(19) 대한민국특허청(KR)
(12) 특허공보(B1)

(51) Int. Cl.[6]
G06F 3/14

(45) 공고일자 1996년03월04일
(11) 공고번호 특1996-0003041

(21) 출원번호	특1992-0023469	(65) 공개번호	특1993-0013968
(22) 출원일자	1992년12월07일	(43) 공개일자	1993년07월22일
(30) 우선권주장	813,318 1991년12월24일 미국(US)		
(71) 출원인	인터내셔널 비지네스 머신즈 코포레이션 존 디. 크래인 1996년03월04일		
(72) 발명자	벵그트-올라프 쉬나이더 미국, 뉴욕 10562, 오시닝, 차알스 플레이스 2		
(74) 대리인	이병호, 최달용		

심사관 : 홍순우 (책자공보 제4359호)

(54) 그래픽 시스템용의 확장 가능한 다영상 버퍼

요약

내용 없음.

대표도

도1

명세서

[발명의 명칭]

그래픽 시스템용의 확장 가능한 다영상 버퍼

[도면의 간단한 설명]

제1도는 복수의 영상버퍼를 구비한 그래픽스 시스템의 전체 구성을 도시한 블럭 다이어그램.

제2도는 본 발명에 따라 구성된 다영상 버퍼(M 버퍼) 구성(Multiple Image Buffer architecture)의 블럭 다이어그램.

제3도는 본 발명의 일 태양인 픽셀 버스(Pixel Bus)를 보다 상세히 보여주는 도면.

제4a도는 M 버퍼의 기본적인 동작 순서를 예시하는 도면.

제4b도는 M 버퍼에 의해 성취되는 간접적인 픽셀 어드레싱(indirect Pixel Addressing)을 위한 확장된 동작 순서를 예시하는 도면.

제5a도는 M 버퍼의 한 구성 요소인 표면 버퍼(Surface Buffer)의 전체 블럭 다이어그램.

제5b도는 제5a도의 표면 버퍼를 보다 상세히 보여주는 블럭 다이어그램.

제5c도는 표면 버퍼 콘트롤러(Surface Buffer controller)의 구조를 보다 상세히 보여주는 블럭 다이어그램.

제5d도는 표면 버퍼 데이타 경로(data path)의 구조를 예시하는 도면.

제5e도는 표면 버퍼 시험 유닛(test unit)의 구조를 보다 상세히 보여주는 도면.

제5f도는 표면 버퍼의 결정 테이블(Decision Table)의 구조를 보다 상세히 보여주는 도면.

제6도는 역시 M 버퍼의 한 구성 요소인 제어 버퍼(Control Buffer)의 블럭 다이어그램.

제7a도는 판독 액세스(read access)를 위한 대기 사이클(wait cycle)을 가지는 픽셀 버스의 타이밍 다이어그램(timing diagram).

제7b도는 기록 액세스를 위한 대기 사이클을 가지는 픽셀 버스의 타이밍 다이어그램.

제7c도는 리프레시 사이클(refresh cycle)을 보여주는 픽셀 버스의 타이밍 다이어그램.

제8a도 및 제8b도는 픽셀 메모리의 한 조직(organization)을 보여주는 바, 제8A도는 디스플레이 스크린상의 픽셀의 배열을 예시한 도면이고, 제8b도는 픽셀 메모리내의 픽셀들의 해당위치를 예시한 도면.

제9a도는 M 버퍼의 한 구성 요소인 어드레스 매니저(Address Manager)의 구조를 보여주는 블럭 다이어그램.

제9b도는 어드레스 매니저의 어드레스 발생기(Address Generator)의 구성을 보다 상세히 보여주는 블럭 다이어그램.

제9c도는 어드레스 매니저 오프셋 유닛(Address Manager Offset unit)의 블럭 다이어그램.

제10도는 가상 버퍼(Virtual Buffer)의 일실시예를 보여주는 블럭 다이어그램.

제11도는 M 버퍼를 채용하는 어플리케이션(application)의 동작을 예시하는 플로우챠트.

제12도는 본 발명에 따라 구성된 프레임 버퍼(frame buffer)의 블럭 다이어그램.

* 도면의 주요부분에 대한 부호의 설명

20 : 주사 변환(레지스터화 유닛) 22 : 그래픽 버퍼 유닛(M 버퍼)

32 : 표면 버퍼 모듈 34 : 제어 버퍼 모듈

36 : 가상 버퍼 38 : 픽셀 버퍼

39 : 버스 제어부 40 : 결정 테이블

46 : 버퍼 메모리 74 : 비교기

76 : 시험 레지스터 78 : 기준 레지스터

82 : 가산기 154 : 파이프 라인 레지스터

[발명의 상세한 설명]

본 발명은, 일반적으로는, 그래픽 디스플레이 장치 및 방법(graphical display apparatus and method), 좀더 구체적으로 말하면, 고성능 그래픽 시스템용의 영상 버퍼(an image buffer for a high perfomance graphics system)에 관한 것이다.

고성능 그래픽스 시스템은 표면 데이타(surface data)를 기억하기 위하여 버퍼 메모리를 채용한다. 이들 버퍼의 내용은, 가령, 색상 또는 시청자로부터 Z축 방향의 거리(심도(depth))와 같은 대상물 표면(object's surface)의 어떤 특징을 표현한다. 그 데이타는, 이면 제거(hidden surface removal) 또는 안티-얼라이어징(anti-aliasing) 같은 문제를 해결하는 픽셀-배향 알고리듬(pixel-oriented algorithm)을 지원(support)하기 위하여 각 픽셀에 대하여 기억된다. 이들 알고리듬은 메모리 지점(memory location)에 기억된 값을 다른 값과 비교하는등의 비교적 간단한 동작을 실행하는 하드웨어 장치에 의해 지원된다.

전통적으로, 이들 버퍼의 구성은 오히려 제한적이어서 특정의 미리 규정된 용법(usage)만을 허용한다. 이러한 제한은 주로 하드웨어 능력의 결과이다. 이들 제한은 버퍼가 보다 많은 처리능력(processing power)을 갖추는 것을 방해하고, 또한, 보다 넓은 대역폭을 가져 용이하게 액세스가능한 데이타 경로(readily accessible data paths with higer band widhts)를 제공하는 것을 방해한다. 예를들면, 2개의 값을 2개의 다른 버퍼 지점과 동시에 비교하는 것은 대개 불가능하다. 통상의 버퍼 구성의 그러한 제한때문에, 버퍼에 대하여 보다 강력하고 복잡한 동작 및 시험을 필요로 하는 진보된 그래픽 알고리듬에는 일반적으로 그러한 버퍼 구성을 사용할 수 없다.

특히, 많은 과학, 기술 및 의료 분야에서 컴퓨터 그래픽이 필수 불가결한 도구임은 잘 인식되어 있다. CAD 프로그램을 이용하는 엔지니어와 같은 경우 사용자 능률은 그래픽 디스플레이가 사용자의 입력 및 명령에 대화식으로 응답하는(interactively respond) 경우 현저히 향상된다. 그러나, 그래픽 워크스테이션(graphics workstation)의 메인(main) CPU를 이용하여 실행되는 경우 상호 대화식 능력을 제공할만큼 충분히 간단한 알고리듬은 일부에 불과하다. 그러므로, 대부분의 알고리듬 및 어플리케이션(application)이 필요한 실시간(real-time)성능을 유지하기 위하여 필수적인 처리량(throughput) 및 응답시간을 확보하는 데에는 하드웨어 지원이 필요하다.

결국, 컴퓨터 그래픽 워크스테이션은 특정 알고리듬 및 어플리케이션을 지원하는 다종의 하드웨어 프로그램(hardware feature)을 제공한다. 이들 프로그램중에는 4×4 매트릭스 연산과 같은 전형적인 기하학적 계산을 신속히 실행하도록 된 특별한 처리 스테이지(stage)가 있다. 또한, 많은 알고리듬은 특정 데이타 형태들을 기억하도록 할당된 대용량 버퍼 메모리를 설치함으로써, 그리고, 이들 버퍼에 특별한 동작을 제공함으로써 지원된다. 이들 동작은 지원된 알고리듬의 핵심 기능(core functions)을 이행한다. 할당된 그래픽 버퍼에 의해 지원된 기능들의 예로서는, 이면 제거, 안티-얼라이어징, 윈도우잉(windowing) 및, 텍스쳐 맵핑(texture mapping)이 있다.

그래픽 디스플레이상의 각각의 픽셀에 대하여 기억하는 메모리 또는 그래픽 버퍼는 모든 래스터 그래픽 시스템(raster graphics system)의 일부를 형성한다. 전통적으로, 그러한 버퍼들은 그래픽 스크린상에 디스플레이될 정보들을 기억했으므로, 프레임 버퍼(frame buffer) 또는 디스플레이 메모리로서 알려져 있었다.

보다 진보된 워크스테이션에는, 예컨대, IEEE Computer Graphics & Applications, 1989년 7월호 제9권, 제4번, 71-83면의 “The Silicon Graphics 4D/240 GTX Superworkstation”에서 커트 아켈레이가 설명한 바와 같이, 그리고, Computer Graphics & Applications, 1989년 7월호 제4권, 제9번 제56-62면의 “Graphics Processing on a Graphics Supercomputer”에서 브루스 에스. 보덴이 설명한

바와 같이 부가적인 픽셀 메모리가 장착된다. 이 부가적인 픽셀 메모리가 특정의 예정 기능들을 지원한다. 다음에는 몇가지의 보다 대중화된 기능 및 그들이 픽셀 버퍼들에 의해 지원되는 방법을 설명하기로 한다.

[이면 제거]

픽셀 버퍼는, 1974년 유타주, 솔트 레이크시티 소재, 유타 대학교, 컴퓨터 과학과, 에드윈 캣멀(Edwin Catmul)의 PhD 논문 "A Subdivision Algorithm for Computer Display of Curved Surfaces" (보고서 UTEC-CSc-74-133로서 발행됨)에서 유래한 Z-버퍼(심도 버퍼)를 지원한다. 각각의 픽셀에 대해서, 시청자에게 가장 근접한 표면의 심도를 제공하는 심도치(depth Value)가 기억된다. 어떤 픽셀의 심도치 및 색값(color value)은, 신규한 심도치가 기억된 심도치보다 적은 경우에만 교체되어 관련 픽셀이 선행 픽셀의 정면에 있음을 나타낸다.

[엔티-얼라이징/투명도]

픽셀 메모리에는, 적-녹-청(Red-Green-Blue ; RGB) 색값에 추가하여, 적용 범위 또는 투명도 계수(a coverage or transparency coefficient ; 알파)가 기억된다. 신규한 색이 픽셀 메모리에 기록되는 경우, 그것은 기억된 알파 및 새로운 알파의 값들을 기초로한 선행의 색과 조합된다. 이 방법은 ACM Computer Graphics(Proc. Siggraph) 1985년 7월호, 제19권, 제3호, 제41-44면의 "Compositing 3D-Rendered Images"에서 톰 더프(Tom Duff)가 설명한 바와 같이 알파 혼합(alpha-blending) 또는 배합법으로 알려져 있다.

[텍스쳐 맵핑]

텍스쳐라 불리는 포인트 샘플(point sampled) 영상 또는 패턴이 부가된 픽셀 메모리에 기억된다. 그 메모리속의 값들은 디스플레이될 대상물의 특정 파라메타, 가장 간단한 경우에는, 색을 수정하는데 이용된다.

확장된 픽셀 메모리를 이용하는 다른 한 방법은 여분의 제어 비트 평면(extra control bit plane)을 제공하는 것, 즉 별도로 또는 조합해서 액세스되고 조작될 수 있는 개별적인 비트들을 각각의 픽셀에 제공하는 것이다. 이들 비트들은 통상 다른 한 픽셀 버퍼 메모리에 기록하는 것을 제어하는 정보를 기억한다. 제어 비트 평면에 대한 어플리케이션은, 스크린 마스크(screen mask)의 구성, 픽셀 플래그(pixel flag)의 기억 및 픽셀 카운터(pixel counter)를 구성하는 것을 포함한다.

일반적으로, 그래픽 버퍼 구성에는 2개의 요소가 구비된다. 이들 요소는, 그래픽 메모리와 그 메모리상의 동작을 실행하는 관련 데이타 경로이다. 예를들면, Z 버퍼 알고리듬은 심도 메모리(depth memory)에 기억된 심도치를 들어오는 심도치(incoming depth value)와 비교할 것을 요구한다. 그러나, 공지 형태의 그래픽 버퍼 구성(graphics buffer architectures)은 버퍼들이 지원하도록 의도됐던 알고리듬을 실행하는데 필요한 기능성(functionality)만을 제공한다. 환언하면, 그 그래픽 버퍼 구성은 그래픽 버퍼의 용법을 미리 규정한다. 그러나, 신규한 알고리듬 및 기술이 출현함에 따라, 이들 종래의 그래픽 버퍼 구성들은 새로운 요건에 적응할 융통성(flexibility)이 부족하다. 그것들은 일반적으로 신규한 그래픽 방법을 지원할 수는 없으며, 따라서, 그러한 방법의 하드웨어 지원 실행(hardware-supported implementation)을 방해한다.

그러므로, 본 발명의 한 목적은 종래의 시스템의 한계를 극복하는 데이터 경로와 관련된 신규한 구성의 그래픽 버퍼를 제공하는 것이다.

본 발명의 다른 한 목적은 프레임 구조(frame work)내에 버퍼상에서의 동작을 지원하는 모듈형 그래픽 구성(modular buffer architecture)을 제공하는 것이다.

본 발명의 또다른 목적은, 일반적이고 장방형의 버퍼 모듈 및 데이타 경로 배열과 함께 버퍼마다 하나의 프로세서(processor)를 가지는 그래픽 버퍼 구성을 제공하는 것이다.

역시 본 발명의 또다른 목적은, 버퍼마다 하나의 프로세서와, 모듈형 버퍼를 함께 연결시키는 버스 및 주어진 그래픽 어플리케이션에 대해 그래픽 버퍼를 프로그램 가능하게 구성하는 능력을 가지는 모듈형의 프로그램가능한 그래픽 버퍼 구성을 제공하는 것이다.

래스터 그래픽 시스템(raster graphics system)용의 그래픽 버퍼 구성에 의해 전술의 문제점 및 다른 문제점들이 극복되고 본 발명의 목적이 실현된다. 본 구성은 모두 공통의 픽셀 버스에 접속되는 모듈로서 규정되고 이루어진다. 그 모듈들은, 호스트 프로세서에 의해 프로그램되며, 호스트 버스(host bus)를 경유하여 호스트 프로세서와 통한다. 각각의 모듈은 픽셀 버스 및 로컬 메모리(local memory)에 대한 액세스를 가지는 로컬 프로세서 및 메모리를 내장한다. 픽셀 버스는 픽셀 데이타에 대한 시험 결과를 표현하는 복수의 라인(line)을 포함하는 제어 경로와 어드레스 경로 및 데이타 경로를 포함한다.

