 자유 공간이 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들에 의해 발생된 데이터를 저장하기 위해 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하도록, 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼의 상태에 적어도 부분적으로 기초하여 상기 셰이더 유닛들 상에서의 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들의 실행을 스케줄링하는 수단을 더 포함하는, 장치.

청구항 13

제 11 항에 있어서,

상기 제 2 링 버퍼는 상기 제 2 링 버퍼에 저장된 데이터를 캐시하기 위해 캐시 모드에서 동작하며, 상기 제 1 링 버퍼는 선입선출 (FIFO) 모드에서 동작하는, 장치.

청구항 14

제 11 항에 있어서,

상기 제 1 링 버퍼에 저장된 상기 제 1 및 제 2 데이터는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 정점 데이터를 포함하며, 상기 제 2 링 버퍼에 저장된 상기 제 3 및 제 4 데이터는 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 발생된 프리미티브들을 포함하는, 장치.

청구항 15

명령들을 저장하는 비일시적 컴퓨터 판독가능 저장 매체로서,

상기 명령들은, 실행될 때, 하나 이상의 프로그래밍가능 프로세서들로 하여금,

온칩 그래픽스 메모리에서 제 1 공유 데이터 채널로서 제 1 링 버퍼를 할당하도록 하는 것으로서, 상기 제 1 링 버퍼는 그래픽스 프로세싱 파이프라인의 각각의 제 1 의 2개의 스테이지들에 의해 발생된 제 1 및 제 2 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 공유되는, 상기

제 1 링 버퍼를 할당하도록 하고;

상기 그래픽스 프로세싱 파이프라인의 각각의 제 2 의 2개의 스테이지들에 의해 발생된 제 3 및 제 4 데이터를 저장하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 공유되는 상기 온칩 그래픽스 메모리에서 제 2 공유 데이터 채널로서 제 2 링 버퍼를 할당하도록 하고;

상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 하나의 스테이지에 의해, 상기 제 1 링 버퍼에 저장되는 상기 제 1 데이터를 발생시키기 위해 상기 제 2 링 버퍼에 저장된 상기 제 3 데이터를 소비하는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들을 셰이더 유닛들 상에서 실행하도록 하고;

상기 제 1 링 버퍼와 상기 제 2 링 버퍼 사이의 교착상태를 방지하기 위해 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 저장하기 위한 상기 제 1 링 버퍼에서의 자유 공간 및 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들 중 하나의 스테이지에 의해 발생된 상기 제 4 데이터를 저장하기 위한 상기 제 2 링 버퍼에서의 자유 공간을 예비하도록 하고;

온칩 그래픽스 메모리에서의 상기 링 버퍼에, 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터를 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 상기 제 1 및 제 2 데이터의 큐들로서 저장하도록 하고; 그리고

상기 제 1 링 버퍼로부터 판독되는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를 상기 링 버퍼로부터 삭제하여, 상기 하나 이상의 프로그래밍가능 프로세서들이 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 하나의 스테이지에 의해 발생된 추가적인 데이터를 저장하게 하기 위해 상기 제 1 링 버퍼에서의 공간을 증가시키는 것을 포함하는, 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들 중 상기 제 2 스테이지에 의해 발생된 상기 제 2 데이터를, 온칩 그래픽스 메모리에서의 상기 제 1 링 버퍼로부터 판독하도록 하는, 비일시적 컴퓨터 판독가능 저장 매체.

청구항 16

제 15 항에 있어서,

상기 명령들은, 실행될 때, 추가로, 상기 하나 이상의 프로그래밍가능 프로세서들로 하여금,

데이터가 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 하나 이상의 스테이지들에 의해 소비될 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하고, 그리고 자유 공간이 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들에 의해 발생된 데이터를 저장하기 위해 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼에서 이용가능하도록, 상기 제 1 링 버퍼 또는 상기 제 2 링 버퍼의 상태에 적어도 부분적으로 기초하여 상기 셰이더 유닛들 상에서의 상기 그래픽스 프로세싱 파이프라인의 상기 하나 이상의 스테이지들의 실행을 스케쥴링하도록 하는, 비일시적 컴퓨터 판독가능 저장 매체.

청구항 17

제 15 항에 있어서,

상기 제 2 링 버퍼는 상기 제 2 링 버퍼에 저장된 데이터를 캐시하기 위해 캐시 모드에서 동작하며, 상기 제 1 링 버퍼는 선입선출 (FIFO) 모드에서 동작하는, 비일시적 컴퓨터 판독가능 저장 매체.

청구항 18

제 15 항에 있어서,

상기 제 1 링 버퍼에 저장된 상기 제 1 및 제 2 데이터는 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 1 의 2개의 스테이지들에 의해 발생된 정점 데이터를 포함하며, 상기 제 2 링 버퍼에 저장된 상기 제 3 및 제 4 데이터는 상기 셰이더 유닛들 상에서 실행하는 상기 그래픽스 프로세싱 파이프라인의 상기 제 2 의 2개의 스테이지들에 의해 발생된 프리미티브들을 포함하는, 비일시적 컴퓨터 판독가능

저장 매체.

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

청구항 23

삭제

청구항 24

삭제

청구항 25

삭제

청구항 26

삭제

청구항 27

삭제

청구항 28

삭제

청구항 29

삭제

청구항 30

삭제

발명의 설명

기 술 분 야

[0001] 본 개시물은 그래픽스 프로세싱 파이프라인의 스테이지들에 의해 발생되어 소비되는 데이터를 저장하는 공유 데이터 채널들에 관한 것이다.

배 경 기 술

[0002] 컴퓨팅 디바이스의 그래픽스 프로세싱 유닛 (GPU) 은 3차원 장면의 2차원 표현을 렌더링하기 위해 그래픽스 지령들을 프로세싱하는 복수의 스테이지들을 포함하는 그래픽스 프로세싱 파이프라인을 실행할 수 있다. 3차원 장면은 정점들로 일반적으로 이루어지며, 그래픽스 프로세싱 파이프라인은 3차원 장면의 2차원 표현을 렌더링하기 위해 3차원 장면에서의 각각의 정점에 대해, 고정된 순서로 실행되는 스테이지들의 시리즈를 포함한다.

[0003] 그래픽스 프로세싱 파이프라인은 3차원 장면의 정점들을 변환하도록 실행하는 셰이더 스테이지들의 체인을 포함할 수도 있다. 셰이더 스테이지들의 각각은 이전 스테이지들에 의해 발생되는 데이터를 소비하고 다음 스테이지들에 대한 데이터를 발생시킨다. 셰이더 스테이지들의 체인을 통해서 흐르는 막대한 데이터의 양 때문에, 셰이더 스테이지들의 체인에 대한 데이터가 관리되는 방법이 GPU 의 성능 및 메모리 효율에 영향을 미칠 수 있다.

발명의 내용

과제의 해결 수단

[0004] 본 개시물의 일 예에서, 그래픽스 프로세싱을 위한 방법은 그래픽스 프로세싱 유닛 (GPU) 에 의해, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들에 의해 공유되는 GPU 의 온칩 그래픽스 메모리에서 공유 데이터 채널을 할당하는 단계를 포함할 수도 있다. 본 방법은 GPU 에서의 셰이더 유닛들 상에서, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들을 실행하는 단계를 더 포함할 수도 있다. 본 방법은 GPU 에 의해 온칩 그래픽스 메모리에서의 공유 데이터 채널에, 셰이더 유닛들 상에서 실행하는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들의 각각에 의해 발생된 데이터를 저장하는 단계를 더 포함할 수도 있다.

[0005] 본 개시물의 다른 예에서, 그래픽스 프로세싱을 위한 장치는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들에 의해 공유되는 GPU 의 온칩 그래픽스 메모리에서 공유 데이터 채널을 할당하고; GPU 에서의 셰이더 유닛들 상에서, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들을 실행하고; 그리고 온칩 그래픽스 메모리에서의 공유 데이터 채널에, 셰이더 유닛들 상에서 실행하는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들의 각각에 의해 발생된 데이터를 저장하도록 구성된 그래픽스 프로세싱 유닛 (GPU) 을 포함할 수도 있다.

[0006] 본 개시물의 다른 예에서, 그래픽스 프로세싱을 위한 그래픽스용 장치는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들에 의해 공유되는 그래픽스 프로세싱 유닛 (GPU) 의 온칩 그래픽스 메모리에서 공유 데이터 채널을 할당하는 수단을 포함할 수도 있다. 본 장치는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들을 실행하는 수단을 더 포함할 수도 있다. 본 장치는 온칩 그래픽스 메모리에서의 공유 데이터 채널에, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들의 각각의 실행에 의해 발생되는 데이터를 저장하는 수단을 더 포함할 수도 있다.

[0007] 본 개시물의 다른 예에서, 컴퓨터 판독가능 저장 매체는, 실행될 때, 하나 이상의 프로그래밍가능 프로세서들로 하여금, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들에 의해 공유되는 온칩 그래픽스 메모리에서 공유 데이터 채널을 할당하도록 하고; 셰이더 유닛들 상에서, 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들을 실행하도록 하고; 그리고, 온칩 그래픽스 메모리에서의 공유 데이터 채널에, 셰이더 유닛들 상에서 실행하는 그래픽스 프로세싱 파이프라인의 적어도 2개의 스테이지들의 각각에 의해 발생된 데이터를 저장하도록 하는 명령들을 저장할 수도 있다.

[0008] 하나 이상의 예들의 세부 사항들이 첨부도면 및 아래의 상세한 설명에서 개시된다. 다른 특성들, 목적들, 및 이점들은 설명 및 도면들로부터, 그리고 청구항들로부터 명백히 알 수 있을 것이다.

도면의 간단한 설명

[0009] 도 1 은 다수의 그래픽스 파이프라인 스테이지들 간에 데이터 채널들을 공유하는 본 개시물의 하나 이상의 양태들을 구현하도록 구성될 수도 있는 예시적인 컴퓨팅 디바이스를 예시하는 블록도이다.

도 2 는 3차원 장면의 2차원 표현을 생성하기 위해 GPU 에 의해 수행될 수도 있는 예시적인 그래픽스 프로세싱 파이프라인을 예시하는 블록도이다.

도 3 은 도 1 의 CPU, GPU, 및 시스템 메모리의 예시적인 구현예들을 좀더 상세히 예시하는 블록도이다.

도 4 는 그래픽스 프로세싱 파이프라인에 사용되고 있는 공유 데이터 채널들의 일 예를 예시하는 블록도이다.

도 5 는 도 4 의 캐시 모드 공유 채널을 좀더 상세히 예시하는 블록도이다.

도 6 은 그래픽스 프로세싱 파이프라인의 스테이지들에 의해 데이터 채널들을 공유하는 예시적인 프로세스를 예시하는 플로우차트이다.

발명을 실시하기 위한 구체적인 내용

[0010] 일반적으로, 본 개시물은 공유 데이터 채널들을 가지는 제작자-소비자 모델을 이용하는 단일 패스 셰이더 파이프라인을 위한 기법들을 기술한다. 컴퓨팅 디바이스에서의 그래픽스 프로세싱 유닛 (GPU) 은 셰이더 파이프라인의 다수의 스테이지들을 GPU 상에서 동시에 실행할 수도 있는 셰이더 유닛들 상에서 셰이더 파이프라인을 실행할 수 있다. GPU 의 온칩 메모리에 저장된 데이터가 일반적으로 컴퓨팅 디바이스의 시스템 메모리에 저장된 데이터보다 더 빠르고 효율적으로 액세스될 수 있기 때문에, GPU 의 온칩 메모리에서의 데이터 채널들로부터의 데이터를 소비함으로써 그리고 GPU 의 온칩 메모리에서의 데이터 채널들에 또한 저장되는 데이터를 발생시킴으로써, GPU 에서의 셰이더 유닛들의 효율이 증가될 수 있다.