표면 버퍼, 제어 버퍼 및 가상 버퍼를 포함하는 여러가지 종류의 버퍼 모듈이 있다. 표면 버퍼는 내부의 픽셀 메모리내에는 표면 정보를 기억하고, 픽셀 데이타 버스에 표면 정보를 기록하고, 또한, 그 픽셀 데이타 버스로부터 표면 정보를 판독한다. 제어 버퍼들은, 픽셀 플래그와 같은 각각의 픽셀에 대한 정보 제어를 기억하고, 또한, 픽셀 버스, 구체적으로는, 픽셀 어드레스 어드레스버스와 대화한다(interact). 가상 버퍼들은 픽셀 데이타와 픽셀 어드레스 또는 그들중 하나를 소오스 또는 싱크 작용을 하는(source or sink) 외부 에이전시(external agencies)와 픽셀 버스 사이에 공통 인터페이스를 제공한다.

각각의 버퍼 모듈은 픽셀상의 데이타 관련 내부 메모리로부터의 데이타를 이용하여 시험 및 연산(tests and operations)을 수행한다. 각각의 버퍼 모듈은 다른(가능하기로는 모든) 모듈의 시험 결과를 고려하는 조건하에서 내부 메모리를 업데이트(update)한다. 마찬가지로, 신규한 픽셀치를 계산하는 연산은 타 버퍼 모듈에 의해 실행된 시험에 의하여 영향을 받을 수 있다.

모든 버퍼 모듈은 병렬로 동작하며 프로그램 가능한 기능 및 시험을 실행한다. 모듈들은 그 버퍼 모듈의 동작을 결정하는 상태 변수들을 지정함(specifying)으로써 프로그램된다. 그 모듈들은 일단 프로그램되면 호스트 프로세서와는 독립적으로 작동할 수 있다.

픽셀 버스는 몇개의 서브-버스(sub-bus)를 포함한다. 그 서브 버스들은 하나의 버퍼 모듈에 의해서, 또는, 여러가지 다른 버퍼 모듈에 의해 제어될 수 있다. 서브-버스를 구동하고 있는(driving) 버퍼 모듈 및 서브-버스를 판독하고 있는 모듈을 선택하는 것은 호스트 프로세스로부터 구성가능하다(configurable).

제어 버퍼 및 표면 버퍼들과 마찬가지로 픽셀 버스에 대하여 같은 인터페이스를 나타내는 가상 버퍼에 의해 버퍼 모듈에 데이타가 입력되고 그 모듈로부터 데이타가 출력된다(exported).

영상을 표현하는 픽셀들을 나타내는 정보를 기억하는 그래픽 버퍼를 보다 구체적으로 설명하기로 한다. 그 그래픽 버퍼는 복수의 버퍼 모듈을 포함한다. 최소한도의 구성은 종래의 그래픽 버퍼 시스템과 등가의 기능성을 제공하는 구성으로 볼 수도 있다. 예컨대, 3개의 표면 버퍼 모듈을 채용할 수도 있다. 예를들면, 하나의 표면 버퍼 모듈은, 각각의 픽셀에 대한 심도를 기억한다(Z 버퍼). 나머지 2개의 버퍼는 각각의 픽셀에 대한 색 정보를 기억한다(프레임 버퍼). 2개의 색 버퍼(color buffer)는 2중 버퍼 구조(double buffer configuration)를 형성하는바, 한 버퍼로부터 디스플레이로 픽셀 데이타가 출력되면 타 버퍼는 래스터라이저(rasterizer)로부터 픽셀 데이타를 받아서 기억한다.

다시 예를들면, 그래픽 버퍼는 복수의 표면 버퍼 모듈을 포함하는데, 각각의 모듈은 복수의 픽셀에 대한 픽셀 정보를 기억하는 제1메모리를 포함하다. 표면 버퍼 모듈은 제1메모리에 접속되어 그로부터 판독된 픽셀 정보를 수정하고 수정된 픽셀 정보를 제1메모리로 복귀시키는 제1의 프로세서를 각기 포함한다. 그래픽 버퍼는 각각의 픽셀에 대한 타 픽셀 정보를 기억하는 제2메모리와 그 타정보를 수정하는 제2프로세서를 가지는 적어도 하나의 제어 버퍼 모듈을 또한 포함한다. 복수의 표면 버퍼 모듈과 적어도 하나의 제어 버퍼 모듈 사이에는 픽셀 버스가 접속되어 픽셀 데이타 및 픽셀 어드레스를 포함한 픽셀 관련 정보를 그들 사이에서 전달한다. 그 픽셀 버스에는 또한 가상 버퍼 모듈이 접속되어 픽셀 데이타 및 어드레스를 그 버스에 입력하고 그 버스로부터 픽셀 데이타 및 어드레스를 출력한다.

전술된 본 발명의 특징 및 기타의 특징은 후술되는 발명의 상세한 설명을 첨부 도면과 관련하여 읽으면 보다 명확해질 것이다.

전형적인 래스터 그래픽 시스템(10)은, 제1도에 도시된 바와 같이, 메인(호스트) 프로세서(12)와 그래픽 서브 시스템(14 ; graphics subsystem)을 포함한다. 호스트 프로세서(12)는 어플리케이션 프로그램을 실행하고 그래픽 서브 시스템에 그래픽 태스크(graphics task)를 급송한다(dispatch).

그래픽 서브 시스템(14)는 래스터 디스플레이 장치(16)상에 디스플레이하기 위한 기하학적 정보(geometric entities)를 작성하는데 필요한 동작을 수행한다. 본 발명을 설명할 목적으로, 후술되는 기능유닛을 내장하는 그래픽 서브 시스템(14)의 모델을 이용한다. 이 특별한 모델을 본 발명을 실시하는데 대하여 제한적인 관점에서 이해하지 말아야 함을 인식해야 한다.

기하학적 처리 유닛(Geometric Processing Unit ; 18)은 스크린(윈도우) 경계에 대한 클립핑(clipping)뿐아니라 기하학적 투시 변환(geometric and perspective transformations)을 수행한다. 결과적인 그래픽 프리미티브(graphics primitive), 예컨대, 정점(vertices), 선(lines), 삼각형(triangles)등이 스크린 공간 좌표에 표현된다.

그 그래픽 프리미티브들은 주사변환(Scan Conversion(래스터화(Rasterization)) 유닛(20)에 의해 래스터 정보, 즉, 그래픽 프리미티브에 의해 포괄되는(covered) 디스플레이 스크린 픽셀들의 표현(a description display screen pixels)으로 분석된다.

그 픽셀들을 그래픽 버퍼 유닛(22)이 받아서 그 픽셀들을 픽셀-바이-픽셀 체재로(on a pixel-by-pixel basis)로서 기억, 시험 및, 조작한다.

디스플레이 유닛(24)은 그 그래픽 버퍼 유닛(22)으로부터 픽셀을 받아서 그 픽셀을 통상 래스터 스크린인 출력 장치(16)상에 디스플레이되는 정보로 변환한다.

본 발명은, 구체적으로 말해서, 그 그래픽 버퍼 유닛(22)의 신규한 구성에 관한 것이다.

본 명세서에서 사용되는 다음의 용어들은 다음과 같은 의미로서 사용하고자 한다.

버퍼 : 픽셀 정보를 기억하는 메모리를 내장하는 모듈. 본 발명에 따르면, 버퍼는 또한 메모리에 기억된 정보를 관리 및 조작하는 프로세서, 또는, 데이타 경로를 포함한다. 본 명세서에서는 또한 버퍼를 버퍼 모듈이나 픽셀 버퍼라고도 부른다.

버퍼 메모리 : 버퍼 모듈 내부의 픽셀 메모리.

버퍼 프로세서 : 버퍼 메모리내의 데이타를 조작하는 버퍼 모듈 내부의 프로세서(데이타 경로).

픽셀 : 주사 변환 유닛(20)으로부터 받는 각각의 픽셀은, 디스플레이 장치(16)의 스크린상의 픽셀의 지점을 규정하는 픽셀 어드레스에 의하여, 그리고, 관련 픽셀을 지정하는 픽셀 데이타에 의하여 표현된다. 픽셀 데이타는, 단지 예를들자면, RBG(색), 알파(투명도) 및 심도(Z축) 정보로서 표현된다. 또한, 가령 픽셀처리 알고리듬에 의해 사용되는 픽셀 플래그(pixel flags)와 같은 다른 정보가 각각의 픽셀과 연관될 수도 있다.

[M-버퍼구성]

제2도는, 이후에는 M 버퍼(22)라 호칭되는 그래픽 버퍼 유닛(22)의 단순화된 블럭 다이어그램이다. 다음에 구성 요소를 개략적으로 설명하고, 각각의 구성요소에 대해서는 보다 상세히 후술한다.

호스트 버스(30)는, 호스트 프로세서(12)가 M 버퍼(22)를 구성하고 구성 버퍼 모듈을 프로그램하는데 사용하는 인터페이스이다. 복수의 표면 버퍼 모듈(32)이 픽셀 표면 데이타를 저장, 시험 및 조작한다. 하나 이상의 표면 버퍼 모듈(34)이 픽셀 제어 평면을 기억하고 시험함과 아울러 조작한다. 가상 버퍼 모듈(36)은 M 버퍼(22)와, 주사 전환 유닛(20) 및 디스플레이 유닛(24)과 같은 타 구성 요소와의 사이의 인터페이스기능을 한다. 가상 버퍼(36) 및 "실(real)" 버퍼(32,34) 양자 모두는 공통의 프로토콜(common protocol)에 결합하여 픽셀 버스(38)를 액세스한다. 즉, 가상 버퍼(36)는 픽셀 버스(38)를 개제한 통신(communication over the pixel Bus : 38)과 관련하여 버퍼 모듈(32,34)이 하는 바와 같이 기능한다. 그러나, 가상 버퍼(36)는 픽셀 데이타와 픽셀 어드레스 또는 그중의 하나를 버퍼링하는 피지컬 메모리(physical memory)를 필요로 하지는 않지만, 그것이 구비되어 있을 수도 있다. 픽셀 버스(38)는 M 버퍼(22)내의 모든 버퍼 모듈(32,34,36)에 접속되어 픽셀 데이타 및 픽셀 어드레스를 전이(transferring)하고, 픽셀 시험 결과를 전파하며(broadcasting pixel test results), 버퍼 모듈들(32,34,36)의 동작을 동기화하는 데이타 경로를 제공한다.

[픽셀 버스]

제3도에 도시된 바와 같이, 픽셀 버스(38)는 버퍼 모듈들 사이에서 정보를 전달하고 버퍼 모듈들의 동작을 동기화하는 몇개의 서브-버스(sub-buses)로 이루어진다. 현재 본 발명의 바람직한 실시예와 관련하여 이들 서브-버스들을 후술한다.

2개의 픽셀 데이타 버스(PData 1(38a), PData 2(38b))는 표면 버퍼 모듈(32)과 가상 버퍼 모듈(36) 사이에서 픽셀 데이타를 전이한다. 버퍼 모듈(32,36)중 하나가 픽셀 데이타 버스(38a,38b)중 하나를 구동한다. 다른 버퍼 모듈(32,36)은 구동된 픽셀 데이타 버스상의 데이타를 판독하거나 무시해버린다. 픽셀 데이타 버스(38a,38b)의 폭(width)은 버퍼 메모리내에 기억된 데이타의 단어 길이의 함수(a function of wordlength)이다. 한정하는 것은 아니나, 현재 바람직한 본 발명의 실시예에 있어서는 픽셀 데이타 버스들(38a,38b)이 각기 32비트의 폭을 가진다.

본 발명의 다른 실시예에서는 2개보다 많거나 또는 적은 수의 픽셀 데이타 버스를 채용할 수 있음을 주목해야 한다.

픽셀 어드레스 버스(PAddr)(38c)는 다른 버퍼 모듈(32,34,36)에 픽셀 어드레스(픽셀 디스플레이 스크린 좌표)를 제공한다. 이들 버퍼 모듈중 하나가 픽셀 어드레스 버스(38c)를 구동하는 한편, 다른 모든 모듈들은 그로부터 어드레스를 받는다. 픽셀 어드레스 버스(38c)의 폭은 의도된 디스플레이 장치(16)상의 모든 픽셀들을 어드레스 하기에 충분하다. 본 발명의 현재 바람직한 실시예에 있어서, 픽셀 어드레스 버스(38c)는 1280×1024픽셀을 어드레스하기에 충분한 갯수인 22개의 어드레스 라인을 가진다. 의도된 어플리케이션에 따라, 22개보다 더 많거나 더 적은 어드레스 라인을 구비할 수도 있다.

픽셀 결과(Pixel Result ; 이후에는 PResult) 버스(38d)는 시험결과를 각각의 버퍼 모듈로부터 다른 모든 버퍼 모듈로 전파하는 복수-신호 라인 버스(multi-signal line bus)이다. 픽셀 결과 버스(38d)의 각 라인은 하나의 버퍼 모듈에 의해서 구동된다. 다른 버퍼 모듈은 이들 라인에 응답하여 그들의 내부 버퍼 메모리를 업데이트 할 것인지 여부를 결정한다. 즉, 각각의 버퍼 모듈(32,34,36)은 PResult 버스(38d)의 한라인을 구동하고, PResult 버스(38d)의 모든 라인을 받아들인다.

PResult 버스(38d)의 하나 이상의 라인을 구동하는 버퍼 모듈을 갖추는 것은 본 발명의 범위내에 있음을 주지해야 한다. 예를들면, 픽셀 결과 버스(38d)의 2개의 라인을 구동하는 버퍼 모듈이 픽셀 비교 결과를 적다, 같다, 크다와 같이(as being less than, equal to, or greater than) 표현할 수 있다.

픽셀 제어(Pixel control ; 이후에는 PCtrl) 버스(38e)가 버퍼 모듈(32,34,38) 사이의 통신 및 동기화를 위하여 신호를 이송한다. PCtrl버스(38e)는 다음의 신호를 포함한다.

신호 라인 NEW는, 버퍼 모듈이 어서트하면(when asserted by a Buffer Module), 픽셀 버스(38)상에 새로운 픽셀, 즉, 픽셀 어드레스 및 픽셀 데이타가 있음을 나타낸다.