[0011] 일 예에서, GPU 는 GPU 에서의 셰이더 유닛들에 의해 셰이더 파이프라인의 실행에 의해 소비되고 발생되는 데이터를 저장하기 위해 동일한 사이즈의 데이터 채널들을 할당할 수도 있다. 그러나, GPU 에서의 온칩 메모리가 일반적으로 컴퓨팅 디바이스의 시스템 메모리보다 훨씬 적은 저장 공간을 포함하기 때문에, GPU 에서의 온칩 메모리는 GPU 에서의 셰이더 유닛들에 의해 발생되고 소비되는 데이터의 모두에 대해 별개의 데이터 채널들을 할당하기에 충분한 저장 공간을 가지지 않을 수도 있다. 더욱이, 셰이더 파이프라인의 일부 스테이지들이 셰이더 파이프라인의 다른 스테이지들보다 더 많은 데이터를 발생시키는 경향이 있어 셰이더 파이프라인의 스테이지들이 평형되지 않을 수도 있기 때문에, 셰이더 파이프라인의 각각의 스테이지에 의해 발생되는 데이터에 대해 온칩 메모리에서 동일한 공간을 할당하는 것은 온칩 메모리에서의 저장 공간을 낭비할 수도 있다. 게다가, 온칩 메모리는 셰이더 파이프라인의 각각의 스테이지에 의해 발생되는 데이터에 대해 동일한 공간을 할당하기에 충분한 저장 공간을 가지지 않을 수도 있으며, 그 결과, 셰이더 파이프라인의 스테이지들에 의해 발생되는 데이터 중 적어도 일부는 더 느린 시스템 메모리에 저장될 필요가 있으며, 그에 따라서 GPU 의 성능을 감소시킬 수도 있다.

[0012] 본 개시물의 양태들에 따르면, GPU 는 셰이더 파이프라인의 2개 이상의 스테이지들이 단일 공유 데이터 채널을 공유할 수 있도록 GPU 의 온칩 메모리에서 공유 데이터 채널들을 할당할 수도 있으며, 그 결과, 공유 데이터 채널을 공유하고 있는 셰이더 파이프라인의 제 1 스테이지로부터의 데이터를 저장하는데 사용되고 있지 않는 공유 데이터 채널에서의 공간이 데이터 채널을 공유하고 있는 셰이더 파이프라인의 제 2 스테이지로부터의 데이터를 저장하는데 사용될 수도 있다. 이러한 방법으로, GPU 의 온칩 메모리가 좀더 효율적인 방법으로 이용될 수도 있다. 더욱이, 다른 접근법들에 비해 더 많은, 셰이더 파이프라인에 의해 발생되는 데이터를 잠재적으로 저장하는 좀더 효율적인 방법으로 GPU 의 온칩 메모리를 이용함으로써, GPU 에서의 온칩 메모리는 셰이더 파이프라인의 스테이지들을 실행하는 셰이더 유닛들에 의해 소비될 준비가 된 더 많은 데이터를 저장함으로써, 셰이더 유닛들의 이용을 증가시키고 GPU 의 성능을 증가시킬 수도 있다.

[0013] 도 1 은 다수의 그래픽스 파이프라인 스테이지들 간에 데이터 채널들을 공유하는 본 개시물의 하나 이상의 양태들을 구현하도록 구성될 수도 있는 예시적인 컴퓨팅 디바이스를 예시하는 블록도이다. 도 1 에 나타낸 바와 같이, 디바이스 (2) 는 비디오 디바이스들, 미디어 플레이어들, 셋-탑 박스들, 무선 핸드셋들, 예컨대 모바일 전화기들 및 소위 스마트폰들, 개인 휴대정보 단말기들 (PDA들), 데스크탑 컴퓨터들, 랩탑 컴퓨터들, 게이밍 콘솔들, 화상 회의 유닛들, 태블릿 컴퓨팅 디바이스들 등을 포함하지만, 이에 한정되지 않는 컴퓨팅 디바이스일 수도 있다. 도 1 의 예에서, 디바이스 (2) 는 중앙 프로세싱 유닛 (CPU) (6), 시스템 메모리 (10), 및 GPU (12) 를 포함할 수도 있다 디바이스 (2) 는 또한 디스플레이 프로세서 (14), 송수신기 모듈 (3), 사용자 인터페이스 (4), 및 디스플레이 (8) 를 포함할 수도 있다. 송수신기 모듈 (3) 및 디스플레이 프로세서 (14) 는 양쪽다 CPU (6) 및/또는 GPU (12) 와 동일한 집적 회로 (IC) 의 부분일 수도 있거나, 양쪽다 CPU (6) 및/또는 GPU (12) 를 포함하는 IC 또는 IC들의 외부에 있을 수도 있거나, 또는 CPU (6) 및/또는 GPU (12) 를 포함하는 IC 의 외부에 있는 IC 에 형성될 수도 있다.

[0014] 디바이스 (2) 는, 명료성의 목적들을 위해 도 1 에 미도시된 추가적인 모듈들 또는 유닛들을 포함할 수도 있다. 예를 들어, 디바이스 (2) 는 디바이스 (2) 가 모바일 무선 전화기 또는 스피커이거나 디바이스 (2) 가 미디어 플레이어인 예들에서 회선 통신들 (telephonic communications) 을 실시하기 위해 도 1 에 어느 것도 도시되지 않은 스피커 및 마이크로폰을 포함할 수도 있다. 디바이스 (2) 는 또한 비디오 카메라를 포함할 수도 있다. 더욱이, 디바이스 (2) 에 나타낸 여러 모듈들 및 유닛들은 디바이스 (2) 의 모든 예에서 필요하지 않을 수도 있다. 예를 들어, 사용자 인터페이스 (4) 및 디스플레이 (8) 는 디바이스 (2) 가 데스크탑 컴퓨터, 또는 외부 사용자 인터페이스 또는 디스플레이와 인터페이스하기 위해 탑재되는 다른 디바이스인 예들에서 디바이스 (2) 의 외부에 있을 수도 있다.

[0015] 사용자 인터페이스 (4) 의 예들은 트랙볼, 마우스, 키보드, 및 다른 유형들의 입력 디바이스들을 포함하지만 이에 한정되지 않는다. 사용자 인터페이스 (4) 는 또한 터치 스크린일 수도 있으며, 디스플레이 (8) 의 일부분으로서 포함될 수도 있다. 송수신기 모듈 (3) 은 컴퓨팅 디바이스 (2) 와 또 다른 디바이스 또는 네트워크 사이에 무선 또는 유선 통신을 가능하게 하기 위해 회로를 포함할 수도 있다. 송수신기 모듈 (3) 은 변조기들, 복조기들, 증폭기들 및 유선 또는 무선 통신을 위한 이러한 다른 회로를 포함할 수도 있다.

[0016] CPU (6) 는 실행을 위한 컴퓨터 프로그램의 명령들을 프로세싱하도록 구성된 중앙 프로세싱 유닛 (CPU) 과 같은, 마이크로프로세서일 수도 있다. CPU (6) 는 컴퓨팅 디바이스 (2) 의 동작을 제어하는 범용 또는 특수-목적 프로세서를 포함할 수도 있다. 사용자는 CPU (6) 로 하여금 하나 이상의 소프트웨어 애플리케이션들을 실행하도록 하기 위해서 컴퓨팅 디바이스 (2) 에 입력을 제공할 수도 있다. CPU (6) 상에서 실행하는 소프트웨어 애플리케이션들은 예를 들어, 운영 시스템, 워드 프로세서 애플리케이션, 이메일 애플리케이션, 스프레드 시트 애플리케이션, 미디어 플레이어 애플리케이션, 비디오 게임 애플리케이션, 그래픽 사용자 인터페이스 애플리케이션 또는 또 다른 프로그램을 포함할 수도 있다. 게다가, CPU (6) 는 GPU (12) 의 동작을 제어하기 위해 GPU 드라이버 (22) 를 실행할 수도 있다. 사용자는 키보드, 마우스, 마이크로폰, 터치 패드 또는 컴퓨팅 디바이스 (2) 에 사용자 인터페이스 (4) 를 통해서 커플링된 다른 입력 디바이스와 같은 하나 이상의 입력 디바이스들 (미도시) 을 통해서 컴퓨팅 디바이스 (2) 에 입력을 제공할 수도 있다.

[0017] CPU (6) 상에서 실행하는 소프트웨어 애플리케이션들은 CPU (6) 에게 디스플레이 (8) 로의 그래픽스 데이터의 렌더링을 발생시키도록 명령하는 하나 이상의 그래픽스 렌더링 명령들을 포함할 수도 있다. 일부 예들에서, 소프트웨어 명령들은 예컨대, 개방 그래픽스 라이브러리 (OpenGL®) API, OpenGL ES (Open Graphics Library Embedded Systems) API, Direct3D API, X3D API, RenderMan API, WebGL API, 또는 임의의 다른 공개 또는 독점 표준 그래픽스 API 와 같은, 그래픽스 애플리케이션 프로그래밍 인터페이스 (API) 를 따를 수도 있다. 그래픽스 렌더링 명령들을 프로세싱하기 위해, CPU (6) 는 하나 이상의 그래픽스 렌더링 지령들을 (예컨대, GPU 드라이버 (22) 를 통해서) GPU (12) 로 발하여, GPU (12) 로 하여금 그래픽스 데이터의 렌더링의 일부 또는 모두를 수행하게 할 수도 있다. 일부 예들에서, 렌더링될 그래픽스 데이터는 그래픽스 프리머티브들의 리스트, 예컨대, 포인트들, 라인들, 삼각형들, 사변형들, 삼각형 스트립들, 등을 포함할 수도 있다.

[0018] GPU (12) 는 하나 이상의 그래픽스 프리머티브들을 디스플레이 (8) 에 렌더링하는 그래픽스 동작들을 수행하도록 구성될 수도 있다. 따라서, CPU (6) 상에서 실행하는 소프트웨어 애플리케이션들 중 하나가 그래픽스 프로세싱을 필요로 할 때, CPU (6) 는 디스플레이 (8) 에의 렌더링을 위해 그래픽스 지령들 및 그래픽스 데이터를 GPU (12) 에 제공할 수도 있다. 그래픽스 데이터는 예컨대, 그리기 지령들, 상태 정보, 프리미티브 정보, 텍스쳐 정보, 등을 포함할 수도 있다. GPU (12) 는 일부 경우, CPU (6) 보다 좀더 효율적인 복합 그래픽 관련 동작들의 프로세싱을 제공하는 고속-병렬 구조 (highly-parallel structure) 로 형성될 수도 있다. 예를 들어, GPU (12) 는 다수의 정점들 또는 픽셀들에 대해 병렬 방식으로 동작하도록 구성되는, 셰이더 유닛들과 같은, 복수의 프로세싱 엘리먼트들을 포함할 수도 있다. GPU (12) 의 고속 병렬 성질은 일부 경우, GPU (12) 로 하여금, CPU (6) 를 이용하여 장면들을 디스플레이 (8) 에 직접 그리는 것 보다 더 빨리, 그래픽스 이미지들 (예컨대, GUI들 및 2차원 (2D) 및/또는 3차원 (3D) 그래픽스 장면들) 을 디스플레이 (8) 상에 그리는 것을 가능하게 할 수도 있다.

[0019] GPU (12) 는 일부 경우, 컴퓨팅 디바이스 (2) 의 마더보더에 통합될 수도 있다. 다른 경우, GPU (12) 는 컴퓨팅 디바이스 (2) 의 마더보더에서의 포트에 설치된 그래픽스 카드 상에 존재할 수도 있거나 또는 아니면, 컴퓨팅 디바이스 (2) 와 상호운용하도록 구성된 주변장치 디바이스 내에 통합될 수도 있다. GPU (12) 는 하나 이상의 마이크로프로세서들, 주문형 집적회로들 (ASIC들), 필드 프로그래밍가능 게이트 어레이들 (FPGA들), 디지털 신호 프로세서들 (DSP들), 또는 다른 등가의 통합 또는 이산 로직 회로와 같은, 하나 이상의 프로세서들을 포함할 수도 있다. GPU (12) 는 또한 하나 이상의 프로세서 코어들을 포함할 수도 있으며, 따라서 GPU (12) 가 멀티-코어 프로세서로서 지칭될 수도 있다.

[0020] GPU (12) 는 그래픽스 메모리 (40) 에 직접 커플링될 수도 있다. 따라서, GPU (12) 는 버스를 이용하지 않고 그래픽스 메모리 (40) 로부터 데이터를 판독하고 그래픽스 메모리 (14) 에 데이터를 기록할 수도 있다. 즉, GPU (12) 는 오프-칩 메모리 대신, 로컬 스토리지를 이용하여, 데이터를 로컬로 프로세싱할 수도 있다. 이러한 그래픽스 메모리 (40) 는 온칩 메모리로서 지칭될 수도 있다. 이것은 막대한 버스 트래픽을 겪을 수도 있는, GPU (12) 가 버스를 통해서 데이터를 판독하고 기록할 필요성을 제거함으로써, GPU (12) 로 하여금 좀더 효율적인 방법으로 동작가능하게 한다. 일부의 경우, 그러나, GPU (12) 는 별개의 메모리를 포함하지 않고, 대신 버스를 통해서 시스템 메모리 (10) 를 이용할 수도 있다. 그래픽스 메모리 (40) 는 예컨대, 랜

덤 액세스 메모리 (RAM), 정적 RAM (SRAM), 동적 RAM (DRAM), 소거가능한 프로그래밍가능 ROM (EPROM), 전기적 소거가능한 프로그래밍가능 ROM (EEPROM), 플래시 메모리, 자기 데이터 매체들 또는 광학 저장 매체들과 같은, 하나 이상의 휘발성 또는 비-휘발성 메모리들 또는 저장 디바이스들을 포함할 수도 있다.

[0021] 일부 예들에서, GPU (12) 는 완전히 형성된 이미지를 시스템 메모리 (10) 에 저장할 수도 있다. 디스플레이 프로세서 (14) 는 시스템 메모리 (10) 로부터 이미지를 취출하고 디스플레이 (8) 의 픽셀들로 하여금 이미지를 디스플레이하기 위해 조사하도록 하는 값들을 출력할 수도 있다. 디스플레이 (8) 는 GPU (12) 에 의해 발생되는 이미지 콘텐츠를 디스플레이하는 디바이스 (2) 의 디스플레이일 수도 있다. 디스플레이 (8) 는 액정 디스플레이 (LCD), 유기 발광 다이오드 디스플레이 (OLED), 음극선관 (CRT) 디스플레이, 플라즈마 디스플레이, 또는 다른 유형의 디스플레이 디바이스일 수도 있다.

[0022] 본 개시물의 양태들에 따르면, GPU (12) 는 그의 셰이더 유닛들 상에서 복수의 그래픽스 파이프라인 스테이지들을 실행할 수도 있다 GPU (12) 는 셰이더 유닛 상에서 실행하는 복수의 그래픽스 파이프라인 스테이지들의 2개의 스테이지들에 의해 공유되는 그래픽스 메모리 (40) 에서 공유 데이터 채널을 생성할 수도 있다. GPU (12) 는 셰이더 유닛들 상에서 실행하는 복수의 그래픽스 파이프라인 스테이지들의 2개의 스테이지들의 각각에 의해 발생되는 데이터를 공유 데이터 채널에 저장할 수도 있다.

[0023] 도 2 는 3차원 장면의 2차원 표현을 생성하기 위해 GPU (12) 에 의해 수행될 수도 있는 예시적인 그래픽스 프로세싱 파이프라인 (24) 을 예시하는 블록도이다. 그래픽스 프로세싱 파이프라인 (24) 은 그래픽스 프로세싱 지령들을 실행하기 위해 함께 동작하는 복수의 그래픽스 프로세싱 스테이지들을 포함할 수도 있다. 도 2 에 나타낸 바와 같이, 그래픽스 프로세싱 파이프라인 (24) 은 입력-어셈블러 (26), 정점 셰이더 스테이지 (28), 헐 셰이더 스테이지 (30), 테셀레이터 스테이지 (32), 도메인 셰이더 스테이지 (34), 지오메트리 셰이더 스테이지 (36), 및 픽셀 셰이더 스테이지 (38) 를 포함할 수도 있다. 그래픽스 프로세싱 파이프라인 (24) 에서의 구성요소들의 각각은 고정-함수 구성요소들로서, 프로그래밍가능 구성요소들로서 (예컨대, 프로그래밍가능 셰이더 유닛 상에서 실행하는 셰이더 프로그램의 일부로서), 또는 고정-함수 구성요소들과 프로그래밍가능 구성요소들의 조합으로서 구현될 수도 있다.

[0024] GPU (12) 는 CPU (6) 로부터, GPU 드라이버 (22) 를 통해서, 하나 이상의 그래픽스 프로세싱 지령들을 수신하고 그래픽스 프로세싱 지령들을 그래픽스 프로세싱 파이프라인 (24) 을 통해서 실행하여 디스플레이가능한 그래픽스 이미지들을 발생시키도록 구성될 수도 있다. 위에서 설명한 바와 같이, 그래픽스 프로세싱 파이프라인 (24) 은 그래픽스 프로세싱 지령들을 실행하기 위해 함께 동작하는 복수의 스테이지들을 포함한다.

[0025] 그래픽스 프로세싱 파이프라인 (24) 에서의 입력-어셈블러 (26) 는 일반적으로 그래픽스 데이터 (예컨대, 삼각형들, 라인들, 및 포인트들) 를 그래픽스 프로세싱 파이프라인 (24) 에 공급하는 것을 담당하는 고정 함수 스테이지일 수도 있다. 예를 들어, 입력 어셈블러 스테이지 (26) 는 고차 표면들, 프리미티브들, 및 기타 등등에 대한 정점 데이터를 수집하고, 정점 데이터 및 속성들을 정점 셰이더 스테이지 (28) 로 출력할 수도 있다. 따라서, 입력 어셈블러 스테이지 (26) 는 고정 함수 동작들을 이용하여, 시스템 메모리 (10) 와 같은, 오프-칩 메모리로부터 정점들을 판독할 수도 있다. 입력 어셈블러 스테이지 (26) 는 그후 이들 정점들로부터 파이프라인 작업 아이템들을 생성할 수도 있으며, 또한 정점 식별자들 ("VertexID들"), 인스턴스 식별자들 (정점 셰이더에 이용가능한 "InstanceID들") 및 프리미티브 식별자들 (지오메트리 셰이더 및 픽셀 셰이더에 이용가능한 "PrimitiveID들") 을 발생시킬 수도 있다. 입력 어셈블러 스테이지 (26) 는 정점들을 판독하자 마자, VertexID들, InstanceID들, 및 PrimitiveID들을 자동적으로 발생시킬 수도 있다.

[0026] 정점 셰이더 스테이지 (28) 는 수신된 정점 데이터 및 속성들을 프로세싱할 수도 있다. 예를 들어, 정점 셰이더 스테이지 (28) 는 변환들, 스키닝 (skinning), 정점 변위, 및 정점-당 재료 속성들을 계산하는 것과 같은, 정점-당 프로세싱을 수행할 수도 있다. 일부 예들에서, 정점 셰이더 스테이지 (28) 는 텍스쳐 좌표들, 정점 칼라, 정점 조명, 안개 인자들 등을 발생시킬 수도 있다. 정점 셰이더 스테이지 (28) 는 일반적으로 단일 입력 정점을 취하고, 단일, 프로세싱된 출력 정점을 출력한다.

[0027] 헐 셰이더 스테이지 (30), 테셀레이터 (32), 및 도메인 셰이더 스테이지 (34) 는 테셀레이션 스테이지들로서 일괄하여 지칭될 수도 있다. 테셀레이션 스테이지들은 낮은-세부 재분할 표면들을 더 높은-세부 프리미티브들로 변환하고, 높은-차수 표면들을 렌더링에 적합한 표면들 (예컨대, 삼각형들) 로 타일링한다. 헐 셰이더 스테이지 (30) 는 정점 셰이더 스테이지 (28) 로부터 프리미티브들을 수신하고, 적어도 2개의 액션들을 실행하는 것을 담당한다. 먼저, 헐 셰이더 스테이지 (30) 는 테셀레이션 인자들의 세트를 결정하는 것을 일반적으로 담당한다. 헐 셰이더 스테이지 (30) 는 프리미티브 당 한번 테셀레이션 인자들을 발생시킬 수도 있다.

테셀레이션 인자들은 주어진 프리미티브로 미세하게 테셀레이트하는 (예컨대, 프리미티브를 더 작은 부분들로 분할하는) 방법을 결정하기 위해 테셀레이터 스테이지 (32) 에 의해 사용될 수도 있다. 헐 셰이더 스테이지 (30) 는 또한 도메인 셰이더 스테이지 (34) 에 의해 추후 사용될 제어 포인트들을 발생시키는 것을 담당한다. 즉, 예를 들어, 헐 셰이더 스테이지 (30) 는 결국 렌더링에 사용되는 실제 테셀레이트된 정점들을 생성하기 위해 도메인 셰이더 스테이지 (34) 에 의해 사용될 제어 포인트들을 발생시키는 것을 담당한다.

[0028] 테셀레이터 스테이지 (32) 가 헐 셰이더 스테이지 (30) 로부터 데이터를 수신할 때, 테셀레이터 스테이지 (32) 는 현재의 프리미티브 유형에 적합한 샘플링 패턴을 결정하기 위해 여러 알고리즘들 중 하나를 이용한다. 예를 들어, 일반적으로, 테셀레이터 스테이지 (32) 는 (헐 셰이더 스테이지 (30) 에 의해 결정된 것과 같은) 요청된 테셀레이션의 양을 현재의 "도메인" 내 좌표 포인트들의 그룹으로 변환한다. 즉, 헐 셰이더 스테이지 (30) 로부터의 테셀레이션 인자들 뿐만 아니라, 테셀레이터 스테이지 (32) 의 특정의 구성에 따라서, 테셀레이터 스테이지 (32) 는 현재의 프리미티브에서 어느 포인트들이 입력 프리미티브를 더 작은 부분들로 테셀레이트하기 위해 샘플링될 필요가 있는지를 결정한다. 테셀레이터 스테이지 (32) 의 출력은 무게중심 좌표들을 포함할 수도 있는 도메인 포인트들의 세트일 수도 있다.

[0029] 도메인 셰이더 스테이지 (34) 는 헐 셰이더 스테이지 (30) 에 의해 발생되는 제어 포인트들에 더해서, 도메인 포인트들을 취하고, 그 도메인 포인트들을 이용하여 새로운 정점들을 생성한다. 도메인 셰이더 스테이지 (34) 는 현재의 프리미티브, 텍스쳐들, 절차적 알고리즘들, 또는 그밖의 다른 것에 대해 발생된 제어 포인트들의 완전한 리스트를 이용하여, 각각의 테셀레이트된 포인트에 대한 무게중심 "로케이션" 을 파이프라인에서 다음 스테이지 상으로 전달되는 출력 지오메트리로 변환할 수 있다.

[0030] 지오메트리 셰이더 스테이지 (36) 는 그의 정점 데이터 (예컨대, 삼각형에 대해 3개의 정점들, 라인에 대해 2개의 정점들, 또는 포인트에 대해 단일 정점) 에 의해 정의되는 프리미티브를 수신하고 프리미티브를 추가로 프로세싱할 수도 있다. 예를 들어, 지오메트리 셰이더 스테이지 (36) 는 다른 가능한 프로세싱 동작들 중에서, 실루엣-에지 검출 및 쉐도우 용적 압출 (shadow volume extrusion) 과 같은 프리미티브 당 프로세싱 (per-primitive processing) 을 수행할 수도 있다. 따라서, 지오메트리 셰이더 스테이지 (36) 는 하나의 프리미티브를 (하나 이상의 정점들을 포함할 수도 있는) 입력으로서 수신하고, (하나 이상의 정점들을 또한 포함할 수도 있는) 제로, 하나, 또는 다수의 프리미티브들을 출력할 수도 있다. 출력 프리미티브는 지오메트리 셰이더 스테이지 (36) 없이 가능할 수도 있는 더 많은 데이터를 포함할 수도 있다. 출력 데이터의 전체 양은 정점 카운트로 곱한 정점 사이즈와 동일할 수도 있으며, 호출 당 제한될 수도 있다. 지오메트리 셰이더 스테이지 (36) 로부터의 스트림 출력은 이 스테이지에 도달하는 프리미티브들이 시스템 메모리 (10) 와 같은 오프-칩 메모리에 저장될 수 있도록 할 수도 있다. 스트림 출력은 일반적으로 지오메트리 셰이더 스테이지 (36) 에 속박되며, 양쪽 다 (예컨대, API 를 이용하여) 함께 프로그래밍될 수도 있다.