신호 라인 WAIT는, 어서트시, 현재의 픽셀 버스 사이클중에는 모듈(32,34,36)중 최소한 하나가 작동을 완료할 수 없음을 알리고 현 사이클의 연장을 요청한다. 예를들면, 버퍼 모듈(32,34)중 하나는 동적 메모리(dynamic memory)의 새로운 페이지/컬럼에 액세스할 것을 요청받을 수 있는데, 이는 부가적인 메모리 설정시간(memory set-up time)을 필요로 한다.

STEP는 버퍼 모듈, 구체적으로는 후술되는 바와 같은 그 모듈의 내부 파이프라인 스테이지(internal pipeline stages)을 미리 규정된 위상(predefined PHASEs)으로 구동하는 신호 라인이다.

CLOCK은 모든 버퍼 모듈(32,34,36)에 동기화 클럭 신호(synchronizing clock signal)을 분배하는 한 세트의 라인이다. 본 발명의 현재 바람직한 실시예에 있어서, CLOCK은, 비록 다른 주기도 본 발명의 기술적 사상 범위내에 속하기는 하지만, 40나노초(nanoseconds)의 주기를 가진다.

RFSH는, 어서트시, 동적 메모리를 내장하는 버퍼 모듈(32,34)이 메모리 정보를 보존하기 위하여 리프레시 사이클을 수행할 것임을 신호로 알린다.

픽셀 제어 버스(38e)에는 WAIT신호를 받아서 전술된 CLOCK 및 STEP 신호를 발생시키는 버스 제어기(Bus Control ; CNTL)(19)가 접속된다. 버스 CNTL(39)는 또한 RFSH 신호를 발생시킬 수도 있다.

이들 신호 라인들의 동작을 제4a도 및 제4b도에 따라 보다 상세히 설명하기 앞서, M 버퍼(22)에 의해 실행되는 동작의 기본 순서를 설명키로 한다.

단계 A. 어드레스 발생시켜라(Generate Address ; GA) : 픽셀 어드레스 버스(38c)를 구동하는 버퍼 모듈(32,34,36)이 픽셀 어드레스를 발생시켜 그것을 픽셀 어드레스 버스(38c)상으로 구동한다. 어드레스를 발생시키는 버퍼 모듈은 또한 신호 NEW를 발생시킨다. 이 상태는 위상=0, STEP=0에

해당한다.

단계 B. 어드레스를 수정하라(Modify Address ; MA) : 모든 버퍼 모듈은 픽셀 어드레스 버스(38c)로부터 어드레스를 판독하여 그것을 수정할 수 있다. 이 상태는 위상=0,STEP=1에 해당한다.

단계 C. 판독하라(Read ; RD) : 버퍼 모듈은(수정된) 픽셀 어드레스를 사용하여 관련 버퍼 메모리(46)로부터 데이타를 판독한다. 픽셀 버스(38)를 구동하는 버퍼 모듈(들)은 데이타를 픽셀 버스(38)의 서브-버스들(PDATA 1(38a), PDATA 2(38b))중 하나에 위치시킨다. 이 상태는 위상=1, STEP=0에 해당한다.

단계 D. 시험하고 계산하라(Test & Compute : TC) : 표면 버퍼 모듈(32) 또는 가상 버퍼 모듈(36), 또는 양자 모두가 픽셀 버스의 하나 또는 몇개의 서브-버스로부터 픽셀 데이타를 받는다. 모듈은 받은 데이타에 응답하여 다음의 2가지 작용을 수행한다.

단계 D1. 받은 데이타값이 시험되고 제 2 값과 비교된다. 1비트(결과)로서 표현되는 시험 결과가 픽셀 결과 버스(38d)의 관련 라인에 기록되고 다른 모든 버퍼 모듈(32,34,36)에 전파된다. 앞서 설명했던 바와 같이, 본 발명의 다른 실시예에 있어서는 시험결과가 1비트보다 큰 비트로서 표현될 수 있다.

단계 D2. 받은 데이타 값이 단계 F(후술)에서 관련 버퍼 메모리(46)에 기록될 수 있는 새로운 값을 계산하는데 이용된다.

단계 D는 위상=1, STEP=1에 해당한다.

단계 E. 결정하고 수정하라(Decide & Modift ; DM) : 모든 버퍼 모듈(32,34,36)이 픽셀 결과 버스(38d)를 판독하고 단계 D2에서 계산된 값이 버퍼 메모리(46)에 기록될 것인지 여부를 결정한다. 이 상태는 위상=2, STEP=0에 해당한다.

단계 F. 기록하라(Write ; WR) : 단계 E에서 인에이블되었다면(if enabled) 단계 D2에서 계산된 값이 버퍼 메모리(46)에 기록된다. 이 상태는 위상=2, STEP=1에 해당한다.

전기한 단계들의 완전한 순서는 본 명세서에서 일동작으로서 설명된다. GA+MA, RD+TC 및 DM+WR 단계 각각은 병렬로 실행될 수 있는 스테이지 그룹(stage group)을 형성한다. 본 명세서에서는 이들 스테이지 그룹들을 일동작의 위상들(PHASEs)로서 설명하는 바, 그 스테이지 그룹들중 몇개 또는 모두는 특정 시간에 활성화할 수 있다. 스테이지 그룹의 활성화(activation)는, 콘트롤러(56)와 관련하여 후술되는 3비트 시프트 레지스터(파이프(PIPE)레지스터)(154)에 의해 제어된다. 파이프 레지스터(154)의 각각의 비트는 스테이지 그룹중 하나에 해당한다. 이 시프트 레지스터에 세트된 비트는 현 위상중에 해당 스테이지 그룹이 활성 상태(active)임을 표시한다. 하나 이상의 스테이지 그룹이 동시에 활성상태일 수 있다.

제4a도는 동작을 쾌속화(speed up)하기 위하여 이들 단계들을 3스테이지 파이프라인에 배열하는 방법을 예시한다. 예시된 바와 같이, 3스테이지 파이프라인은 그것이 채워져 있는 경우 매 단계마다 버퍼 메모리(46)에 액세스한다.

제4a도는 모든 버퍼 모듈에 의해 수행되는 동작의 기본 순서를 예시한다. 그러나, 몇가지 어플리케이션은, 다른 한 픽셀 또는 다른 한 버퍼 또는 양자 모두에 기억된 값을 근거로한 픽셀의 어드레스를 결정하는 어드레싱 스킴(addressing scheme)을 필요로 한다. 그러한 액세스 메카니즘은 간접 어드레싱 또는 맵핑이라 불린다. 예를들면, 텍스쳐 맵핑은 버퍼로부터 텍스쳐 좌표를 판독함으로써 스크린상에 디스플레이될 값을 결정한다. 그후, 그 텍스쳐 좌표는 디스플레이되거나 또다시 처리될 값을 가지는 텍스쳐 맵을 내장하는 다른 하나의 버퍼를 어드레스하는데 이용된다.

M 버퍼(22)는 다음과 같은 능력을 제공함으로써 그러한 동작들을 수행한다.

첫째로, 버퍼 메모리(46)로부터 판독된 값들은 픽셀 어드레스 버스(38c)로 향할 수 있다. 이 능력은 버퍼 메모리(46)에 기억된 픽셀값을 인에이블하여 픽셀 어드레스로서 해석될 수 있게 한다.

둘째로, 그 기본적인 동작 순서가 확장되어, 제4b도에 도시된 바와 같이, 간접 어드레싱에 의해 픽셀값을 액세스하기 위하여, 여분의 사이클(exetra cycle)을 수용한다. 제2 및 제3 파이프라인 동작은 각기 부가적인 MA 및 RD 단계를 포함하는 바, 받은 픽셀 어드레스는 버퍼 메모리(46)로부터 또다른 픽셀 어드레스를 추출하는데(to extract) 이용된다.

그러므로 M 버퍼(22)는 몇가지 레벨의 간접적인 픽셀 액세스(pixel accesses with several levels of indirection)를 허용한다. 제4b도는 한 레벨의 간접적인 액세스를 보여줌과 아울러, 동작의 기본 순서를 확장하면 버퍼 모듈 하드웨어의 이용이 감소됨을 보여준다. 즉, 3가지 동작이 기본적인 동작 순서로서 병렬로 수행되거나, 간접 어드레싱을 이용하면, 평균적인 병렬 동작수가 2 이하로 감소된다.

이제, 제7a-7c도에 대해 설명할 것인바, 그들 도면에는 픽셀 버스(38)의 타이밍 특성이 도시되어 있다.

신규한 픽셀 어드레스(NEW 활성)는 파이프라인 위상(위상=0, STEP=0)의 제 1 단계 중에만 어서트된다. 더우기, RFSH 또는 WAIT중 어느것이 활성 상태이면 신규한 픽셀 어드레스는 인가되지 않는다. 제7a 및 7b도에서, WAIT 신호 라인의 어서숀(assertion)은 또다른 STEP신호의 발생을 일시 정지시키는 것으로 도시되어 있는바, 이로써, 하나 이상의 WAIT 상태를 삽입한다. STEP 신호의 제1어서숀은 GA파이프라인 단계를 발생시키는 바, 그 단계에는 MA 파이프라인 단계가 이어지고, 다시 MA 파이프라인 단계에는 RD 파이프라인 단계가 이어진다. 예시된 실시예의 경우, WAIT 상태가, 제1픽셀에 대해, DM과 WR 단계 사이에 삽입된다. 제2픽셀에 대해 WAIT 상태가 RD와 TC 단계 사이에 삽입되며, 제3픽셀에 대해, WAIT 상태가 GA 및 MA 단계 사이에 삽입된다. 즉, 제3픽셀에 대해 실행되는 판독 액세스는 WAIT 신호 라인의 어서숀을 초래한다. 제7b도의 경우, 제1픽셀에 대한 기록 액세스(WR)의 실

행으로 WAIT 신호 라인이 어서트된다.

제7c도에서 알 수 있는 바와 같이, 파이프라인 위상의 제2단계(STEP=1)중에 RFSH를 어서트함으로써 리프레시 사이클이 요청된다(requested). 그 리프레시는 현재의 파이프 라인 단계가 완료되는 즉시 실행된다. RFSH를 디어서트한 후(after deasserting) RFSH, 리프레시 사이클이 완료되고 정규의 동작이 계속된다.

WAIT 신호는 신규한 메모리 페이지를 필요로 하는 메모리 액세스의 개시시에 활성화된다. WAIT 신호는 메모리 액세스가 완료되기전에 1클럭 사이클 디어서트된다. WAIT는 또한 리프레시 동작중에 활성화하여, RFSH신호의 디어서트후 모든 픽셀 버퍼가 진행중인 리프레시 사이클(pending refresh cycle)을 완료할 수 있도록 보장한다.

[표면 버퍼]

표면 버퍼(32)는 M 버퍼(22)용의 범용의 버퍼 리소스(buffer resourc)를 형성한다. 그 버퍼들은 대개 디스플레이될 표면과 연관되는 데이타, 한정하는 것은 아니지만 가령, 심도치, 색 값 및 알파값을 기억한다. 호스트 프로세서(12)에서 가동하는 어플리케이션 프로그램은 이들 버퍼들이 어떻게, 그리고 무슨 목적으로 이용되는지를 결정한다.

제5a도 및 제5b도에 볼 수 있는 바와 같이, 표면 버퍼 모듈(32)은 몇개의 블럭으로 구성된다. 그 블럭은 호스트 버스(30)를 거쳐, 이들 블럭에 내장된 레지스터에 정보를 기록함으로써 지정된 호스트 프로그램 가능한(host-programmable) 기능을 수행한다. 다음에, 표면 버퍼 모듈(32)의 각각의 구성 블럭을 설명한다.

결정 테이블(40)(제5f도)은 메모리 업데이트 신호(memory update signal : MUpdate : 40a)를 제어한다. 결정 테이블은 PResult 버스(38d)로부터 받은 라인이 있는 만큼, 본 발명의 이 실시예의 경우 버퍼 모듈이 있는 만큼, 많은 변수의 부울리안 함수(boolean tunction)를 이행한다. MUpdate(40a) 함수는 픽셀 업데이트 조건이 충족되면 진(眞 : TRUE)이고, 그렇지 않으면 위(僞 : FALSE)이다.

보다 구체적으로 말하면, 모든 버퍼 모듈(32,34,36)의 관련 시험 유닛(42)에 의해 보고된 결과에 의해 형성되는 특정 조건하에서만 버퍼 메모리(46)가 업데이트된다. 그 결과 비트들은 함께 결정 테이블(40)로 들어가는 어드레스를 형성한다. 현재 바람직한 실시예에 있어서, 결정 테이블(40)은 2개의 정적 램(static RAM ; S램 ; 60a,60b)을 포함한다. 결정 테이블(40)을 위해서 이용되는 S램의 용량(size)은 픽셀 버퍼의 최대수의 하수이다. 예컨대, M 버퍼(22)가 n개의 표면 버퍼(32), 1개의 제어 버퍼(34) 및 2개의 가상 버퍼(36)를 포함한다고 가정하면, 그 용량은 $2^{n+3} \times 1(2^{n+3}$ by one)비트이다. S램(60a,60b)은 호스트 버스(30) 인터페이스(62)를 경유하여 로드된다(loaded). Presult 버스(38d)의 하위의 4비트들은, 멀티플렉서(64) 및 논리부(60c)를 경유하여, 램(60)으로부터 판독된 16비트들중 하나를 선택한다. 호스트 프로세서(12)는 레지스터 기록 작용(HWR)을 수행함으로써 램(60)에 데이타를 기록한다. 그 데이타 카운터(counter ; 66)의 내용에 의해 지정된 램(60) 지점에 기록된다. 램(60) 데이타는 또한 선택 신호(A13)와 함께 HRD를 어서트함으로써 그 호스트(Host)에 의하여 판독될 수도 있다. 호스트 어드레스 버스(HAddr : 30b)는 디코더(decoder ; 63)에 인가되어 2개의 선택 신호 A12 및 A123중 하나를 발생시킨다. A12는 호스트 데이타를 HData 버스(30a)로부터 카운터(66)로 로드하는 것을 인에이블한다. A13 선택신호는 레지스터 카운터(register-counter 66)의 내용을 증분함(increment)과 아울러 그것의 출력을 램(60) 어드레스 버스(68)상에 인에이블 한다. A13이 어서트되지 않은 경우, 버퍼(68a)는 PResult 버스(38d)를 어드레스 버스(68)상에 위치시킨다.

또한, 논리부(60c)는 PResult 버스(38d)의 비트 3을 게이트하여(gate) S램(60a)과 S램(60b) 사이에서 선택한다. 멀티플렉서(multiplexer ; 64)의 출력은 PResult 버스(38d)의 비트[0 : 2]에 의해 선택되어 1비트 레지스터(70)내에 기록되고 MUpdate 신호(40a)를 형성한다. 그 MUpdate 신호(40a)는 모든 PResult 버스(38d) 신호 라인들의 논리 상태 및 S램(60a) 또는 S램 (60b)의 어드레스된 지점의 미리 프로그램된 내용에 따라서 0 또는 1이다.