[0031] 래스터라이저 스테이지 (37) 는 일반적으로 프리미티브들을 클리핑하고 픽셀 셰이더 스테이지 (38) 에 대한 프리미티브들을 준비하는 것을 담당하는 고정 함수 스테이지이다. 예를 들어, 래스터라이저 스테이지 (37) 는 (맞춤 클립 경계들을 포함한) 클리핑, 원근 분할 (perspective divide), 뷰포트/가위 (scissor) 선택 및 구현, 렌더 목표 선택 및 프리미티브 셋업을 수행할 수도 있다. 이러한 방법으로, 래스터라이저 스테이지 (37) 는 픽셀 셰이더 스테이지 (38) 에 의한 셰이딩을 위해 다수의 단편들을 발생시킬 수도 있다.

[0032] 픽셀 셰이더 스테이지 (38) 는 래스터라이저 스테이지 (37) 로부터 단편들을 수신하고 칼라와 같은 픽셀 당 데이터를 발생시킨다. 픽셀 셰이더 스테이지 (38) 는 또한 텍스쳐 블렌딩 및 조명 모델 계산과 같은, 픽셀 당 프로세싱을 수행할 수도 있다. 따라서, 픽셀 셰이더 스테이지 (38) 는 하나의 픽셀을 입력으로서 수신할 수도 있으며, 동일한 상대적인 위치 (또는, 그 픽셀에 대해 제로 값) 에서 하나의 픽셀을 출력할 수도 있다.

[0033] 본 개시물의 양태들에 따르면, 그래픽스 프로세싱 파이프라인 (24) 의 2개 이상의 스테이지들은 그래픽스 메모리 (40) 에서 공유 데이터 채널을 공유할 수도 있다. 예를 들어, 정점 셰이더 스테이지 (28) 및 도메인 셰이더 스테이지 (34) 에 의해 발생되는 정점들은 공유 데이터 채널에 저장될 수도 있다. 더욱이, 헐 셰이더 스테이지 (30) 및 지오메트리 셰이더 스테이지 (36) 에 의해 발생되는 프리미티브들은 다른 공유 데이터 채널에 저장될 수도 있다. 이러한 방법으로, GPU (12) 는 그래픽스 메모리 (40) 를 좀더 효율적으로 이용할 수도 있다.

[0034] 도 3 은 도 1 의 CPU (6), GPU (12), 및 시스템 메모리 (10) 의 예시적인 구현예들을 좀더 상세히 예시하는 블록도이다. 도 3 에 나타낸 바와 같이, CPU (6) 는 적어도 하나의 소프트웨어 애플리케이션 (18), 그래픽스 API (20), 및 GPU 드라이버 (22) 를 포함할 수도 있으며, 이들 각각은 CPU (6) 상에서 실행하는 하나 이상의 소

프트웨어 애플리케이션들 또는 서비스들일 수도 있다.

[0035] CPU (6) 및 GPU (12) 에 이용가능한 메모리는 시스템 메모리 (10) 및 프레임 버퍼 (16) 를 포함할 수도 있다. 프레임 버퍼 (16) 는 시스템 메모리 (10) 의 일부분일 수도 있거나 또는 시스템 메모리 (10) 로부터 분리될 수도 있다. 프레임 버퍼 (16) 는 렌더링된 이미지 데이터를 저장할 수도 있다.

[0036] 소프트웨어 애플리케이션 (18) 은 GPU (12) 의 기능을 이용하는 임의의 애플리케이션일 수도 있다. 예를 들어, 소프트웨어 애플리케이션 (18) 은 GUI 애플리케이션, 운영 시스템, 휴대형 맵핑 애플리케이션, 엔지니어링 또는 예술적 애플리케이션들을 위한 컴퓨터 지원 설계 프로그램, 비디오 게임 애플리케이션, 또는 2D 또는 3D 그래픽스를 이용하는 또다른 유형의 소프트웨어 애플리케이션일 수도 있다.

[0037] 소프트웨어 애플리케이션 (18) 은 GPU (12) 에게 그래픽 사용자 인터페이스 (GUI) 및/또는 그래픽스 장면을 렌더링하도록 명령하는 하나 이상의 그리기 명령들을 포함할 수도 있다. 예를 들어, 그리기 명령들은 GPU (12) 에 의해 렌더링될 하나 이상의 그래픽스 프리머티브들의 세트를 정의하는 명령들을 포함할 수도 있다. 일부 예들에서, 그리기 명령들은 일괄하여, GUI 에 사용되는 복수의 윈도우잉 표면들의 모두 또는 일부를 정의할 수도 있다. 추가적인 예들에서, 그리기 명령들은 일괄하여, 애플리케이션에 의해 정의되는 모델 공간 또는 세계 공간 내에 하나 이상의 그래픽스 오브젝트들을 포함하는 그래픽스 장면의 모두 또는 일부를 정의할 수도 있다.

[0038] 소프트웨어 애플리케이션 (18) 은 하나 이상의 그래픽스 프리머티브들을 디스플레이가능한 그래픽스 이미지들로 렌더링하기 위해, 그래픽스 API (20) 를 통해서, GPU 드라이버 (22) 를 호출하여, 하나 이상의 지령들을 GPU (12) 로 발할 수도 있다. 예를 들어, 소프트웨어 애플리케이션 (18) 은 그래픽스 API (20) 를 통해서, GPU 드라이버 (22) 를 호출하여, 프리미티브 정의들을 GPU (12) 에 제공할 수도 있다. 일부의 경우, 프리미티브 정의들은 GPU (12) 에 그리기 프리미티브들의 리스트의 유형, 예컨대, 삼각형들, 직사각형들, 삼각형 팬들, 삼각형 스트립들, 등으로 제공될 수도 있다. 프리미티브 정의들은 렌더링될 프리미티브들과 연관되는 하나 이상의 정점들을 규정하는 정점 사양들을 포함할 수도 있다. 정점 사양들은 각각의 정점에 대한 위치 좌표들 및, 일부 경우, 정점과 연관되는 다른 속성들, 이를 테면 예컨대, 칼라 좌표들, 법선 벡터들, 및 텍스쳐 좌표들을 포함할 수도 있다. 프리미티브 정의들은 또한 프리미티브 유형 정보 (예컨대, 삼각형, 직사각형, 삼각형 팬 (triangle fan), 삼각형 스트립, 등), 스케일링 정보, 회전 정보 등을 포함할 수도 있다. 소프트웨어 애플리케이션 (18) 에 의해 GPU 드라이버 (22) 로 발해진 명령들에 기초하여, GPU 드라이버 (22) 는 프리미티브를 렌더링하기 위해 수행할 GPU (12) 에 대한 하나 이상의 동작들을 규정하는 하나 이상의 지령들을 형성할 수도 있다. GPU (12) 가 CPU (6) 로부터 지령을 수신할 때, 그래픽스 프로세싱 파이프라인 (24) 은 지령을 디코딩하고, 그 지령에 규정된 동작을 수행하도록 그래픽스 프로세싱 파이프라인 (24) 을 구성한다. 예를 들어, 그래픽스 프로세싱 파이프라인 (24) 에서의 입력-어셈블러 (26) 는 프리미티브 데이터를 판독하고 그래픽스 프로세싱 파이프라인 (24) 에서 다른 그래픽스 파이프라인 스테이지들에 의한 사용을 위해 그 데이터를 프리미티브들로 조립할 수도 있다. 규정된 동작들을 수행한 후, 그래픽스 프로세싱 파이프라인 (24) 은 렌더링된 데이터를 디스플레이 디바이스와 연관되는 프레임 버퍼 (16) 로 출력한다.

[0039] 프레임 버퍼 (16) 는 GPU (12) 에 대한 목적지 픽셀들을 저장한다. 각각의 목적지 픽셀은 고유한 스크린 픽셀 로케이션과 연관될 수도 있다. 일부 예들에서, 프레임 버퍼 (16) 는 각각의 목적지 픽셀에 대한 칼라 성분들 및 목적지 알파 (alpha) 값을 저장할 수도 있다. 예를 들어, 프레임 버퍼 (16) 는 각각의 픽셀에 대해 적색, 녹색, 청색, 알파 (RGBA) 성분들을 저장할 수도 있으며, 여기서, "RGB" 성분들은 칼라 값들에 대응하며 "A" 성분은 목적지 알파 값에 대응한다. 프레임 버퍼 (16) 및 시스템 메모리 (10) 는 별개의 메모리 유닛들인 것으로 예시되지만, 다른 예들에서, 프레임 버퍼 (16) 는 시스템 메모리 (10) 의 일부일 수도 있다.

[0040] 일부 예들에서, 그래픽스 프로세싱 파이프라인 (24) 의 정점 셰이더 스테이지 (28), 헐 셰이더 스테이지 (30), 도메인 셰이더 스테이지 (34), 지오메트리 셰이더 스테이지, 및 픽셀 셰이더 스테이지 (38) 는 셰이더 스테이지들로서 간주될 수도 있다. 이들 셰이더 스테이지들은 GPU (12) 의 셰이더 유닛들 (46) 상에서 실행하는 하나 이상의 셰이더 프로그램들로서 구현될 수도 있다. 셰이더 유닛들 (46) 은 프로세싱 구성요소들의 프로그래밍가능한 파이프라인으로서 구성될 수도 있다. 일부 예들에서, 셰이딩 유닛 (46) 은 "셰이더 프로세서들" 또는 "통합 셰이더들" 로서 지칭될 수도 있으며, 지오메트리, 정점, 픽셀, 또는 다른 셰이딩 동작들을 수행하여 그래픽스를 렌더링할 수도 있다. 셰이더 유닛들 (46) 은 프로세서 코어들 (48) 을 포함할 수도 있으며, 그 프로세서 코어의 각각은 페칭 및 디코딩 동작들을 위한 하나 이상의 구성요소들, 산술적 계산들을 실행하는 하나 이상의 산술 로직 유닛들, 하나 이상의 메모리들, 캐시들, 및 레지스터들을 포함할 수도 있다.

[0041] GPU (12) 는 그래픽스 프로세싱 파이프라인 (24) 에서의 정점 셰이더 스테이지 (28), 헐 셰이더 스테이지 (30), 도메인 셰이더 스테이지 (34), 지오메트리 셰이더 스테이지 (36), 및 픽셀 셰이더 스테이지 (38) 중 하나 이상을 실행하기 위해 지령들을 셰이더 유닛들 (46) 로 전송함으로써 정점 셰이딩, 헐 셰이딩, 도메인 셰이딩, 지오메트리 셰이딩, 픽셀 셰이딩, 및 기타 등등과 같은 다양한 셰이딩 동작들을 수행하도록 셰이더 유닛들 (46) 을 지정할 수도 있다. 일부 예들에서, GPU 드라이버 (22) 는 하나 이상의 셰이더 프로그램들을 컴파일하고, 그 컴파일된 셰이더 프로그램들을 GPU (12) 내에 포함되는 하나 이상의 프로그래밍가능 셰이더 유닛들 상으로 다운로드하도록 구성될 수도 있다. 셰이더 프로그램은, 예컨대, GLSL (OpenGL Shading Language), HLSL (High Level Shading Language), Cg (C for Graphics) 셰이딩 언어, 등과 같은, 고급 셰이딩 언어로 기록될 수도 있다. 컴파일된 셰이더 프로그램들은 GPU (12) 내 셰이더 유닛들 (46) 의 동작을 제어하는 하나 이상의 명령들을 포함할 수도 있다. 예를 들어, 셰이더 프로그램들은 셰이더 유닛들 (46) 에 의해 실행될 수도 있는, 정점 셰이더 스테이지 (28) 의 기능들을 수행하는 정점 셰이더 프로그램들, 셰이더 유닛들 (46) 에 의해 실행될 수도 있는, 헐 셰이더 스테이지 (30) 의 기능들을 수행하는 헐 셰이더 프로그램들, 셰이더 유닛 (46) 에 의해 실행될 수도 있는, 도메인 셰이더 스테이지 (34) 의 기능들을 수행하는 도메인 셰이더 프로그램들, 셰이더 유닛 (46) 에 의해 실행될 수도 있는, 지오메트리 셰이더 스테이지 (36) 의 기능들을 수행하는 지오메트리 셰이더 프로그램들 및/또는 셰이더 유닛들 (46) 에 의해 실행될 수도 있는, 픽셀 셰이더 (38) 의 기능들을 수행하는 픽셀 셰이더 프로그램들을 포함할 수도 있다. 정점 셰이더 프로그램은 프로그래밍가능 정점 셰이더 유닛 또는 통합된 셰이더 유닛의 실행을 제어할 수도 있으며, 하나 이상의 정점-당 동작들을 규정하는 명령들을 포함할 수도 있다.

[0042] 그래픽스 메모리 (40) 는 GPU (12) 의 집적 회로에 물리적으로 통합되는 온칩 스토리지 또는 메모리이다. 그래픽스 메모리 (40) 가 온칩이기 때문에, GPU (12) 는 시스템 버스를 통해서 시스템 메모리 (10) 로부터 값들을 판독하거나 그에 기록하는 것보다 더 빨리, 그래픽스 메모리 (40) 로부터 값들을 판독하고 그에 값들을 기록가능할 수도 있다. 이와 같이, 셰이더 유닛들 (46) 의 성능이 그래픽스 프로세싱 파이프라인 (24) 의 셰이더 스테이지들에 의해 발생되어 소비되는 데이터를 그래픽스 메모리 (40) 로부터 저장하고 판독함으로써 증가될 수도 있다.

[0043] 본 개시물의 양태들에 따르면, 셰이더 유닛들 (46) 은 다수의 셰이딩 동작들을 프로세서 코어들 (48) 상에서 동시에 수행할 수도 있다. GPU (12) 는 그래픽스 프로세싱 파이프라인 (24) 의 상이한 셰이딩 스테이지들이 상이한 프로세서 코어들 (48) 상에서 실행되고 이에 의해 그래픽스 프로세싱 파이프라인 (24) 의 스테이지들을 인터리브가능하게 하는 지령들을 셰이딩 유닛 (46) 으로 전송할 수도 있다. 예를 들어, GPU (12) 는 셰이딩 유닛 (46) 이 정점 셰이더 스테이지 (28) 및 지오메트리 셰이더 스테이지 (36) 를 셰이더 유닛들 (46) 의 상이한 프로세서 코어들 (48) 상에서 동시에 실행하도록 하는 지령들을 셰이딩 유닛 (46) 으로 전송할 수도 있다.
다른 예에서, GPU (12) 는 셰이딩 유닛 (46) 이 지오메트리 셰이더 스테이지 (36) 의 다수의 인스턴스들을 다중 프로세서 상에서 동시에 실행하도록 하는 지령들을 셰이딩 유닛 (46) 으로 전송할 수도 있다.

[0044] 본 개시물의 양태들에 따르면, 그래픽스 메모리 (40) 는 그래픽스 프로세싱 파이프라인 (24) 의 상이한 스테이지들에 의해 발생되는 데이터가 단일 데이터 채널을 공유가능하게 함으로써, GPU (12) 가 그래픽스 메모리 (40) 에서 제한된 공간을 좀더 효율적으로 이용가능하게 하고 또한 셰이더 프로세서 클러스터 (46) 가 그의 프로세서 코어들 (48) 의 이용을 증가가능하게 하여, 그래픽스 프로세싱 파이프라인 (24) 의 다수의 스테이지들을 동시에 실행가능하게 하는 공유 데이터 채널들 (50A-50N) ("공유 데이터 채널들 (50)") 중 하나 이상을 포함할 수도 있다.

[0045] 공유 데이터 채널들 (50) 에서의 각각의 공유 데이터 채널은 그래픽스 프로세싱 파이프라인 (24) 의 2개 이상의 스테이지들에 의해 발생되는 데이터를 저장할 수도 있다. 그래픽스 프로세싱 파이프라인 (24) 의 개개의 스테이지들에 대해 데이터 채널들을 할당하는 것과는 반대로, 공유 데이터 채널들 (50) 에서의 공유 데이터 채널을 공유함으로써, 그래픽스 프로세싱 파이프라인 (24) 에서의 스테이지가 더 적은 데이터를 발생하면, 동일한 공유 데이터 채널을 공유하는 다른 스테이지는 공유 데이터 채널에서 발생하는 데이터 중 많은 데이터를 저장함으로써, 그 사실을 이용가능할 수도 있다.

[0046] 본 개시물의 일 양태에 따르면, 지오메트리 프로세싱 유닛 (GPC) (42) 은 공유 데이터 채널들 (50) 의 상태에 기초하여 셰이더 프로세서 클러스터 (46) 의 실행을 스케쥴링할 수도 있다. GPC (42) 는 셰이더 프로세서 클러스터 (46) 에 의해 실행될 그래픽스 프로세싱 파이프라인 (24) 의 스테이지들에 의해 소비될 공유 데이터 채널들 (50) 에서의 데이터가 충분한지를 결정하기 위해 공유 데이터 채널들 (50) 을 모니터링할 수도 있다.
GPC (42) 는 또한 셰이더 프로세서 클러스터 (46) 에 의해 실행될 그래픽스 프로세싱 파이프라인 (24) 의 스

테이지들에 의해 발생되는 데이터를 저장하기에 충분한 자유 공간이 공유 데이터 채널들 (50) 에 있는지를 결정하기 위해 공유 데이터 채널들 (50) 을 모니터링할 수도 있다. GPC (42) 가 공유 데이터 채널들 (50) 에 충분한 데이터 및 자유 공간이 있다고 결정하면, GPC (42) 는 실행 지령들을 셰이더 프로세서 클러스터 (46) 로 전송하여, 그래픽스 프로세싱 파이프라인 (24) 의 스테이지들의 배치 (batch) 를 실행할 수도 있다. 스테이지들의 배치의 실행을 완료하는 것에 응답하여, 셰이더 프로세서 클러스터 (46) 는 프로세서 클러스터 (46) 가 스테이지들의 배치의 실행을 완료하였다는 것을 표시하는 신호를 GPC (42) 로 전송할 수도 있다. 이에 응답하여, 데이터 채널 관리기 (44) 는 공유 데이터 채널들 (50) 에 대한 관련된 판독 및 기록 포인터들을 업데이트할 수도 있다. GPC (42) 는 공유 데이터 채널들 (50) 을 관리하는 데이터 채널 관리기 (44) 를 포함할 수도 있다. 데이터 채널 관리기 (44) 는 공유 데이터 채널들 (50) 에 데이터를 기록하고 그로부터 데이터를 판독하기 위해 공유 데이터 채널들 (50) 내 로케이션들을 가리키는 공유 데이터 채널들 (50) 에 대한 판독 및 기록 포인터들을 관리할 수도 있다.

[0047] 본 개시물의 양태들에 따르면, 공유 데이터 채널 (50A) 은 공유 데이터 채널 (50A) 이 그래픽스 프로세싱 파이프라인 (24) 의 제 1 스테이지에 의해 출력된 데이터 (55A) 및 그래픽스 프로세싱 파이프라인 (24) 의 제 2 스테이지에 의해 출력된 데이터 (55B) 양쪽을 저장할 수 있도록, 그래픽스 프로세싱 파이프라인 (24) 의 2개 이상의 스테이지들에 의해 공유되는 데이터 채널일 수도 있다. 공유 데이터 채널 (50A) 은 데이터 (55A 및 55B) 가 양쪽다 그들이 발생되고 및/또는 소비될 때 그의 사이즈를 동적으로 증가시키고 감소시킴으로써 공유 데이터 채널 (50A) 에 할당된 메모리 블록의 좀더 효율적인 사용을 가능하게 할 수 있도록, 링 버퍼일 수도 있다. GPC (42) 는 기록 포인터들 (51A 및 51B) 및 판독 포인터들 (53A 및 53B) 을 관리할 수도 있다. 기록 포인터 (51A) 는 데이터 (55A) 를 기록할 공유 데이터 채널 (50A) 의 메모리 로케이션을 가리킬 수도 있으며, 판독 포인터 (53A) 는 데이터 (55A) 를 판독할 공유 데이터 채널 (50A) 의 메모리 로케이션을 가리킬 수도 있다.

[0048] 일반적으로, GPU (12) 는, 판독 포인터들 (53A 및 53B) 이 큐의 헤드로서 종종 지칭되는, 데이터 (55A 및 55B) 에서의 데이터의 가장 오래된 조각을 저장하는 공유 데이터 채널 (50A) 의 메모리 로케이션들을 각각 가리키도록, 그리고 기록 포인터들 (51A 및 51B) 이 큐의 미부 (tail) 로서 종종 지칭되는, 데이터 (55A 및 55B) 에서의 데이터의 가장 새로운 조각을 저장하는 공유 데이터 채널 (50A) 의 메모리 로케이션들을 각각 가리키도록, 데이터 (55A 및 55B) 를 공유 데이터 채널 (50A) 에서 선입선출 (FIFO) 순서로 저장한다.

[0049] 공유 데이터 채널 (50A) 은 또한 데이터 (55A 및 55B) 로부터 판독된 데이터가 공유 데이터 채널 (50A) 로부터 삭제되고 그들 메모리 로케이션들이 할당 해제될 수 있도록 FIFO 모드에서 동작할 수도 있다. 볼 수 있는 바와 같이, GPU (12) 가 공유 데이터 채널 (50A) 로부터 데이터 (55A) 를 판독할 때, 공유 데이터 채널 (50A) 에서의 자유 공간 (57) 이 증가함으로써, GPU (12) 가 데이터를 데이터 (55B) 에 기록하기 위한 공유 데이터 채널 (50A) 에서의 추가적인 공간을 허용가능하게 한다. 이와 유사하게, GPU (12) 가 공유 데이터 채널 (50A) 로부터 데이터 (55B) 를 판독할 때, 공유 데이터 채널 (50A) 에서의 자유 공간 (59) 이 증가함으로써, GPU (12) 가 데이터를 데이터 (55A) 에 기록하기 위한 공유 데이터 채널 (50A) 에서의 추가적인 공간을 허용가능하게 한다. 단지 공유 데이터 채널 (50A) 이 위에서 자세하게 설명되었지만, 공유 데이터 채널들 (50) 에서의 각각의 공유 데이터 채널이 공유 데이터 채널 (50A) 에 대해 위에서 설명된 특징들을 공유할 수도 있는 것으로 이해되어야 한다.

[0050] 도 4 는 그래픽스 프로세싱 파이프라인 (24) 에서 사용하고 있는 공유 데이터 채널들 (50) 의 일 예를 예시하는 블록도이다. 도 4 에 나타낸 바와 같이, 공유 데이터 채널 (50A) 은 스테이지들에 의해 발생되는 데이터를 저장하기 위해 그래픽스 프로세싱 파이프라인 (24) 의 스테이지들에 의해 공유될 수도 있다.