시험 데이타 경로(42)(제5e도)는 3개의 기준(criteria)로부터 2개를 이용하여 픽셀 시험을 실행한다. 구체적으로 말하면, 시험 데이타 경로(42)는 픽셀 버스(38)(서브 버스 PData 1(38a), PData 2(38b)중 하나 또는 양자 모두)의 데이타 내부 데이타 PData 0(38a), 또는 시험 데이타 경로(42)에 기억된 기준치를 이용한다. 시험 결과는 PResult 버스(38d)의 한 라인에 접속되어 그것을 구동하는 신호 결과(42a)로서 표현된다. 그 기준치는 1 또는 2가지 방법으로 이용될 수 있다. 그것은 다른 한 픽셀값에 대한 비교를 위해서, 또는, 비교되는 2개의 값중 하나에 더해지는 공차값(tolorance value)으로서 채용될 수 있다. 비교되는 2개의 데이타 아이템(item)의 소정의 비트는 판독 마스크 레지스터(Readmask register ; 72)의 사용을 통해서 마스크될 수 있다.

보다 구체적으로 설명하면, 시험 데이타 경로(42)는 호스트 버스(30) 및 픽셀 버스(38)에 접속된다. 결과(42a) 출력은 PResult 버스(38d)로 구동된다. 시험 데이타 경로(42)의 동작은 제어부(56)의 상태 레지스터(Status Register ; 152) 및 파이프라인 레지스터(154)(후술됨)로부터 제어된다. 비교기(comparator ; 74)는, 시험 레지스터(76)의 제어하에 표 1에 기재되어 있는 한 세트의 시험을 실행한다.

[표 1]

동 작	표 현
NEVER	비교결과는 항상 FALSE로 복귀한다.
LESS	A<B
LEQUAL	A≤B
	A=B
NEQUAL	A≠B
GEQUAL	A≥B
GREATER	A>B
ALWAYS	비교 결과는 항상 TRUE로 복귀한다.

수행될 특정 시험은 시험 레지스터(76)에 로드되는 호스트 프로세서(30)의 내용에 의해 선택된다. 또한, 시험 레지스터(76)의 내용은 기준 레지스터(Reference Register)(78)에 기억된 값이 제1의 비교 아규먼트(the first argument of the comparison)에 더해지는지 여부를 결정한다. 제1의 아규먼트는, 일반적으로, 버퍼 메모리(46)의 출력인 PDATA 0(32a)로부터 받는다. 32비트 판독 마스크 레지스터(72)에는 비교중에 무시될 비트에 대하여 0값들이 로드된다. 멀티플렉서 그룹(80)은, 가산기(adder : 82)와 함께, 비교기(74)에 A와 B 입력을 제공한다. 비교 결과는 시험 데이타 경로(42)를 PResult 버스(38b)에 인터페이스하는 레지스터(84)에 기록된다. 비교기(74)의 출력은 들어오는 데이타(incoming data)가 타당한 경우, 즉, 파이프 라인 스테이지 TC가 활성 상태이면 레지스터(84)에 기록된다. 이는, 파이프 레지스터(154)의 출력인 위상[1]과 STEP 신호와의 조합에 의해서 신호로서 통보된다.

계산 데이타 경로(44)(제5d도)는 시험 데이타 경로(42)와 같은 입력을 받는다. 이들 입력을 근거로, 계산 데이타 경로(44)는 DataIn(44a)로 표현된 신규한 픽셀값을 계산한다. 그러나, 이 값은 결정 테이블(40)에 의해 결정되는 메모리 업데이트 조건이 충족되는 경우에만 버퍼 메모리(46)에 기록된다.

보다 구체적으로 말하면, 계산 데이타 경로(44)는 소정의 동작에 따라 픽셀값을 변환한다. 이 변환으로 3개의 데이타 요소, 즉, (a) 픽셀 메모리(46)로부터 판독된 픽셀 값. (b) PDATA 1(38a)나 PDATA 2(38b)로부터 판독된 픽셀 값 및 (c) 호스트 프로세서(12)에 의해 로드되는 오퍼랜드 레지스터(90)(Host processor 12 loaded Operand register 90)내의 데이타 경로내에 기억된 오퍼랜드중 어떤 2개를 조합한다. 연산 결과는 역시 호스트 프로세서(12)에 의해 로드되는 기록 마스크 레지스터(Writemask register ; 92)의 내용에 따라 다시 수정된다. 그 동작 자체는 멀티플랙서(94a)를 경유하여 연산 레지스터(Operation register ; 94)의 내용에 의하여 저장된다. 계산 데이타 통로(44)는 표 2에 기재된 소정의 연산을 실행하는 ALU(96)를 포함한다.

[표 2]

연 산	표 현
ADD	A plus B
SUBA	B minus A
SUBB	A minus B
AND	A and B
OR	A or B
XOR	A exor B
NAND	A nand B
NOR	A nor B
NXOR	A exnor B

연산은 ALU(96)와 함께 연산 레지스터(94)의 내용에 의해 선택된다. 보다 간단한 연산, 예컨대, ALU

출력을 0으로 설정하거나 또는 입력들중 하나를 ALU 출력으로 복제하는 것(copying)은 오퍼랜드 레지스터(90)에 적절한 값을 로드함으로써 성취된다.

연산 레지스터(94)의 2개의 비트들은 표 3에 따라 ALU(96)의 입력을 선택한다.

[표 3]

ALU 입력 A	ALU 입력 B
버퍼 메모리(46)	Pdata 1(38a) 또는 Pdata 2(28b)
버퍼 메모리(46)	오퍼랜드 레지스터(90)
오퍼랜드 레지스터(90)	Pdata 1(38a) 또는 Pdata 2(38b)
오퍼랜드 레지스터(90)	오퍼랜드 레지스터(90)

기록 마스크 레지스터(92)에는 0들(zeros)이 로드되는데, 버퍼 메모리(46) 지점의 원내용(original contents)을 변경되지 않게 유지하는 것이 바람직한다. 이는, ALU(96) 둘레에 Pdata 0 버스(32a)를 루팅(routing)하고 멀티 플렉서(98)를 통해서 그것의 선택된 비트를 게이팅(gating)함으로써 성취된다.

최종 결과는 버퍼 메모리(46)에 입력되는 DATaIn 버스(44a)의 소오스 역할을 하는 레지스터(100)에 기록된다. 최종 결과는 들어오는 데이타가 값, 즉, 파이프라인 스테이지 TC가 활성상태(active)인 경우 레지스터(100)에 기록된다. 이는, 위상[1], 콘트롤러(56)내의 전술된 파이프 레지스터(154)의 출력 및, STEP 신호의 조합에 의하여 표시된다.

버퍼 메모리(46)는 픽셀 데이타를 기억한다. 버퍼 메모리(46)는 반도체 메모리, 바람직하기로는, 동적 랜덤 액세스 메모리(DRAM)로 구성된다. 버퍼 메모리(46)는 메모리 어드레스(MAddr) 버스(48a) 및 메모리 제어(MCTrl) 버스(48b)를 경유하여 어드레스 메니저(48)에 의하여 제어되고 액세스된다. 버퍼 메모리(46)에는 계산 데이타 경로(44)에 의하여 DataIn 버스(44a)를 경유하여 데이타가 기록된다. 버퍼 메모리(46)의 출력은 내부 픽셀 데이타 버스(PData 0)(32a)에 접속되는 데이타 출력 버스(46b)이다.

본 발명의 현재 바람직한 실시예에 있어서, 버퍼 메모리(46)는 1024×1024픽셀로서 구성되는데, 각각의 픽셀은 34비트로 표현된다. 버퍼 메모리(46)는, 예컨대, 페이지 모드 액세스(page-mode access)를 제공하는 D램 장치의 32평면(32 planes of DRAM devices)으로 구성된다. 픽셀들은 행을 위주로한 형태로(in row-major form) 버퍼 메모리(46)내에 기억되는바, 즉 각각의 주사선(scanline)은 세그먼트당 256픽셀씩 4개의 세그먼트(segment)로 분할된다. 제8b도에서 볼 수 있는 바와 같이, 각각의 주사선 세그먼트는 버퍼 메모리(46)의 한 페이지에 기억된다.

이 특별한 메모리 구성은, 많은 어플리케이션들이 픽셀 버퍼들간의 픽셀전이를 확장적으로 이용하기 때문에(make extensive use of pxel transfer) 선택된다. 픽셀 전이가 일반적으로 주사선마다(on a scanline-by-scanline basis) 이루어진다는 점에서, 한 주사선내의 픽셀에 효과적으로 액세스한다는 것은 중요하다.

그러나, 여러가지의 많은 버퍼 메모리(46) 구성이 채용될 수 있음을 인식해야 한다. 예컨대, 2개의 인접한 주사선의 주사선 세그먼트를 기억하는데 8뱅크(bank)의 메모리를 사용할 수 있는데, 2개의 주사선의 픽셀 어드레스는 모두 다른 뱅크 각각의 동일 페이지에(in a same page of each of the different banks) 있다. 따라서 제8a도 및 제8b도에 도시된 구성으로 본 발명의 실시를 한정하려고 의도한 것은 아니다.

어드레스 매니저(48)는 픽셀 어드레스를 발생시키고 수정하기 위하여 동작한다. 이들 어드레스는 버퍼 메모리(46)에 그리고, 적절히 구성된 경우(if configured accordingly), 픽셀 어드레스 버스(38c)에 인가된다. 그 어드레스 매니저(48)는 또한 MCtrl 신호(48b)를 발생시킴으로써 버퍼 메모리(46)의 동작을 제어한다. 앞서 설명한 바와 같이, 버퍼 메모리는 신호 Mupdate(40a)가 어서트되는 경우에만 개시된다.

어드레스 매니저(48)는 몇가지의 명령에 응답한다. 이들 명령은 다음과 같다.

GENERATE 스크린 박스를 규정하는 좌표내에 모든 픽셀에 대한 픽셀 어드레스를 발생시켜라 ;

HOLD 스크린 박스 좌표값을 변화되지 않게 유지하라 ;

TRACKAddr PAddr 버스(38c)상에 나타낸 모든 픽셀들의 좌표에 의하여 규정된 경계 박스(bounding box)를 발생시켜라 ;

TRACKUpd 관련 버퍼 메모리내에서 업데이트 됐던 모든 픽셀에 의해 규정된 경계 박스를 발생시켜라.

어드레스 매니저(48)는 또한 리세트 명령에 응답하는바, 좌측 및 상측(LEFT and TOP) 레지스터(50b,50d)에는 큰 값(0xff…f)가 로드되고 우측 및 하측(RIGHT and BOTTOM) 레지스터(50a,50c)에는 0이 로드된다. 그러므로, 좌측>우측이고 상측>하측이다. 이는, Y=0가 스크린의 상측에 위치하고 x=0가 스크린의 좌측에 있다고 가정하면, "부(negative)" 픽셀 경계 박스를 표현한다.

어드레스 매니저(48)는 2개의 모드, 구체적으로는 수동 모드(passive mode)와 능동모드(active

mode)중 하나로 구성될 수 있다.

수동 모드에서, 어드레스 매니저(48)는 픽셀 어드레스 버스(38c)로부터 픽셀 어드레스를 판독하여 그 어드레스들을 버퍼 메모리(46)에 전파한다. 어드레스 매니저(46)는, 픽셀 버스(38)상에서 전이된 모든 픽셀에 대해서든(TRACKAddr 명령), 또는 관련 버퍼 메모리(46)에서 업데이트된 픽셀만에 대해서든(TRACK-Upd 명령) 픽셀 어드레스 버스(38c)(TRACK 명령)로부터 판독되는 픽셀들을 포괄하는 스크린 박스(screen box enclosing pixels)를 동적으로 구성한다. 그 스크린 박스 좌표는 최소 및 최대 X 및 Y 픽셀 좌표(좌측, 상측, 우측, 하측)를 내장하는 레지스터(50a,50b,50c,50d)에 기억된다. 레지스터(50a-50d)는, 스크린이 업데이트되어 있는지, 그리고 어떤 스크린 영역이 영향을 받았는지를 결정하기 위하여 호스트 버스(30)를 거쳐 호스트 프로세서(12)에 의해 판독 가능하다. 초기화 목적으로(for initialization purposes), 레지스터(50a-50d)는 0영역 스크린 박스로 리세트된다(RESET 명령). 어드레스 매니저(48)는 또한, 이들 레지스터의 내용에 영향을 주지 않고서, 스크린 박스를 규정하는 레지스터를 업데이트하는 것을 중단시키도록 프로그램될 수도 있다(HOLD 명령).

능동 모드에서, 어드레스 매니저(48)가 픽셀 어드레스를 발생시키는 장방형 스크린 영역을 규정하기 위하여 스크린 박스를 구성하는데 사용되는 것과 같은 내부 레지스터(50a-50d)가 이용된다. 스크린 박스 좌표를 규정하는 레지스터(50a-50d)의 내용은 프로세서(12)에 의해 외부적으로
프로그램되거나, 미리 실행된 TRACK 명령의 결과로서 프로그램된다. 픽셀 어드레스가 발생되는 순서는 발생될 제1픽셀 어드레스를 지정함으로써 프로그램된다. 발생된 어드레스는, 어드레스 매니저(48)의 출력을 픽셀 어드레스 버스(38c)에 접속시키는 버스 인터페이스(52)를 통해서 픽셀 어드레스 버스(38c)에 기억된다. 수동 모드에서처럼, 픽셀 어드레스도 또한 버퍼 메모리(46)로 통과된다.

버퍼 메모리(46)로 통과되는 픽셀 어드레스는, 픽셀 어드레스 버스(38c)로부터 받은 것이든 또는 내부적으로 발생된 것이든지간에, 그것에 오프셋 값(offset value)를 더함으로써 수정될 수 있다. 이러한 기능을 성취하기 위해서, 어드레스 매니저(48)는 X 오프셋 레지스터(54a)와 Y 오프셋 레지스터(54b)를 포함한다. 어드레스 매니저(48)는, (a) 불가(never), (b) 판독 동작 전용(only for read operations), (c) 기록 동작 전용(only for write operations), (d) 판독 및 기록 동작 겸용(both read and write operations)인 오프셋을 더하도록 프로그램할 수 있다. 이 능력은, 예컨대, 픽셀들을 복제하기 위하여 다른 스크린 영역들을 연관짓도록 허용할 수 있다.