[0051] 구체적으로 설명하면, 공유 데이터 채널 (50A) 은 그래픽스 프로세싱 파이프라인 (24) 의 헐 셰이더 스테이지 (30) 에 의해 발생되는 데이터 (52) 를 저장할 수도 있으며 그래픽스 프로세싱 파이프라인 (24) 의 지오메트리 셰이더 스테이지 (36) 에 의해 발생되는 데이터 (54) 를 추가로 저장할 수도 있다. 데이터 (52) 는 그래픽스 프로세싱 파이프라인 (24) 의 도메인 셰이더 스테이지 (34) 에 의해 소비될 수도 있으며, 데이터 (54) 는 그래픽스 프로세싱 파이프라인 (24) 의 픽셀 셰이더 스테이지에 의해 소비될 수도 있다.

[0052] 헐 셰이더 스테이지 (30) 및 지오메트리 셰이더 스테이지 (36) 에 의해 공유 데이터 채널 (50A) 에 저장되는 데이터 (52) 및 데이터 (54) 는 각각 헐 셰이더 스테이지 (30) 에 의해 출력되는 패치 제어 포인트들 및 지오메트리 셰이더 스테이지 (36) 에 의해 출력되는 정점들을 포함할 수도 있다. 데이터 채널 (50A) 이 데이터 (52 및 54) 를 캐시하지 않기 때문에, 데이터 (52 및 54) 는 각각 데이터 (52 및 54) 로부터 판독된 데이터가 공유 데이터 채널 (50A) 로부터 삭제되는 FIFO 큐로서 기능할 수도 있다.

[0053] 일부 예들에서, 그래픽스 프로세싱 파이프라인 (24) 의 일부 스테이지들에 의해 발생되는 동일한 데이터는 그래픽스 프로세싱 파이프라인 (24) 의 다른 스테이지들에 의해 다수회 소비될 수도 있다. 데이터가 FIFO 큐로서 기능하는 공유 데이터 채널들 (50) 중 하나에 저장되었다면, 데이터를 발생시키는 그래픽스 프로세싱 파이프라인 (24) 의 스테이지들은, FIFO 큐에 저장된 데이터가 FIFO 큐로부터 판독될 때 삭제될 수도 있기 때문에, 동일한 데이터를 발생시키기 위해 다수회 실행할 필요가 있을 수도 있다. 동일한 정점을 다수회 발생시키기 위해 정점 셰이더 (28) 또는 도메인 셰이더 (34) 를 다수회 실행하는 대신, GPU (12) 는 정점 셰이더 (28) 및 도메인 셰이더 (34) 에 의해 발생되는 데이터를 캐시 모드 공유 채널 (56) 에서 대신 캐시할 수도 있다.

[0054] 예를 들어, 정점 셰이더 스테이지 (28) 에 의해 변환된 정점들을 포함한, 그래픽스 프로세싱 파이프라인 (24) 의 정점 셰이더 스테이지 (28) 에 의해 발생되는 데이터는, 그래픽스 프로세싱 파이프라인 (24) 의 헐 셰이더 스테이지 (30) 에 의해 소비될 수도 있다. 이와 유사하게, 도메인 셰이더 스테이지 (34) 에 의해 출력된 정점 위치들과 같은, 그래픽스 프로세싱 파이프라인 (24) 의 도메인 셰이더 스테이지 (34) 에 의해 발생되는 데이터는 그래픽스 프로세싱 파이프라인 (24) 의 지오메트리 셰이더 스테이지 (36) 에 의해 소비될 수도 있다. 예를 들어, 인접한 프리미티브들 (예컨대, 삼각형들) 이 정점들을 공유할 수도 있기 때문에, 동일한 정점이 2개의 인접한 삼각형들을 형성하기 위해 사용될 수도 있다. 따라서, 정점 셰이더 스테이지 (28) 및 도메인 셰이더 스테이지 (34) 에 의해 발생되는 정점 데이터가 다수회 소비될 수도 있다. 정점 셰이더 스테이지 (28) 및 도메인 셰이더 스테이지 (34) 에 의해 발생되는 데이터가 다수회 소비될 수도 있기 때문에, 이들 스테이지들에 의해 발생되는 데이터는 캐시된 데이터가 캐시 모드 공유 채널 (56) 로부터 판독되는 것에 응답하여 삭제되지 않도록, 캐시 모드 공유 채널 (56) 에서 캐시될 수도 있다.

[0055] 도 5 는 캐시 모드 공유 채널 (56) 을 예시하는 블록도이다. 도 5 에 나타낸 바와 같이, 캐시 모드 공유 채널 (56) 은 2개의 공유 데이터 채널들: 공유된 프리미티브 큐 (50B) 및 공유된 정점 캐시 (50C) 뿐만 아니라, 캐시 윈도우 (70) 를 포함할 수도 있다. 공유된 정점 캐시 (50C) 는, 공유된 정점 캐시 (50C) 에 저장된 데이터가 공유된 정점 캐시 (50C) 로부터 판독 시 삭제되지 않도록, 캐시 모드에서 동작할 수도 있다. 공유된 프리미티브 큐 (50B) 에 저장된 데이터 (62) 및 데이터 (64) 는 정점 셰이더 스테이지 (28) 및 도메인 셰이더 스테이지 (34) 에 의해 발생되는 프리미티브 데이터를 포함할 수도 있다. 예를 들어, 데이터 (62) 는 각각의 프리미티브에 대해 정점 셰이더 스테이지 (28) 에 의해 발생되는 공유된 정점 캐시 (50C) 에 저장되는 정점 데이터의 로케이션들 및 정점 인덱스들을 포함할 수도 있으며, 데이터 (64) 는 각각의 프리미티브에 대해 도메인 셰이더 스테이지 (34) 에 의해 발생되는 공유된 정점 캐시 (50C) 에 저장되는 정점 데이터의 로케이션들 및 정점 인덱스들을 포함할 수도 있다. 데이터 (62 및 64) 는 또한 연관된 프리미티브들의 각각에 대한 할당해제 플래그들을 포함할 수도 있다. 공유된 정점 캐시 (50C) 에 저장되는 데이터 (66) 는 정점 셰이더 스테이지 (28) 에 의해 변환된 정점들을 포함할 수도 있으며, 한편 공유된 정점 캐시 (50C) 에 저장되는 데이터 (68) 는 도메인 셰이더 스테이지 (34) 에 의해 출력된 정점 위치들을 포함할 수도 있다. GPC (42) 는 캐시 모드 공유 채널 (56) 이 데이터를 수용하기에 충분한 자유 공간을 가지고 있는지를 결정하기 위해 공유된 프리미티브 큐 (50B) 및 공유된 정점 캐시 (50C) 양쪽의 자유 공간을 체크할 수도 있다.

[0056] 캐시 윈도우 (70) 는 특정의 정점이 공유된 정점 캐시 (50C) 의 제한된 윈도우에 이미 저장되어 있는지의 표시를 저장할 수도 있다. 예를 들어, 캐시 윈도우 (70) 는 완전 연관된 캐시로서 기능하며, 정점 인덱스, 공유된 정점 캐시 (50C) 내 정점의 데이터 로케이션, 및 플래그와 같은, 정점을 소비할 수도 있는 셰이더의 표시를 저장할 수도 있다.

[0057] GPC (42) 는 지오메트리를 프리미티브 단위로 (primitive by primitive) 프로세싱한다. 정점 셰이더 (28) 및 도메인 셰이더 (34) 에 대해, GPC (42) 가, 정점 인덱스, 및/또는 정점이 속하는 셰이더에 관해 캐시 윈도우 (70) 를 체크하는 것에 기초하여, 프리미티브의 특정의 정점이 공유된 정점 캐시 (50C) 에 있지 않다고 결정하면, 캐시 미스 (miss) 가 일어날 수도 있으며, GPC (42) 는 지령들을 셰이더 유닛들 (46) 로 전송하여, 적합한 셰이더 스테이지 (예컨대, 정점 셰이더 (28) 또는 도메인 셰이더 (34)) 를 실행하여 원하는 정점을 발생하고 그 발생된 정점 데이터를 캐시 모드 공유 채널 (56) 에 저장할 수도 있다. GPC (42) 는 공유된 정점 캐시 (50C) 에서의 정점 데이터의 로케이션들 및 정점 인덱스들을 공유된 프리미티브 큐 (50B) 에 추가할 수도 있다.

GPC (42) 는 캐시 모드 공유 채널 (56) 에서 지금 캐시된 정점에 대한 적합한 데이터로 캐시 윈도우 (70) 를 증가시킬 수도 있다. 캐시 윈도우 (70) 는, 캐시 미스 후 캐시 윈도우 (70) 에 어떤 여유도 없으면 캐시 윈도우 (70) 에서 가장 오래된 슬롯과 연관되고 공유된 프리미티브 큐 (50B) 에서 그의 할당해제 플래그 세트를 가지는 정점이 캐시 모드 공유 채널 (56) 에 추가된 최근의 정점에 관한 정보로 설정될 수 있도록, 선입선출 (FIFO) 방식으로 동작할 수도 있다. 그러나, 특정의 정점이 캐시 모드 공유 채널 (56) 에서 캐시된다고 GPC

(42) 가 결정하면, GPC (42) 는 원하는 정점의 공유된 정점 캐시 (50C) 에서의 메모리 로케이션을 이용하여, 공유된 정점 캐시 (50C) 에서의 정점 데이터의 로케이션들 및 정점 인덱스들을 공유된 프리미티브 큐 (50B) 에 추가할 수 있다. 이러한 방법으로, GPU (12) 는 그래픽스 프로세싱 파이프라인 (24) 에서의 스테이지들의 이질적인 프로세싱을 감소시킬 수 있다.

[0058] 헐 셰이더 (30) 및 지오메트리 셰이더 (36) 를 실행하기 위해, GPC (42) 는 공유된 프리미티브 큐 (50B) 및 공유된 정점 캐시 (50C) 양쪽으로부터 데이터를 소비할 수도 있다. GPC (42) 는 공유된 프리미티브 큐 (50B) 로부터 공유된 정점 캐시 (50C) 에서의 정점 데이터의 로케이션들 및 정점 인덱스들을 판독할 수도 있다. GPC (42) 는 그후 공유된 프리미티브 큐 (50B) 로부터 판독된 로케이션들을 이용하여, 공유된 정점 캐시 (50C) 로부터 정점 데이터를 판독할 수도 있다. GPC (42) 는 데이터를 판독한 후, 공유된 프리미티브 큐 (50B) 의 판독 포인터를 이동시킬 수도 있다. 그러나, 다음 프리미티브가 또한 공유된 정점 캐시 (50C) 로부터 방금 판독된 동일한 정점을 이용할 수도 있기 때문에, GPC (42) 는 캐시된 정점이 공유된 정점 캐시 (50C) 로부터 판독된 직후 공유된 정점 캐시 (50C) 의 판독 포인터를 즉시 이동시키지 않을 수도 있다. 정점을 소비하는 프리미티브에 대한 공유된 프리미티브 큐 (50B) 에서의 연관된 할당해제 플래그가 설정되면, GPC (42) 는 판독 포인터들을 이동시키고 캐시 모드 공유 채널 (56) 로부터 정점을 할당 해제하도록 허용될 수도 있다. GPC (42) 는 지령들을 셰이더 유닛들 (46) 로 전송하여, 셰이더 스테이지 (예컨대, 헐 셰이더 (30) 및 지오메트리 셰이더 (36)) 를 실행하여 정점 데이터를 소비하고, 다음 셰이더 스테이지에 대한 정점을 발생하여 그 발생된 정점 데이터를 공유 데이터 채널 (50A) 에 저장할 수도 있다.

[0059] GPC (42) 는 교착상태에 대해 캐시 모드 공유 채널 (56) 및 공유 데이터 채널 (50A) 을 모니터링할 수도 있다. 일 예에서, 교착상태는, 캐시 모드 공유 채널 (56) 이 정점 셰이더 스테이지 (28) 에 의해 발생되는 데이터로 가득 차 있으면, 그리고 공유 데이터 채널 (50A) 이 헐 셰이더 스테이지 (30) 에 의해 발생되는 데이터로 가득 차있으면, 일어날 수도 있다. 이 경우, 헐 셰이더 스테이지 (30) 가 정점 스테이지 (28) 에 의해 발생되는 데이터를 소비하기 때문에, 헐 셰이더 스테이지 (30) 는, 새로 발생되는 데이터를 저장하기 위해 공유 데이터 채널 (50A) 에 어떤 자유 공간도 없기 때문에, 공유 데이터 채널 (50A) 에 저장되는 데이터를 발생하기 위해 정점 셰이더 스테이지 (28) 에 의해 발생되어 캐시 모드 공유 채널 (56) 에 저장되는 데이터를 소비할 수 없다.

더욱이, 캐시 모드 공유 채널 (56) 이 정점 셰이더 스테이지 (28) 에 의해 발생되는 데이터로 가득 차 있고, 그 어떤 데이터도 헐 셰이더 (30) 에 의해 소비되지 않을 수 있기 때문에, 그 어떤 데이터도 도메인 셰이더 (34) 에 의해 발생되는 데이터를 저장하기 위해 캐시 모드 공유 채널 (56) 에 대한 공간을 해방하도록 할당 해제되지 않을 수 있다. 더욱이, 도메인 셰이더 (34) 가 헐 셰이더 스테이지 (30) 에 의해 발생되어 공유 데이터 채널 (50A) 에 저장되는 데이터를 소비하기 때문에, 어떤 헐 셰이더 (30A) 에 의해 발생되어 공유 데이터 채널 (50A) 에 저장되는 데이터도, 공유 데이터 채널 (50A) 이 지오메트리 셰이더 (36) 에 의해 발생되는 데이터를 저장하기 위해 공유 데이터 채널 (50A) 에서의 공간을 해방하도록 도메인 셰이더 (34) 에 의해 소비되지 않을 수 있다.

[0060] 캐시 모드 공유 채널 (56) 과 공유 데이터 채널 (50A) 사이의 교착상태 (deadlock) 상황들을 방지하기 위해, GPC (42) 는 캐시 모드 공유 채널 (56) 및 공유 데이터 채널 (50A) 이 각각 단지 정점 셰이더 (28) 및 헐 셰이더 (30) 에 의해 발생되는 데이터만을 저장하지 않도록, 도메인 셰이더 (34) 및 지오메트리 셰이더 (36) 에 의해 발생되는 데이터를 각각 저장하기 위해 캐시 모드 공유 채널 (56) 및 공유 데이터 채널 (50A) 에 공간을 예비할 수도 있다. GPC (42) 는 예를 들어, 셰이더 클러스터 (46) 에서의 주어진 파들의 개수에 대한 도메인 셰이더 (34) 및 지오메트리 셰이더 (36) 로부터의 출력을 저장하는데 필요한 공간의 양을 결정함으로써, 예비할, 양자의 구성요소들인 공유된 프리미티브 큐 (50B) 및 공유된 정점 캐시 (50C) 에서의 캐시 모드 공유 채널 (56) 의 공간의 양, 및 공유 데이터 채널 (50A) 의 공간의 양을 결정할 수도 있다.

[0061] 도 6 은 그래픽스 프로세싱 파이프라인의 스테이지들에 의해 데이터 채널들을 공유하는 예시적인 프로세스를 예시하는 플로우차트이다. 도 6 에 나타낸 바와 같이, 프로세스는 GPU (12) 에 의해, 그래픽스 프로세싱 파이프라인 (24) 의 적어도 2개의 스테이지들에 의해 공유되는 GPU (12) 의 온칩 그래픽스 메모리 (40) 에서 공유 데이터 채널 (50A) 을 할당하는 단계를 포함할 수도 있다 (502). 프로세스는 GPU (12) 에서의 셰이더 유닛들 (46) 상에서, 그래픽스 프로세싱 파이프라인 (24) 의 적어도 2개의 스테이지들을 실행하는 단계를 더 포함할 수도 있다 (504). 프로세스는 GPU (12) 에 의해, 온칩 그래픽스 메모리 (40) 에서의 공유 데이터 채널 (50A) 에, 셰이더 유닛들 (46) 상에서 실행하는 그래픽스 프로세싱 파이프라인 (24) 의 적어도 2개의 스테이지들에 의해 발생되는 데이터를 저장하는 단계를 더 포함할 수도 있다 (506).

[0062] 일부 예들에서, 프로세스는 GPU (12) 에 의해, 그래픽스 프로세싱 파이프라인 (24) 의 제 2 적어도 2개의 스테

이지들에 의해 공유되는 GPU (12) 의 온칩 그래픽스 메모리 (40) 에서 제 2 캐시 모드 공유 채널 (56) 을 할당하는 단계를 더 포함할 수도 있으며, 여기서, 공유 데이터 채널 (50A) 은 제 1 공유 데이터 채널이다. 일부 예들에서, 프로세스는 GPU (12) 에서의 셰이더 유닛들 (46) 상에서, 그래픽스 프로세싱 파이프라인 (24) 의 제 2 적어도 2개의 스테이지들을 실행하는 단계를 더 포함할 수도 있다. 일부 예들에서, 프로세스는 GPU (12) 에 의해 제 2 캐시 모드 공유 채널 (56) 에, 셰이더 유닛들 (46) 상에서 실행하는 그래픽스 프로세싱 파이프라인 (24) 의 제 2 적어도 2개의 스테이지들의 각각에 의해 발생되는 제 2 데이터를 저장하는 단계를 더 포함할 수도 있다.

[0063] 일부 예들에서, 프로세스는 데이터가 셰이더 유닛들 (46) 상에서 실행할 때 그래픽스 프로세싱 파이프라인 (24) 의 하나 이상의 스테이지들에 의해 소비될 제 1 공유 데이터 채널 (50A) 또는 제 2 캐시 모드 공유 채널 (56) 에서 이용가능하고 그리고 자유 공간이 셰이더 유닛들 (46) 상에서 실행할 때 그래픽스 프로세싱 파이프라인 (24) 의 하나 이상의 스테이지들에 의해 발생되는 데이터를 저장하기 위해 제 1 공유 데이터 채널 (50A) 또는 제 2 캐시 모드 공유 채널 (56) 에서 이용가능하도록, GPU (12) 에 의해, 제 1 공유 데이터 채널 (50A) 또는 제 2 캐시 모드 공유 채널 (56) 의 상태에 적어도 부분적으로 기초하여, 셰이더 유닛들 (46) 에 의한 그래픽스 프로세싱 파이프라인 (24) 의 하나 이상의 스테이지들의 실행을 스케쥴링하는 단계를 더 포함할 수도 있다.

[0064] 일부 예들에서, 그래픽스 프로세싱 파이프라인 (24) 의 적어도 2개의 스테이지들은 정점 셰이더 (28) 및 도메인 셰이더 (34) 를 포함한다. 일부 예들에서, 그래픽스 프로세싱 파이프라인 (24) 의 제 2 적어도 2개의 스테이지들은 헐 셰이더 (30) 및 지오메트리 셰이더 (36) 를 포함한다.

[0065] 일부 예들에서, 프로세스는, 제 1 공유 데이터 채널 (50A) 과 제 2 캐시 모드 공유 채널 (56) 사이에 교착상태를 방지하기 위해 GPU (12) 에 의해, 제 1 공유 데이터 채널 (50A) 및 제 2 캐시 모드 공유 채널 (56) 중 적어도 하나에 자유 공간을 예비하는 단계를 더 포함할 수도 있다.

[0066] 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 임의의 조합으로 구현될 수도 있다. 소프트웨어로 구현되는 경우, 이 기능들은 컴퓨터 판독가능 매체 상에 하나 이상의 명령들 또는 코드로서 저장되거나 또는 송신될 수도 있다. 컴퓨터 판독가능 매체들은 한 장소로부터 또 다른 장소로 컴퓨터 프로그램의 전송을 용이하게 하는 임의의 매체를 포함한 통신 매체들, 또는 컴퓨터 데이터 저장 매체들을 포함할 수도 있다. 데이터 저장 매체는 본 개시물에서 설명하는 기법들의 구현을 위한 명령들, 코드 및/또는 데이터 구조들을 취출하기 위해 하나 이상의 컴퓨터들 또는 하나 이상의 프로세서들에 의해 액세스될 수 있는 임의의 가용 매체들일 수도 있다. 비한정적인 예로서, 이런 컴퓨터 판독가능 매체들은 RAM, ROM, EEPROM, CD-ROM 또는 다른 광디스크 스토리지, 자기디스크 스토리지 또는 다른 자기 저장 디바이스들, 또는 원하는 프로그램 코드를 명령들 또는 데이터 구조들의 형태로 전달하거나 또는 저장하는데 사용될 수 있고 컴퓨터에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 접속이 컴퓨터 판독가능 매체로 적절히 지칭된다. 예를 들어, 소프트웨어가 웹사이트, 서버, 또는 다른 원격 소스로부터 동축 케이블, 광섬유 케이블, 연선, 디지털 가입자 회선 (DSL), 또는 무선 기술들, 예컨대 적외선, 라디오, 및 마이크로파를 이용하여 송신되면, 동축 케이블, 광섬유 케이블, 연선, DSL, 또는 무선 기술들 예컨대 적외선, 라디오, 및 마이크로파가 그 매체의 정의에 포함된다. 디스크 (disk) 및 디스크 (disc) 는, 본원에서 사용될 때, 컴팩트 디스크 (CD), 레이저 디스크, 광 디스크, 디지털 다기능 디스크 (DVD), 플로피 디스크 및 Blu-ray 디스크를 포함하며, 디스크들 (disks) 은 데이터를 자기적으로 보통 재생하지만, 디스크들 (discs) 은 레이저로 데이터를 광학적으로 재생한다. 앞에서 언급한 것들의 결합들이 또한 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

[0067] 코드는 하나 이상의 디지털 신호 프로세서들 (DSP들), 범용 마이크로프로세서들, 주문형 집적회로들 (ASIC들), 필드 프로그래밍가능 로직 어레이들 (FPGA들), 또는 다른 등가의 집적 또는 이산 로직 회로와 같은, 하나 이상의 프로세서들에 의해 실행될 수도 있다. 따라서, 용어 "프로세서" 및 "프로세싱 유닛" 은, 본원에서 사용될 때 전술한 구조 중 임의의 구조 또는 본원에서 설명하는 기법들의 구현에 적합한 임의의 다른 구조를 지칭할 수도 있다. 게다가, 일부 양태들에서, 본원에서 설명하는 기능은 인코딩 및 디코딩을 위해 구성되는 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공되거나, 또는 결합된 코덱에 포함될 수도 있다. 또한, 이 기법들은 하나 이상의 회로들 또는 로직 엘리먼트들로 전적으로 구현될 수 있다.