제9a도를 참고로 보다 구체적으로 설명하면, 어드레스 매니저(48)는 픽셀 메모리(46)에 액세스하기 위한 어드레스를 제공하는 어드레스 발생기(102)를 포함한다. 어드레스 발생기(102)는, 내부 레지스터를 사용하여 어드레스를 발생시키거나 혹은 픽셀 버스(38)로부터 어드레스를 판독함으로써 동작한다. 그후에, 이 어드레스가 오프셋 유닛(104)으로 보내져 그 유닛에서 프로그램된 오프셋에 따라 수정될 수 있다. 오프셋 유닛(104)의 어드레스 출력은, 필요한 D램 타이밍 신호(RAS,CAS 등)와 함께 필수적인 D램 행열 어드레스(DRAM row and columm addresses)를 버퍼 메모리에 제공하는 D램 곤트롤러(106)에 제공된다.

제9b도는 어드레스 발생기(102)의 구조를 보다 상세히 도시하고 있다. 어드레스 발생기(102)는 우측 레지스터(50a), 좌측 레지스터(50b) 하측 레지스터(50c) 및 상측 레지스터(50d)를 포함한다. 이들 레지스터들은 멀티플렉서 그룹(108)을 경유하여 호스트 프로세서 데이타 버스(30a)로부터 직접 로드되거나, 혹은, 오프셋 유닛(104)내에 기억된 어드레스(AWRITE)로부터 로드될 수 있다. 이들 레지스터들은 또한 모두 0 또는 모두 1로 세트될 수도 있다. 어드레스 발생기(102)는 (≥)기능을 수행하는 제 1 비교기(110)와 (≤)기능을 수행하는 제 2 비교기(112)를 포함한다.

비교기(110)의 A 입력은 좌측 레지스터(50b) 또는 상측 레지스터(50b)의 출력을 선택하는 멀티플렉서(114)로부터 제공된다. 비교기(110)의 B 입력은 P 어드레스 버스(38c)상에 나타나는 픽셀 어드레스 또는 XY 카운터(118)에 의해 발생된 픽셀 어드레스(CNT XY)를 선택하는 멀티플렉서(116)로부터 제공된다. 카운터(118)에는 멀티플렉서(120)를 경유하여 멀티플렉서(114)의 출력(LOW) 또는 멀티플렉서(122)의 출력(HIGH)이 미리 세트될 수 있다. 그와같이 하여, 카운터(118)는 규정된 스크린 박스의 4개의 코너(corner)중 어느 하나에서 주사(scanning)를 시작하도록 프로그램할 수 있다. 카운터(118)는 또한 후술되는 상태 레지스터(STATUS register ; 152)에 판독 전용(read-only) 상태 신호로서 제공되는 BUSY 신호를 발생시킨다.

비교기(112)의 A 입력은 멀티플렉서(116)에 의해 제공된다. 비교기(112)의 B 입력은 우측 레지스터(50a)와 하측 레지스터(50c)의 출력 사이에서 선택하는 멀티플렉서(122)를 경유하여 제공된다. 2개의 비교기의 출력은 논리 블럭(logic block ; 124)에 제공되며, 그 논리 블럭은 픽셀 어드레스의 기억된 스크린 박스 좌표와의 비교 결과 픽셀 어드레스가 규정된 스크린 박스의 “내측(inside)”에 있음을 표시하는 경우, 신호(INSIDE)를 발생시킨다. INSIDE신호는, 카운터가 규정된 스크린 박스의 “내측”에 있도록 규정된 스크린 영역에 대해서만 픽셀 어드레스를 발생시키도록 XY 카운터(118)의 인에이블 제어 입력(enabling control input)으로 이용된다.

어드레스 발생기(102)가 한 파이프라인 사이클 내에, 즉 하나의 파이프라인 위상을 걸러서만(only every other pipeline phase) 타스크(task)를 완료할 필요가 있다는 점에서, X 및 Y좌표의 계산은 2개의 파이프라인 위상에 걸쳐 분산된다. 제1위상중에, X좌표가 업데이트되며, 제2위상중에는 Y좌표가 업데이트된다. 픽셀 어드레스 버스(38c)로부터의 X 및 Y 어드레스, 혹은, XY 카운터(118)에 의해 발생된 어드레스는 X 멀티플렉서(126) 및 Y 멀티플렉서(128)를 경유하여 오프셋 유닛(104)으로 출력된다.

앞서 설명한 TRACK 명령과 관련하여, 비교기(110,112)들이 각기 2개의 연속적인 단계에서 표현(PADDR. ?>=? ? ?) 및 표현(P_ADDR. ?<? ? ?)을 결정한다. 이들 시험결과는, FLAGi(i=좌측, 우측, 상측, 하측)으로 도시된 4개의 플립-플롭(110a-112a)에 기억되어 후에 좌측, 우측, 상측, 하측 레지스터(50a-50d)용 업데이트 신호를 한정하는데(to qualify) 사용된다.

예 : FLAGleft=(P_ADDR. x<LEFT)

이들 시험이 단계 GA 및 MA중에 수행되지만, 스크린 박스 레지스터(50a-50d)는 WR 단계중에 단지 업 데이트되기만 한다는 점에서, 픽셀 어드레스가 임시로 세이브(save)되어야 한다. 이는, 후술하겠지만, 레지스터 AWRITE(132)를 사용하여 오프셋 유닛(104)에 의하여 행해진다.

같은 이유로, 즉, 다음 세트의 플래그(FLAGi)가 제1세트의 플래그 사용전에 발생되기 때문에, FLAGi AUX로서 도시된 제2세트의 플립-플롭(110b,112b)이 있는바, 그 플립 플롭은 단계 TC 중에 FLAGi플립-플롭(110a,112a)로부터 값을 받는다.

WR 단계중에, 레지스터 AWRITE(132)의 내용이 좌측, 우측, 상측, 하측 레지스터(50a-50d)의 조건에 따라 복제된다.

이 데이타 전이 조건은 TRACKAddr 명령에 대해서, FLAGi_AUX and PHASE [2] and STEP이고, TRACKUpd 명령에 대해서는, FLAGi_AUX and MUpdate and PHASE [2] STEP and이다.

TRACKAddr 명령은 현재의 스크린 박스의 외측에 있는 PAddress 버스(38c)상의 모든 픽셀에 대한 업 데이트를 수행한다. TRACKUpd 명령의 경우, 이는 버퍼 메모리(46)내의 해당하는 픽셀이 변경된 경우, 즉, MUPdate=1인 경우에만 실행된다.

다음 표 4는 TRACK 명령 동작을 요약한 것이다.

[표 4]

위 상	단 계	작 용
1	GA	X좌표를 시험하라
2	MA	FLAGleft 및 FLAGright를 기록하라.
3	RD	Y좌표를 시험하라
4	TC	FLAGtop 및 FLAGbottom을 기록하라.
5	DM	FLAGi를 FLAGi_AUX로 복제하라.

6	WR	TRACKAddr:
		if(FLAGtop_AUX) then
		TOP=AWRITE.y
		TRACKUpd:
		if(FLAGtop_Aux and MUpdate)
		TOP=AWRITE.y
		dto. for left, right, and bottom.

제9c도는 오프셋 유닛(104)의 구조를 보다 상세히 예시하고 있다. 오프셋 유닛은 조건에 따라 XY좌표에 오프셋 값을 더함으로써 어드레스 발생기(102)로부터 받은 어드레스들을 변환한다. 이 오프셋은 버퍼 메모리(46)에 대한 기록 액세스, 판독 액세스 또는 두 형태의 액세스를 위하여 더해질 수 있다.

전술한 파이프라이닝(pipelining) 기술에 따라서, 연속된 픽셀들의 동작이 중첩(overlap)하므로, 선행 픽셀이 픽셀 메모리(46)에 되돌아가서(back to the pixel memory(46)) 기록되기전에 픽셀 값이 판독된다. 결과적으로, 픽셀 어드레스는 버퍼될(to bo buffered) 필요가 있다. 이러한 버퍼 작용(buffering)은 AREAD 레지스터(130) 및 AWRITE 레지스터(132)에 의해 성취된다. 이들 레지스터는 각기 다음의 판독 액세스 또는 다음의 기록 액세스를 위한(수정되지 않은) 픽셀 메모리(46) 어드레스들을 기억한다. AREAD 레지스터(130)는 어드레스 발생기(102)의 멀티플렉서(126,128)의 출력을 받는다. AREAD 레지스터(130)에 기억된 어드레스는, 실제의 판독 어드레스(actual read address)를 형성하는데 사용된 후에 AWRITE 레지스터(132)로 전이되어 차후의 픽셀을 처리하는 중에 그곳에 보관된다. 멀티플렉서(134)는 레지스터들(130, 132) 사이에서 선택하여 X 가산기(X-ADDER ; 136) 및 Y 가산기(138)에 출력을 제공한다. X 가산기(136)의 제 2 입력은 XOffset 레지스터(54a)로부터 제공된다. Y 가산기 (138)의 제2입력(138)은 YOffset 레지스터 (54b)로부터 제공된다. 레지스터(54a,54b)는 호스트 데이타 버스(30a)로부터 받은 X 및 Y 오프셋 값들을 각기 미리 기억하고 있다. 수정된 어드레스(AMOD)(140)를 기억하는 출력 레지스터는 D 램 콘트롤러(106)의 인터페이스를 형성한다. 논리부(142)는 Y 어드레스가 버퍼 메모리(46)의 최종 액세스로부터 변경되어 있는지를 검출하기 위하여 제공된다. 변경 상태가 검출되는 경우, 다음의 픽셀 메모리(46) 액세스가 버퍼 메모리(46)의 다른 페이지로 향하게 될 것임이 표시된다. PAGE 신호의 어서숀(the assertion of the PAGE signal)으로 D 램 콘트롤러(106)는 버퍼 메모리(46)에 대한 페이지 모드 액세스를 발하는 것

(issuing)을 정지하게 된다. 또한 버퍼 메모리(46)에 대한 표준 액세스가 페이지 모드 액세스보다 더 많은 시간을 필요로 하므로 PAGE 신호의 어서숀에 의해 픽셀 버스(32)상에 WAIT 신호가 발생한다. 표 5는 오프셋 유닛(104)의 동작을 요약한 것으로서 이들 동작을 픽셀 동작의 단계 및 페이스와 연관시킨 것이다.

[표 5]

위 상	단 계	작 용
1	GA	어드레스 발생기가 AREAD를 기록한다.
	MA	오프셋 유닛이 AREAD의 내용을 수정하고 AMOD를 기록한다.
2	RD	D 램 콘트롤러가 메모리에 대한 판독 액세스를 실행한다. AWRITE : =AREAD
	TC	
3	DM	오프셋 유닛이 AWRITE의 내용을 수정하고 AMOD를 기록한다.
	WR	D 램 콘트롤러가 메모리에 대한 기록 액세스를 실행한다.

픽셀 버스 인터페이스(52)는 M 버퍼(22) 시스템의 일부로서 버퍼 모듈을 구성한다. 그것은 무슨 정보가 픽셀 버스(32)로부터 표면 버퍼 모듈(32)를 전파되는지, 그리고, 무슨 정보가 표면 버퍼 모듈(32)로부터 픽셀 버스(38)로 통과되는지를 제어한다.

표면 버퍼 모듈(32)은 또한 그 전체 동작을 제어하는 제어 블럭(56)을 포함한다.

제5c도에는 제어 블럭(56)이 보다 상세히 도시되어 있다. 제어 블럭(56)은 3개의 주요 구성요소, 즉, 호스트 프로세서(12) 인터페이스(150), 전술된 상태 레지스터(152) 및, 전술된 파이프 레지스터(154)를 포함한다. 호스트 인터페이스(150)는 호스트 버스(30)의 제어 라인(HCTr1 ; 30)을 해석하고, 관련 표면 버퍼 모듈(32)의 판독 및 기록 동작을 검증한다. 그러한 동작이 이루어진 경우, 호스트 인터페이스(150)는 HRead 또는 HWrite를 세트하여 호스트 판독 사이클 또는 기록 사이클을 표시한다. 호스트 프로세서(12)가 상태 레지스터(152)를 어드레스하면, 신호 A15가 어서트된다. 신호 A15는 호스트 어드레스 버스(HAddr ; 30b)가 값 15(0×0…of)를 갖는 경우 어서트된다.

상태 레지스터(152)는 2개의 부분으로 분할된다. 제1부분은 호스트 프로세서(12)에 의해 기록 및 판독될 수 있다. 이 부분은 표면 버퍼 모듈(32)용의 구성 데이타(configuration data)를 기록한다. 상태 레지스터(152)의 제 2 부분은 판독 전용 지점으로서 표면 버퍼 모듈(32)의 현재의 작동 상태를 표시한다. 본 발명의 본 실시예에서, 판독 전용 부분은 호스트 프로세서(12)에 모듈의 BUSY 상태 및 업데이트 상태를 제공한다. 상태 레지스터 신호들의 기능을 표 6에 기재한다.

[표 6]

명 칭	기 능
READY(BUSY)	0 : 스크린 영역이 주사되고 있다(being scanned out).
	1 : 버퍼가 스크린 영역에 주사하고 있지 않다.
CHANGED(Update)	버퍼는 이 플래그가 리세트됐던 최종시간 이래 변경되어 왔다.
WVIDEO	이 비트는, 버퍼 메모리가 픽셀 정보를 디스플레이 장치(24)로 주사하는데 사용된 비디오 램(VRAN)으로 이루어진 버퍼 모듈에 이용된다.
	1인 경우 : 픽셀 정보가 주사되야 한다.
	0인 경우 : 아무런 주사 동작도 발생하지 않는다.
ENABLE	0 : 메모리에 기록하지 않음. 픽셀 버스로부터 단절(disconnect)하라.
	1 : 정규 동작이 인에이블된다(enabled).
WADDR	어드레스 버스 PAddress(38c)를 구동하라.
WDATA 2	데이타 버스 PData 2(38b)를 구동하라.
WDATA 1	데이타 버스 PData 1(38a)을 구동하라.
PDATA 2	소오스 데이타 버스 PData 2를 선택하라.
PDATA 1	소오스 데이타 버스 PData 1을 선택하라.
OFFSETMODE(OMode)	오프셋 레지스터를 사용하는 현재의 방법.
DIRMODE(DMode)	스크린 영역에 주사하는 현재의 방향.
ADDRMODE(AMode)	어드레스 발생기를 위한 현재의 동작 모드
FUNCTIONENA	어서트된 경우, 계산 데이타 경로에 의해 수행되는 동작이 OpCode(기능부(34b)) 출력에 의해 결정된다. 세트되지 않은 경우, 계산 데이타 경로의 동작은 연산 레지스터(94)내의 OpCode에 의하여 결정된다. 이 비트는 제어 버퍼(34)를 위해서만 채용된다.