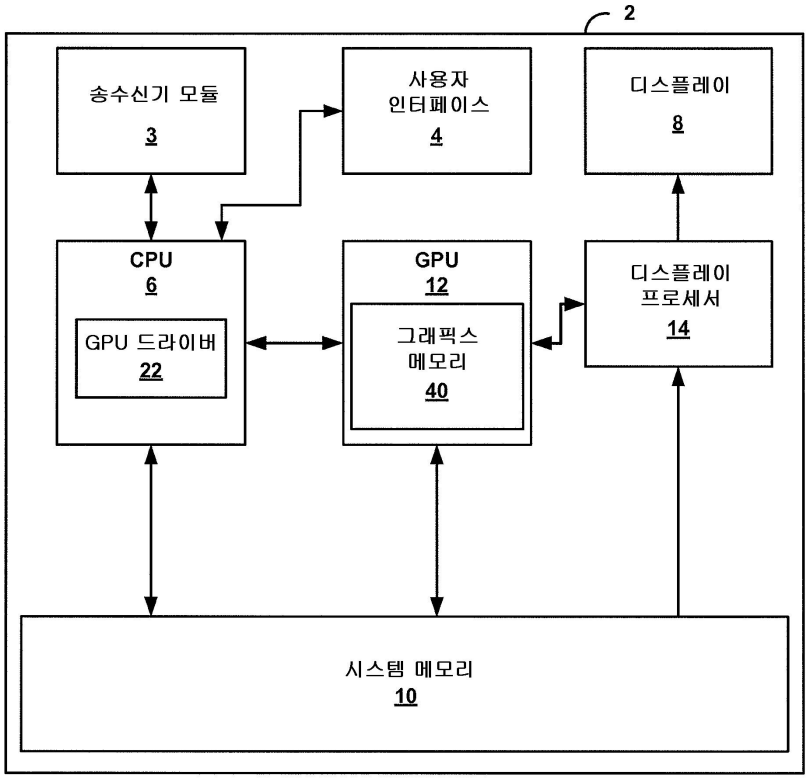
[0068] 본 개시물의 기법들은 무선 핸드셋, 집적 회로 (IC) 또는 IC들의 세트 (즉, 칩 세트) 를 포함한, 매우 다양한 디바이스들 또는 장치들로 구현될 수도 있다. 개시한 기법들을 수행하도록 구성되는 디바이스들의 기능적 양태들을 강조하기 위해서 여러 구성요소들, 모듈들 또는 유닛들이 본 개시물에서 설명되지만, 상이한 하드웨어

유닛들에 의한 실현을 반드시 필요로 하지는 않는다. 대신, 위에서 설명한 바와 같이, 여러 유닛들이 코덱 하드웨어 유닛에 결합되거나 또는 적합한 소프트웨어 및/또는 펌웨어와 함께, 위에서 설명한 바와 같은 하나 이상의 프로세서들을 포함한, 상호작용하는 하드웨어 유닛들의 컬렉션으로 제공될 수도 있다.

[0069] 여러 예들이 설명되었다. 이들 및 다른 예들은 다음 청구항들의 범위 이내이다.

도면

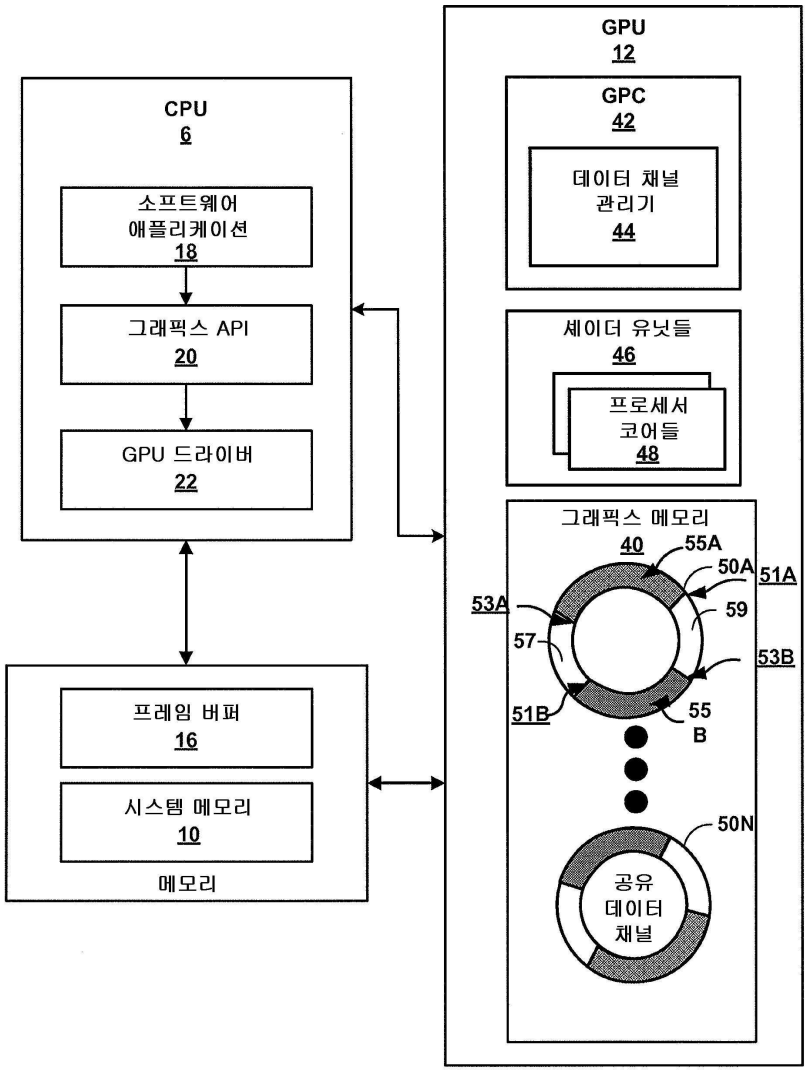
도면1



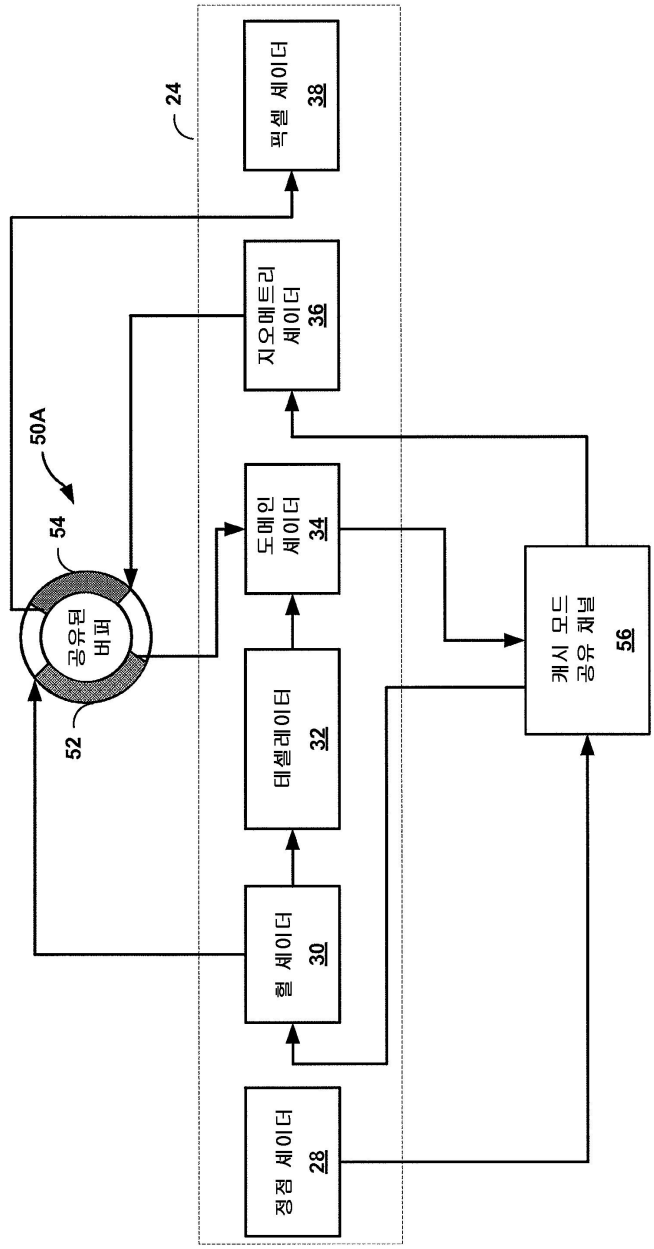
도면2


그래픽스 프로세싱
파이프라인
24
입력-어셈블러
26
정점 셰이더
스테이지
28
헐 셰이더 스테이지
30
테셀레이터 스테이지
32
도메인 셰이더
34
지오메트리 셰이더
36
래스터라이저 스테이지
37
픽셀 셰이더
38

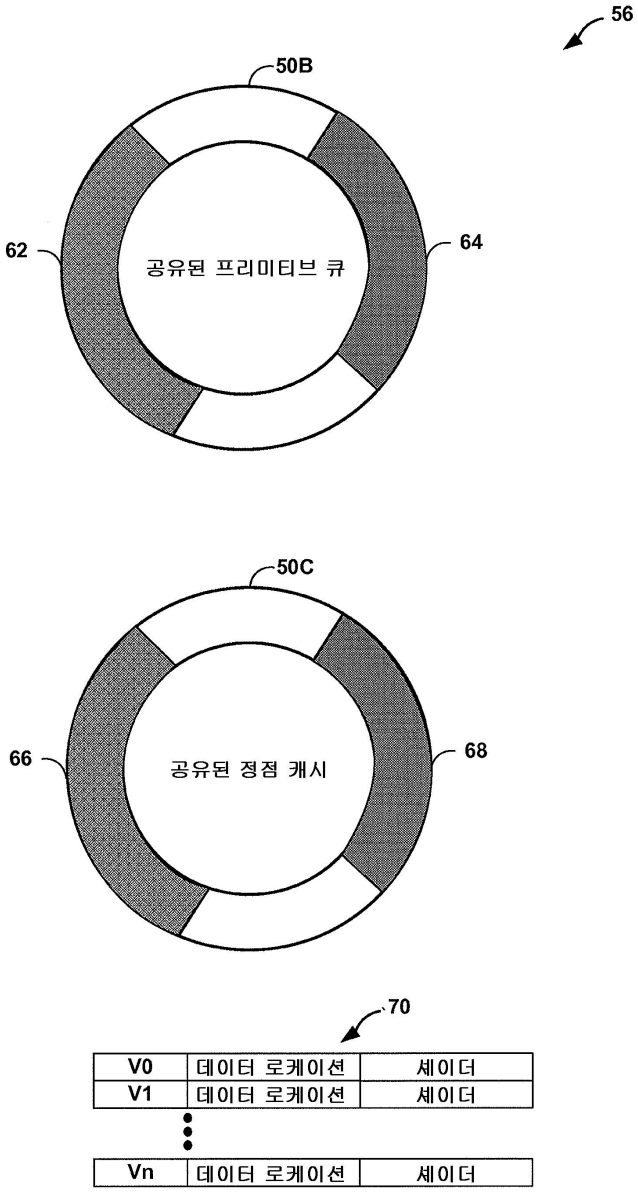
도면3


GPU
12
GPC
42
데이터 채널
관리기
44
셰이더 유닛들
46
프로세서
코어들
48
그래픽스 메모리
40
55A
50A
51A
53A
59
57
53B
51B
55
B
50N
공유
데이터
채널
CPU
6
소프트웨어
애플리케이션
18
그래픽스 API
20
GPU 드라이버
22
프레임 버퍼
16
시스템 메모리
10
메모리

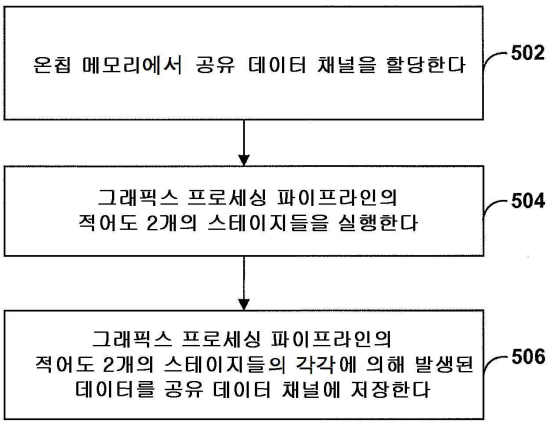
도면4


24
픽셀 세이더
38
지오메트리
세이더
36
50A
54
도메인
세이더
34
공유된
버퍼
52
테셀레이터
32
캐시 모드
공유 채널
56
헐 세이더
30
정점 세이더
28

도면5


56
50B
62
64
공유된 프리미티브 큐
50C
66
68
공유된 정점 캐시
70
V0
데이터 로케이션
셰이더
V1
데이터 로케이션
셰이더
Vn
데이터 로케이션
셰이더

도면6


온칩 메모리에서 공유 데이터 채널을 할당한다
502
그래픽스 프로세싱 파이프라인의
적어도 2개의 스테이지들을 실행한다
504
그래픽스 프로세싱 파이프라인의
적어도 2개의 스테이지들의 각각에 의해 발생된
데이터를 공유 데이터 채널에 저장한다
506