앞서 검토했던 바와 같이, 파이프 레지스터(154)는 3비트 시프트 레지스터로서 구성된다. 각각의 비트는 파이프라인 스테이지들중 하나에 해당한다. 비트 상태에서 논리 1은 해당 스테이지가 활성 상태, 즉, 데이타로 "채워져(filled)" 있음을 표시한다. 역으로, 0은 그 스테이지가 비활성 상태임을 표시한다. 신규한 픽셀 데이타가 픽셀 버스(38)상에 위치될 때마다, 파이프 레지스터에는 NEW 신호에 의해 STEP 신호와 함께 1이 입력된다. STEP 신호의 차후의 엣지(subsequent edges)는 이 논리 1을 파이프 레지스터(154)의 잔여 스테이지를 통해서 전파한다. 파이프 레지스터(154)의 출력은, 논리(154a)에 의해, 들어오는 WAIT 신호 및 ENABLE 상태 비트와 함께 마스크된다.

파이프 레지스터(154)가 순환 시프트 레지스터(circular shift register)로 구성되지 않고, 매번(every time) 신규한 픽셀이 발생하며, NEW가 어서트되므로, 논리 1이 파이프 레지스터(154)의 최하위 비트(least significant bit)로 시프트된다. 동시에, 최하위 비트에 있는 정보는 파이프 레지스터(154)에서 다음의 비트 위치로, 즉, 위상[1]로 시프트 된다. 마찬가지로, 정보는 위상[1]로부터 위상[2]로 전파된다. 그러므로, 논리 1들이 3개까지 동시에 파이프 레지스터(154)에 있을 수 있다.

[실시예]

PIPE 레지스터(154) 내용	의 미
000	활성 픽셀 없음
001	단계 GA 및 MA 활성
011	단계 GA, MA, RD 및 TC 활성
111	모든 단계 활성, 파이프 라인 가득참(pipeline full)
101	단계 RD 및 TC 유휴(idle)

[제어 버퍼]

제6도에 도시된 바와 같이, 제어 버퍼 모듈(34)의 구조는 표면 버퍼 모듈(32)과 닮았다. 현존하는 차이점은 제어 버퍼 모듈(34)의 다른 기능성을 반영한다. 표면 버퍼 모듈(32)의 구성 요소와 공통인 요소에는 프라임부호를 표시한 도면 부호로서 도시했다.

구체적으로 설명하면, 제어 버퍼 모듈(34)에 기억된 픽셀 관련 정보는 단 하나, 또 기껏해야 몇개의 비트폭을 갖는다, 효율을 유지하기 위하여, 이들 데이타 아이템(item)중 몇개가 제어 버퍼 모듈(34)의 버퍼 메모리(46')에 기억된다. 그러므로, 시험 데이타 경로(42') 및 계산 데이타 경로(44')는 메모리 워드의 서브셋(subset) 또는 현재 바람직한 실시예의 경우 32비트인 픽셀 그룹을 조작한다.

또한, 버퍼 메모리(46')의 내용은 외부 소오스로부터 로드되기 보다는 내부적으로 구성된다. 예컨대, 픽셀 플래그는 PResult 버스(38d)상에 보고된 시험 결과에 따라 세트 또는 리세트된다. 그러므로, 제어 버퍼 모듈(34)은 픽셀 데이타 버스(38a, 38b)로의 접속부를 필요로하지도 않고 가지고 있지도 않다.

종종, 다음의 픽셀 값을 계산하는 동작은 픽셀 시험 결과에 종속한다. 예를 들면, 픽셀 카운터는 시험이 성공인 경우에는 증분되고 시험이 실패인 경우에는 감소된다(decremented). 그러므로, 계산 데이타 경로(44')에 의해 수행되는 동작은 시험 결과에 따라 동적으로(dynamically) 선택가능하다. 이는 결정 테이블(40')로부터 출력된 부가적인 기능부(additional function ; 34b)에 의해 성취된다. 그 기능부(34b)의 출력은 계산 데이타 경로(44')에 의해 수행되는 기능을 선택하는바, 기능부(34b) 출력은 계산 경로(44')가 응답하는 OpCode를 이송한다.

제어 모듈(34)에 대하여, 보다 구체적으로 제5d도 및 제5f도를 참고로 설명하면, 결정 테이블(40')은 멀티플렉서(64a) 및 출력 레지스터 (70a)를 포함한다. PResult 버스(38d)는 멀티플렉서(64a)의 출력 비트를 선택하는데, 그 비트들은 레지스터 (70a)에 의해 래치된다(latched). 그 레지스터(70a)의 출력은 기능 버스(34b)를 구동한다. 전술된 바와 같이, FUNCTIONENA로서 지시된 상태 레지스터(152)의 비트에 의해 호스트 프로세서 로드형 연산 레지스터(Host processer-loaded operation register ; 94)의 출력과 결정 테이블(40')에 의해 발생된 OpCode 사이의 선택이 이루어진다. 그러므로, 호스트 프로세서(12)는, 레지스터(94)의 출력 또는 결정 테이블(40')내의 레지스터(70a)의 출력이 될 계산 데이타 경로(44')를 위한 동작(OpCode)을 선택하도록 인에이블된다.

[가상 버퍼]

그래픽 서브 시스템(10)의 타구성 요소들은 가상 버퍼(36)에 의해 M 버퍼(22)와 인터페이스된다. M 버퍼(22)는 픽셀 버스(38)로부터 입력 데이타를 받고 출력 데이타를 그 버스상에 위치 시킨다. 가상 버퍼(36)는 픽셀 버스(38)와 인터페이스되어 픽셀버스(38)에 대한 피지컬 버퍼(physical buffer)로서 기능한다. 이는 여러가지 다른 유닛 및 장치를 M 버퍼(22)에 연결시키는 표준화된 수단을 제공한다. 이 기술은 또한 픽셀을 버퍼 구성을 집중시키는 하나의 지정된 경로 및 픽셀 연산을 위한 다른 버스(들)을 제공하는 종래의 버퍼 구성의 제한을 배제한다. 본 발명은 다른 픽셀 처리 스테이지 사이에서의 픽셀 정보의 전이를 허용하는 중앙의 다방면 버스(central and universal bus)(픽셀 버스 ; 38)를 제공함으로써 이들 한계를 극복한다.

제10도에 도시된 바와 같이, 가상 버퍼(36)는 피지컬 버퍼(32,34)에 의해 제공된 기능성의 일부를 제공한다. 예를 들면, 주사 변환 유닛(scon conversion unit)을 구성하는 가상 버퍼(36)는 픽셀 버스(38)상에 픽셀 어드레스 및 픽셀 데이타를 제공하는 판독 전용 버퍼(로컬 메모리(local memory) ; 36)만을 구비한다. 가상 버퍼(36)에 의해 실행될 수 있는 다른 예시적 기능은 다음과 같다.

가상 버퍼(36)는 프레임 버퍼로서 기능하여 표준 버퍼들의 기능성 모두를 제공할 수 있다. 이 경우, 픽셀 데이타를 출력 장치(24)로 전이시키는 제2인터페이스(I/O 인터페이스(36b))가 제공된다.

가상버퍼(36)는 비디오 버퍼로서도 기능할 수 있다. 이 경우, I/O 인터페이스(36b)는 가령 비디오 카메라와 같은 영상 소오스로부터 픽셀 정보를 받게 되어 있다. 이렇게 받은 영상 데이타는 내부 버퍼 메모리(36a)에 국부적으로(locally) 기억되어, 그후, 픽셀 버스(38)를 통해서 M 버퍼(22)내로 전파된다. 이 실시 예의 경우, 로컬 (36a)는 선입 선출(First In/First Out(FIFO))버퍼로서 충족될 수 있다.

가상 버퍼(36)는 또한, 예컨대, 입력된 이벤트(event)가 사용자가 발하게 되는(issued by a user) 픽셀 좌표용의 픽셀 어드레스를 발생시키는 입력 장치들을 인터페이스하는 기능을 할 수도 있다. 이들 픽셀 좌표는 I/O 인터페이스(36b)를 통해서 받아들여져서 픽셀 어드레스 버스(38c)를 통해서 M 버퍼(22)에 도입된다.

특정 실시예의 경우, 픽셀 버스(38)에 대한 모든 접속부가 필요한 것이 아님을 주지해야 한다. 예를 들면, 가상 버퍼(36)는 PData 버스(38a, 38b)중 단 하나의 버스로만 데이타를 제공하거나, 혹은, 그 단하나의 버스로부터만 데이타를 받는다. 또한 몇몇 어플리케이션의 경우, 가상 버퍼(36)가 자체적으로 내장되어 있을 수도 있고 I/O 인터페이스(36b)를 필요로 하지 않을 수도 있다. 예컨대, 가상 버퍼(36)는 간접 픽셀(U, V) 어드레스를 받아서 해당하는 표면 텍스쳐 정보를 출력하는 서브루틴 텍스쳐 발생기(procedure texture generator)로서의 기능을 수행한다.

더우기, 가상 버퍼(36)는 PResult 버스(38d)를 사용할 필요가 없는 바, 즉, 그 동작이 시험 결과와는 독립되어 있다. 그러나, PResult 버스(38d)는 가상 버퍼(36)를 위해서 유리할 수 있다. 예를들면 가상 버퍼는, 래스터 라이저(rasterizer)로서 기능하는 경우, PResult 버스(38d)의 라인을 제어함으로써 전면 또는 후면에 속하는 픽셀들을 표시(mark)할 수 있다. 이 기술은 후술되는 트리콜 알고리듬(Trickle algorithm)을 실행하는데 유용하다.

요약하면, 본발명의 일 태양은 버퍼 메모리 (46)의 내용에 대한 복수의 병렬 픽셀 시험을 시행하고, 이들 시험 결과를 조합하여 복합 업데이트 조건(complex update condition)을 형성할 수 있는 능력이다. 각각의 버퍼 모듈의 시험 유닛(42)은 호스트 프로세서(12)에 의해 개별적으로 프로그램되어 여러가지 다른 시험을 수행할 수 있다. 이 시험은 버퍼 메모리로부터 판독된 픽셀값, 픽셀 버스(3

8)로부터 받은 픽셀 값(들) 및, 시험 데이타 경로내에 있는 프로그램가능한 레지스터에 기억된 값의 조합을 고려할 수 있다. 주어진 시험 결과(진 또는 위)는 다른 모든 버퍼 모듈(32,34,36)로 전파된다. 모든 시험 결과의 조합은 특징 버퍼 모듈이 신규한 픽셀값을 관련된 내부 버퍼 메모리(46)에 기록할 것인지를 결정한다. 각각의 버퍼 모듈은 또한, 특정 시험에 독립하여, 부울리안 함수(boolean function)를 결정 테이블(40)내에 기억하여 그것의 내부 버퍼 메모리(46)가 업데이트되는 시험 결과의 조합을 규정하는 능력을 가진다.

복잡한 업데이트 조건(complex update conditions)에 의해 제공되는 제어 버퍼 모듈(34)의 융통성(flexibility)은 계산 데이타 경로(44')에 의해 실행될 기능을 동적으로 선택할 수 있는 능력에 의해 더욱 향상된다. 복합 업데이트 조건과 마찬가지로 그기능은 모든 버퍼 모듈에 의해 보고된 시험 결과를 기초로하여 선택된다. 시험 결과들의 여러가지 다른 조합에 대하여 실행될 일련의 기능들은 역시 호스트 프로세서(12)에 의해 업데이트 조건과는 독립해서 프로그램될 수 있다.

본 발명의 M 버퍼(22)는 복수의 다른 실시예로서 기능을 수행할 수 있다.

제1실시예에서, 모든 버퍼들은 하드웨어의 피지컬 버퍼로서 기능을 수행한다. 픽셀 버스(38)는 버퍼 모듈에 접속해서 백 플레인 버스 시스템(backplane bus system)으로서 기능을 수행할 수 있다.

제2실시예에서, 모든 버퍼 모듈들은 단일 프로세서, 다중 타스킹 컴퓨터(single-processor, multi-tasking computer)의 스프트 웨어로서 에뮬레이트 된다(emulated). 버퍼 메모리(46)는 가상 메모리로 할당되고, 버퍼 프로세서는 호스트프로세서(12)에 의해 실행된 여러가지 타스크 또는 프로세서에 의해 에뮬레이트된다. 여러가지 다른 프로세서, 예컨대, 픽셀 데이타 버스(38a, 38b), 픽셀 어드레스 버스(38c) 및, PResult 버스(38d) 사이의 통신은 잘 확립된 기술에 의해 실현된다. 예를들면, 픽셀 전이는 공유 메모리(shared memory)를 이용하여 실행되고, 프로세서 또는 버퍼 모듈간의 통신의 잘려진 인터-타스크 통신 기술(inter-task communication techniques)에 의해 성취된다.

제3실시예에서, M 버퍼(22)중 특정 버퍼는 피지컬 버퍼로서 실현되며, 나머지는 가상 메모리에 할당된 소프트웨어 버퍼를 이용하여 에뮬레이트되고, 병렬 프로세스들(parallel processes)에 의해 제어된다. 피지컬 버퍼와 소프트웨어 버퍼간의 접속은 가상 버퍼(36)를 이용하여 성취된다. 가상 버퍼(36)는 호스트 프로세서(12)를 제공하고, 다시 버퍼 모듈을 에뮬레이팅하는 프로세서가 하드웨어로 이루어지는 M 버퍼(22)의 그 부분에 액세스한다.

제4실시예에서, M 버퍼(22)는 다중 프로세서(multi-processor), 범용 병렬 시스템(general purpose parallel system)을 통해서 에뮬레이트 된다. 각각의 프로세서는 하나의 버퍼 모듈의 기능성을 에뮬레이트한다.

M 버퍼(22)의 구성에 대하여 설명했는 바, 이제 그 기능성을 보다 더 이해할 수 있도록 하기 위하여 M 버퍼(22)를 사용하는 실시예를 설명하기로 한다.

[실시예1]

제1실시예는 복수의 버퍼로부터 이익을 얻는 렌더링 방법(rendering method)을 실행키 위하여 M 버퍼(22)를 사용한다. 이 실시예는 트리클 알고리듬으로 알려진 기술과의 관계에서 제공된다. 트리클 알고리듬은 1989년 12월 뉴욕, 요크타운 헤이츠 소재, 티, 제이. 왓슨 리서어치 센터, 아이비엠 리서어치 디비죤의 기술 보고서 RC 15172 “Z-Buffer Rendering From CGS : The Trickle Algorithm” 에서 데이비드 에이. 엡스테인, 프레데릭 더블유. 쟈션 및 자로스로우 알. 로시그낙이, 그리고, 1990년 9월 EUROGRAPHICS 1990의 그래픽 하드웨어에 대한 유로그래픽 워크숍 회보 “Correct Shading of Regularized CSG Solids Using a Depth-Interval Buffer”에서 쟈렉 로시그낙 및 제프리 우가 설명한 것이다.

1991년 3월 12일자로 데이비드 에이. 엡스테인, 자로슬로우 알. 로시그낙 및 제프리 더블유. 우 명의로 “심도 버퍼의 사용에 의한 CSG 표현의 직접적인 디스플레이”라는 명칭으로서 출원되어 본원과 함께 양도된 미국 특허출원 제07/672, 058호도 참고자료가 된다.

일반적으로, 트리클 알고리듬은 래스터 디스플레이상에 콘스트럭티브 솔리드 지오메트리 (Constructive Solid Geometry(CSG)) 대상물을 그것의 경계 표현(boundary representation)으로 변경시키지 않고 디스플레이 할 수 있게 한다.

글로벌 알고리듬(global algorithm)은 다음과 같이 진행된다.

1. CSG 표현은 대상물의 프리미티브(primitive of objects)의 교선으로 구성되는 소 대상물(sub-objects)의 결합으로서 대상물을 표현하는 가법 표준형(sum-of-products form)으로 변환된다.

2. 각각의 프로덕트(product)는 프로덕트 버퍼(product buffer(P 버퍼))내로 주사 변환된다. (scan-converted). 프로덕트의 영상이 완전히 P-버퍼내로 렌더된(recdered)후에, P 버퍼는 프로덕트의 가시적인 정면을 포함한다. 이 영상은 Z 버퍼를 이용하여 유니온 버퍼(union-buffer(U 버퍼))내로 흡수된다.

다음의 의사 코드는 이들 두 단계를 표현한다.

대상물을 가법 표현형으로 변환하라

버퍼를 초기화하라

각각의 프로덕트에 대하여 :

전면 픽셀들을 P 버퍼로 계산하라

그 결과를 심도 버퍼를 사용하여 U 버퍼내로 병합하라

[참조]

transform the object into a sum-of-products form

initialize the U-Buffer

for each product do :

compute the front pixels into the P-Buffer

merge the result into the U-Buffer using a

depth-buffer

프로덕트의 가시적인 전면 픽셀들의 계산은 보다 복잡한 과정이다. 그 주 아이디어는 프로덕트의 가시적인 표면점이 다른 모든 프리미티브 내부에 있는 가장 근접한 정면 방향 표면의 정면상에 있도록 하는 것이다. 그 표면을 찾는데는 2개의 심도 버퍼가 필요하다.

제1버퍼는 프로덕트의 영상을 최후에 홀드하는(holding) P 버퍼이다. 알고리듬이 진행중인 동안, 이제까지 찾은 가장 근접한 정면 방향 점들(closest front-facing points)이 P 버퍼에 기억된다. 제 2 버퍼인 조사 버퍼(search buffer(S 버퍼))는 P 버퍼에 기억된 표면의 이면에 있는 정면 방향 표면을 조사하는데 사용된다. 그러한 표면은 찾으면, 조사 버퍼는 P 버퍼에 기억된 표면을 교체한다. 이는 P 버퍼로 하여금 장면(scene)을 통해서 전방으로부터 후방으로 진행하거나 또는 “트리클”하게 한다. 다음의 의사 코드는 이 기술을 상세히 표현한다.

P 버퍼를 배경으로 초기화하라

실행되지 않는 동안(while not done), 프로덕트의 프리미티브를 통해서 순환하고 다음을 실행하라 :

P 버퍼 뒤에 있는 다음면을 S 버퍼에 계산해 넣어라

다음의 프리미티브면(primitive face)이 정면을 향하고 있는 픽셀에서는 :

S 버퍼를 P 버퍼내에 복제하라

[참조]

initialize the P-Buffer to the background

while not done, circulate through the primitives fo the

product and do :

compute into the S-Buffer the next face that

lies behind the P-Buffer

at pixels where the next primitive face is front-

facing do :

copy the S-buffer into the P-Buffer

트리클 알고리듬의 이러한 표현은 호스트 프로세서(12)에 의해 실행을 위해 적절한 코드로 번역된다. 제11도의 플로우챠트는 어플리케이션의 전체 구조를 예시한다. 프로그램 세그먼트가 사용된 버퍼 및 그들의 구성을 우선적으로 규정한다는 것은 알려져 있다. 플래그 버퍼 및 P 버퍼는 다른 타스크를 위해서 이용된다. 그러므로, 그것들은 다른 구조로 이용된다. 그것들은, 이들 구조를 메인 루프(main loop) 내측에 규정하는 것을 회피하기 위하여, 사전에 규정되어 변수로 기억된다. 변수들의 내용은 그후에 버퍼로 효과적으로 전이된다.

단계 A는 사용된 버퍼 모듈들의 명칭을 규정한다. 6개의 표면 버퍼 모듈(32)이 이용되는 바, 2개는 U 버퍼(하나는 색상용이고 하나는 심도용), 2개는 P 버퍼(하나는 색상용이고 하나는 심도용) 및 2개의 S 버퍼(하나는 색상용이고 하나는 심도용)이다. 색 및 심도 정보를 픽셀 버스(38)로 입력하기 위하여 2개의 가상 버퍼(36)가 사용된다(주사 C, 주사 Z). 주사 플래그(Scan Flag)는 제1의 가상 평면(V 평면)으로 규정된다. 단계 A는 또한 상수 및 국부적 변수(constants and local variables)를 규정한다.

단계 B는 모든 버퍼 구조를 규정한다. 구체적으로, U 버퍼는 공차 값을 사용하는 Z 버퍼로서 규정된다. 공차 값은 관련된 버퍼 모듈(32)의 오퍼랜드 레지스터(90)내에 기억된다. P 버퍼는 2가지 목적을 위해서 사용된다. 그것은 S 버퍼와 함께 심도 간격 버퍼(Depth Interval Buffer ; DIB)를 형성하는데 사용되고, 그후에도 심도 버퍼로서 사용된다. 이들 구성은 변수로서 기억되는바, 이에 대해서는 이미 설명했다. S 버퍼는 DIB 내에서 타겟 버퍼(target buffer)로서 그리고 백 버퍼(back buffer)로서 이용된다. 제어 모듈(34)내에 기억된 플래그 비트는, DIB의 일부인 P 버퍼에 대하여 심도시험이 통과하고 P 버퍼를 기록하는 DIB의 동작중에 시험되는 경우, 변경된다.

단계 C에서, 대상물은 가법 표현형으로 변환되고, 단계 D에서 U 버퍼가 초기화된다.

단계 E에서, 메인루프는 모든 프로덕트를 P 버퍼에 렌더하고 그후에 그 프로덕트를 U 버퍼내로 병합한다. 메인 루프는 P 버퍼(E1)를 초기화하는 단계 및, 비록 프로덕트 프리미트비트들이 순환 리스트(circular list)에 기억됐더라도 프로덕트에 속하는 각각의 프리미티브를 처리하는 루프의 완료를 기다리는 단계를 포함한다. 이 내측 루프는 S 버퍼(E2)를 초기화하는 단계(E2) 및 P 버퍼 및 S 버퍼

에 기록된다. 매번, 픽셀이 S 버퍼에 기록되고 픽셀 플래그는 역전된다. S 버퍼는 TRACKUpd 명령에 의해 변경되는 픽셀의 궤도를 유지하도록 구성된다(E4). 다음에, 프리미티브가 순환 리스트로부터 호출되고(fetched), 그것의 모든 면들은 DIP 내로 주사 변환된다(E5).

이 프리미티브의 모든 면들이 주사 변환된 후에 S 버퍼는 현재의 프로덕트의 가시적인 정면상에 있을 후보자(candidate) 표면 점들을 포함한다. 이들점은 픽셀 플래그가 어서트되고 S 버퍼 및 P 버퍼에 기억된 Z 값이 다른 경우에 P 버퍼에 기록된다. 그러므로, P 버퍼는 버퍼 메모리(46)내에 기억되는 Z 값과는 다른 들어오는 Z 값을 시험하도록 구성된다(E 6.1). 플래그 버퍼는 픽셀 플래그의 값을 시험하도록 구성된다(E 6.2). P 버퍼의 스크린 박스는 리세트되고 P 버퍼는 그것의 버퍼 메모리(46)내에서 업데이트되는 모든 픽셀들 둘레의 경계 박스를 구성하도록 되어 있다(E 6.3). 마지막으로, S 버퍼는 픽셀들을 그것의 버퍼 메모리(46)로부터 P 버퍼의 버퍼 메모리(46)로 전이시키기 시작한다(E 6.4).

새로이 발생된 픽셀들을 편성하기 위하여(to track) 준비하는 단계들(E4 및 E 6.3)은 픽셀 둘레에 경계 박스를 유지해야 하는 그들 표면 버퍼 모듈(32)을 위하여 어드레스 매니저(48)에 기록하는 단계를 포함한다. 이는 우선 TRACK 명령을 보내고 그후에 TRACK 명령을 보냄으로써 성취된다. 단계 E6.4는 S 버퍼내에서 변경된 픽셀들을 픽셀 버스(38)를 거쳐 P 버퍼로 전이시키는 것을 포함한다. 이는 호스트 프로세서(12)로부터 GENERATE 명령을 발함으로써 픽셀 버스(38)에 픽셀을 기록하도록 S 버퍼의 어드레스 매니저(48)내의 어드레스 발생기(102)를 프로그래밍하는 것을 포함한다. 이는 어드레스 발생기(102)의 카운터(118)로 하여금 변경된 픽셀들을 에워싸는 스크린 박스용의 픽셀 어드레스를 생성하도록 한다.

단계 E7에서, P 버퍼가 업데이트되었는지를 알아보기 위하여 시험이 이루어진다. YES라면, 메인 루프가 완료되고 단계 F로 실행이 진행되어, 그 단계에서 P 버퍼내에 기억된 프로덕트가 U 버퍼내로 병합되어 유니온을 구성한다. NO인 경우, 단계 2에서 계속 실행된다.

[실시예 2]

제2실시예는 1990년 8월 컴퓨터 그래픽스, 24(14) : 309-318의 “The Accumulation Buffer : Hardware Support for High-Quality Rendering”에서 폴 해벌리와 커트 아켈리가 설명한 누산 버퍼(accumulation buffer)와 관련된다. 누산 버퍼는 앤티 얼라이러징, 모션 블러(motion blur), 또는 소프트 섀도우(soft shadow) 같은 렌더링 기술(rendering techniques)을 실행하는데 사용될 수 있다. 저변에 깔린 원리는 약간 다른 렌더링 파라메타 또는 뷰포인트(rendering technique)와 함께 영상을 몇번 렌더링하는 것이다. 그 영상들은 누산 버퍼에 누산된다. 모든 영상들이 렌더링된 후에, 누산 버퍼에 기억된 값들은 최종의 영상 색을 얻기 위하여 렌더링 패스(rendering pass)의 횟수로 나뉘어진다.

다음의 의사코드는 알고리듬의 구조를 표현한다.

```
축적 버퍼를 0으로 초기화하라
렌더링 파라메타를 리세트하라
렌더링될 모든 이미지에 대해서
{렌더링 파라메타를 디스터브하라(disturb)
   Z버퍼를 비워라
   이미지를 렌더링하고 매 픽셀(x,y)마다 실행하라 :
   if(new Z[x, y]<old z[x, y])
   {old Z[x, y]=new Z[x, y] ;
   accBuffer[x, y) +=newColor(x, y] ;
   }
}
모든 픽셀(x, y)에 대해서
accBuffer[x, y]/=number_of_images_rendered ;
```

[참조]

```
initialize the accumulation buffer to zero
reset rendering parameters
for all images to bo rendered
{disturb rendering parameters
clear z-buffer
render the image and for every pixel(x, y) do :
   if(newZ[x, y]<oldZ[x, y])
   {oldZ[x, y]=newZ[x, y] ;
```

```
        accBuffer(x, y]+=newColor[x, y] ;
      }
   }
   for all pixels(x, y)
     accBuffer[x, y]/=number_or_images_rendered ;
```

이 기술이 M 버퍼(22) 구조상에서 실행되는 경우 몇가지 고려사항들(consideration)이 어드레스되야 한다.

우선, 누산 버퍼는 중간 결과를 홀드하기에 충분한 비트를 제공해야 한다. 그러므로, 각각의 색 성분에 대해서 16비트가 제공된다.

둘째로, 각각의 표면 버퍼(32)가 픽셀당 32비트를 기억하기 때문에, 누산 버퍼는 2개의 표면 버퍼(32)를 가로질러 분포된다(distribute). 제1의 표면 버퍼는 적 및 청 성분(Red and Blue component)을, 그리고 제2표면 버퍼는 녹 성분(Green component)을 기억한다.

세째로, 픽셀 버스(38)가 한번에 2개의 데이타 아이템을 전이시키기 때문에, 픽셀을 표현하는 3개의 단어(Z, RB, G)가 2단계로 전이된다. 제1단계에서, Z값은 버스 PData 1(38a)에 위치되고, 적-청 정보는 버스 PData 2(38b)에 위치된다. 제2단계에서, Z값은 버스 PData 1(38a)로 어서트되고 녹 성분은 버스 PData 2(38b)상에 어서트된다. 픽셀 데이타를 2단계로 보내는 이 방법은 2개의 픽셀 데이타 버스들을 이용하는 바람직한 실시예의 결과이다. 3개의 픽셀 데이타 버스가 채용되는 경우, 모든 픽셀 데이타가 동시에 전이되게 된다.

네째로, 이들 2개의 위상을 구별하기 위하여, 제어 버퍼(34)내의 가상 비트 평면(virtual bit plane)이 이용된다. 주사 변환 프로세스는 이 비트 평면을 제 1 단계중에 세트하고, 제 2 단계동안 리세트한다.

두 단계동안 심도 비교(depth comparison)가 수행된다. 그러나, Z값은 제 2 단계중에만 관련된 버퍼 메모리(46)에 기록된다.

마지막으로, RGB 값을 홀드하는 버퍼를 확보하기 위하여 2개의 표면 버퍼들이 합체된다. 이는 RB 표면 버퍼에 특정 비트 평면만을 마스킹하고 기록함으로써 성취된다.

전술의 설명을 기초로하면, 여기에서 설명된 M 버퍼(22)가 많은 관점에서 다중 명령 단일 데이타(Multiple Instruction Single Data(MISD)) 스트림 구성과 유사함을 알 수 있다. M 버퍼(22)는 부가적인 버퍼 모듈을 갖춤으로써 보다 높은 레벨의 복잡도(complexity) 및 처리력(Processing power)에 대하여 쉽게 용량을 맞출 수 있다(scalable). 이와 관련하여, 기본 그래픽 버퍼는 단지 몇개의 버퍼 모듈로 구성될 수 있으며, 그래픽 버퍼는 후에 보다 많은 버퍼 모듈을 추가함으로써 향상될 수 있다.

제12도는 또 다른 예시적 그래픽 버퍼 구성을 보여주고 있다. 가상 버퍼(36)는 주사 변환 유닛(20)으로부터 입력을 받아서 픽셀 버스(38)로 픽셀 데이타 및 픽셀 어드레스를 공급한다. 픽셀 버스(38)에는 3개의 표면 버퍼 모듈(32a,32b,32c)이 접속된다. 표면 버퍼 모듈(32a,32b)은 각기, 비 비월주사 시스템(noninterlaced system)을 위하여, 하나의 완전한 픽셀 데이타 프레임을 기억한다. 이 2개의 표면 버퍼 모듈은 프레임 버퍼 모듈로서 기능하고 픽셀 색 데이타에 대하여 이중 버퍼링(double buffering)을 제공한다. 표면 버퍼 모듈(32c)은 심도 버퍼(Z 버퍼)로서 기능하고, 각 스크린 픽셀에 대해서, 심도치를 기억한다.

표면 버퍼 모듈(32a,32b)은 디스플레이(24)에 직접 인터페이스하는 픽셀 출력 회로(33)를 포함하도록 각기 수정된다. 동작에 있어서, 디스플레이(24)는 표면 버퍼 모듈(32a,32b)중 하나로부터 픽셀을 판독하는 반면에 다른 것에는 다음 프레임을 위한 픽셀이 로드된다. 그러므로, 디스플레이(24)는 2개의 버퍼 모듈(32a,32b) 사이에서 교번한다. 이 실시예의 경우, 표면 버퍼 모듈(32a,32b)은 또한 디스플레이(24)로 인터페이싱하는 가상 버퍼 모듈로서 기능한다. 제어 버퍼모듈(34)은 이러한 실행을 위해서는 필요하지 않다.

(57) 청구의 범위

청구항 1

복수의 픽셀을 디스플레이하는 디스플레이 수단과 함께 사용되며(for use with display means that displays a plurality of pixels) 복수의 픽셀을 표현하는 정보를 기억하는 그래픽 버퍼(the graphic buffer storing information describing the plurality of pixels)에 있어서, 버스에 의해 함께 접속되는 복수의 모듈과 복수의 모듈 각각을 호스트 프로세서 수단(hast processor means)에 연결시키는 수단을 구비하며, 상기 복수의 모듈중 적어도 하나의 제1의 모듈은 복수의 픽셀들 각각에 대하여 대상물 영상의 표면 특성(a surface characteristic of an image of an object)을 지정하는(specifying) 정보를 기억하는 제1메모리를 구비하며, 상기 모듈들중 적어도 하나의 제1모듈의 상기 메모리는 상기 버스에 접속되어 기억된 정보를 상기 버스에 공급하며, 상기 복수의 모듈중 적어도 하나의 제2모듈은 각각의 픽셀에 대하여 다른 정보를 기억하는 제2메모리를 구비하며, 상기 복수의 모듈중 상기 적어도 하나의 제1모듈과 상기 적어도 하나의 제2모듈은 관련된 제1 및 제2메모리에 연결되어 그 관련된 제1 및 제2메모리내의 정보를 수정하는 처리수단(processing means)을 각기 구비하며, 상기 호스트 프로세서 수단은 상기 관련된 제1 및 제2메모리내에 기억된 정보에 따라 제1 및 제2의 상기 처리수단에 의해 수행될 동작을 지정하며, 상기 복수의 모듈중 적어도 하나의 제3모듈은 상기 버스에 접속됨과 아울러 필셀 정보 소오스(a source of pixel information)에 접속되어

상기 모듈중 적어도 하나의 모듈이 받을 픽셀 정보를 상기 버스에 제공하며, 상기 복수의 모듈들 중 적어도 하나의 제4모듈은 상기 버스에 접속됨과 아울러 픽셀 정보의 수요자(consumer)에게 연결되어 픽셀 정보 수요지가 받을 픽셀 정보를 상기 버스로부터 공급하는 것을 특징으로 하는 그래픽 버퍼.

청구항 2

제1항에 있어서, 상기 버스는, 픽셀 어드레스를 이송하는(for conveying pixel address) 복수의 제1신호 라인과, 픽셀 데이타를 이송하는 복수의 제2신호 라인과, 상기 버스의 동작을 부분적으로(in part) 표현하는 다른 정보를 이송하는 복수의 제3신호 라인을 구비하는 것을 특징으로 하는 그래픽 버퍼.

청구항 3

제2항에 있어서, 상기 복수의 제1신호 라인은 상기 제1모듈, 상기 제2모듈, 상기 제3모듈 및, 상기 제4모듈 각각에 공통으로 접속되어, 상기 디스플레이 수단과 관련된 좌표 시스템 기준으로 하는(being referenced to a coordinate system) 픽셀의 어드레스를 가지는 상기 모듈중의 하나에 의하여 구동되는 것을 특징으로 하는 그래픽 버퍼.

청구항 4

제2항에 있어서, 복수의 상기 제2신호 라인들이 복수의 데이타 버스를 구비하는 것을 특징으로 하는 그래픽 버퍼.

청구항 5

제4항에 있어서, 상기 복수의 데이타 버스중 적어도 하나는 상기 제1모듈, 상기 제3모듈 및, 상기 제4모듈에 공통으로 접속되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 6

제2항에 있어서, 상기 복수의 제3신호 라인은 복수의 신호 라인으로 이루어진 픽셀 결과 버스(Pixel Result Bus)를 구비하며, 상기 복수의 신호 라인 각각은 상기 모듈들중 한 모듈에 접속되어 그 모듈이 수행한 픽셀 시험(pixel test) 결과에 따라 상기 모듈에 의해 구동되며, 상기 복수의 신호 라인 각각은 타 모듈에 의해 받아들여져(being received) 그 관련 메모리내에 기억된 정보가 수정될 것인지를 결정하는 상기 타 모듈의 결정 수단(decision means)에 접속되며, 상기 결정은 상기 모든 모듈로부터 받은 픽셀 시험 결과에 따라 이루어지는 것을 특징으로 하는 그래픽 버퍼.

청구항 7

제2항에 있어서, 상기 복수의 제3신호 라인이 제어 버스(control bus)를 구비하고, 버스 사이클을 규정하는 주기(a period defining bus cycle)를 가지는 동기 클럭 신호(a synchronizing clock signal)를 이송하는 신호 라인과, 어서트시(when asserted), 버스상에 신규한 픽셀 정보가 있음을 나타내는 신호 라인과, 어서트시, 상기 모듈들중 적어도 하나가 동작을 완료하기 위하여 하나 이상의 부가적인 버스 사이클을 필요로 함을 나타내는 신호 라인과, 각각의 픽셀에 대한 픽셀 정보를 처리하는 동안 상기 모든 모듈을 복수의 동작 단계로 순서화하는 반복 신호(a repetifive signal for sequencing all of said modules through a plurality of operational steps)를 이송하는 신호라인을 구비한 것을 특징으로 하는 그래픽 버퍼.

청구항 8

제2항에 있어서, 상기 복수의 제3신호 라인이 제어 버스를 구비하며, 어서트시, 동적 메모리 장치(dynamic memory device)로 이루어지는 메모리를 가지는 모듈상에 메모리 리프레쉬 동작이 일어나야 함(memory refresh operation)(a memory refresh operation is to occur)을 나타내는 신호 라인을 구비하는 것을 특징으로 하는 그래픽 버퍼.

청구항 9

복수의 픽셀을 표현하는 정보를 기억하는 그래픽 버퍼에 있어서, 버스에 의해 함께 접속된(coupled together gy a bus) 복수의 모듈을 구비하고, 상기 모듈중 적어도 2개는 디스플레이 수단의 각각의 픽셀에 대하여 픽셀 특성(a characteristic of the pixel)을 나타내는 정보를 기억하는 메모리 수단을 구비하고, 상기 적어도 2개의 모듈은 관련된 메모리 수단에 접속되어 그 메모리 수단에 기억된 정보를 수정하는 처리 수단(processing means)을 각기 구비하며, 상기 버스는, 픽셀 어드레스를 이송하는 복수의 제1신호 라인과, 픽셀 데이타를 이송하는 복수의 제2신호 라인과, 픽셀에 대해 수행된 시험 결과를 표현하는 정보를 이송하는 복수의 제3신호 라인을 구비하고 있는 것을 특징으로 하는 복수의 픽셀을 표현하는 정보를 기억하는 그래픽 버퍼.

청구항 10

제9항에 있어서, 상기 적어도 2개의 모듈은 픽셀 처리 동작을 성취하기 위하여 복수의 연속 처리 상태(a plurality of consecutive processing states)를 발생시키는 수단을 각기 구비하며, 그 처리 상태는, 인에이블된 경우(if enabled), 픽셀 어드레스가 복수의 제1신호 라인으로 구동되는 어드레스 발생(a Generate Address(GA)) 상태와, 복수의 제1신호 라인으로부터 픽셀 어드레스가 판독되고, 필요하다면, 픽셀 처리 동작을 성취하기 위하여 그 픽셀 어드레스가 수정되는 어드레스 수정(a Modify Address(MA)) 상태와, 관련 메모리 수단으로부터 데이타를 판독하기 위하여, 인에이블된 경우, 그 데이타를 복수의 제2신호 라인으로 구동하기 위하여 픽셀 어드레스 또는 수정된 픽셀 어드레스를 사용하는 판독(RD) 상태와, 적어도 하나의 모듈이 복수의 제2신호 라인으로부터 데이타를 판독하고, 필요하다면, 관련 메모리 수단으로부터 판독된 데이타에 따라 동작을 수행하며, 각각의 모듈

은 또한 관련 메모리 수단으로부터 판독된 데이타에 따라 비교 동작(comparison opertion)을 수행하며, 비교 동작의 결과는 각각의 모듈에 의해 복수의 제3신호 라인들중 관련된 신호 라인으로 구동되는 시험 및 계산(a Test & Compute(TC)) 상태와, 각각의 모듈이 복수의 제3신호 라인을 판독하고, 시험 및 계산 상태중에 타 모듈에 의해 그것으로 구동된 비교 결과를 기초로, 시험 및 계산 상태중에 수행된 동작의 결과가 관련 메모리 수단에 기록되야 하는지 여부를 결정하는 결정 및 수정 상태(a Decide & Modify(DM)) 상태와, 결정 및 수정 상태에 의해 인에이블된 경우, 시험 및 계산 상태중에 계산된 값이 픽셀 어드레스 또는 수정된 픽셀 어드레스로서 관련 메모리 수단내에 기록되는 기록(WR) 상태를 포함하는 것을 특징으로 하는 그래픽 버퍼.

청구항 11

제10항에 있어서, 상기 발생 수단은 간접 어드레싱(indirect addressing)을 수행하는 픽셀 처리 동작에 응답하여 RD 상태 후에는 적어도 하나의 부가적인 MA 상태를, 그리고 그 적어도 하나의 부가적인 MA 상태후에는 부가적인 RD 상태를 발생시키는 것을 특징으로 하는 그래픽 버퍼.

청구항 12

n×M 픽셀들을 디스플레이하는 디스플레이 수단과 함께 사용되며, 복수의 모듈을 구비하는 그래픽 버퍼에 있어서, 상기 모듈을 공통 버스를 게재하여 타 모듈에 접속시키는 수단과, 픽셀 정보를 기억하는 적어도 n×m 지점을 가지는 메모리 수단과, 상기 메모리 수단에 접속되어 그것으로부터 기억된 픽셀 정보를 판독해내고, 그 정보를 수정하며, 수정된 정보를 상기 메모리 수단으로 되돌려 보내는 프로세서 수단과, 상기 메모리 수단에 접속되어 그것으로부터 픽셀 정보를 판독해내며, 시험 결과를 표현하는 적어도 하나의 출력을 가지고 있고, 상기 적어도 하나의 출력은 타 모듈에 받아들여지도록 상기 공통 버스의 적어도 하나의 결과 신호 라인에 접속되어 있는 시험 수단(test means)을 구비하고 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 13

제12항에 있어서, 상기 모듈에는, 상기 공통 버스에 접속되어 그로부터 픽셀 어드레스를 받고, 상기 n×m 지점중 하나로부터 기억된 픽셀 정보를 판독해내도록 상기 메모리 수단을 어드레싱하기 위하여 상기 메모리 수단에 픽셀 어드레스를 제공하는 어드레스 매니저 수단(Address Manager means)이 구비되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 14

제13항에 있어서, 상기 어드레스 매니저 수단에는, 받은 픽셀 어드레스를 상기 메모리 수단에 제공하기전에 그 받은 픽셀 어드레스를 수정하는 수단이 구비되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 15

제13항에 있어서, 상기 어드레스 매니저 수단은, 상기 모듈중 선택된 모듈에 제공된 명령에 응답하여, 픽셀 어드레스를 발생시키고 발생된 픽셀 어드레스를 상기 메모리 수단에 제공하는 수단을 구비하며. 상기 어드레스 매니저 수단은 또한 상기 공통 버스에 접속되어 상기 모듈중 선택되지 않은 모듈에 의해 수정되도록 그 발생된 픽셀 어드레스를 공통 버스에 제공하는 것을 특징으로 하는 그래픽 버퍼.

청구항 16

제12항에 있어서, 상기 모듈은, 상기 수정된 픽셀 정보를 상기 메모리 수단에 되돌려 기억하는 것을 인에이블 또는 디스에이블하기 위하여 모든 모듈에 의해 발생된 결과 신호라인에 접속되어 그곳에 표현된 시험 결과에 응답하는 수단을 가지는 것을 특징으로 하는 그래픽 버퍼.

청구항 17

제13항에 있어서, 상기 모듈은 상기 공통 버스의 순서화 신호 라인(sequencing signal line)에 접속되어 복수의 순차적인 픽셀 처리 상태를 걸쳐(through a plurality of sequencial pixel processing states) 그 순서화 신호 라인에 의해 구동되는 입력을 가지는 순서화 수단(sequencing means)을 구비하고 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 18

제13항에 있어서, 상기 어드레스 매니저 수단이 디스플레이 수단의 2차원 픽셀 영역을 규정하는 좌표를 기억하는 레지스터 수단(register means for storing coordinates)을 구비하는 것을 특징으로 하는 그래픽 버퍼.

청구항 19

제16항에 있어서, 상기 인에이블 또는 디스에이블 수단(enable or diabling means)이 상기 결과 신호 라인에 접속된 어드레스 입력을 가지는 룩업 테이블 수단(lookup talbe means)을 구비하고 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 20

제12항에 있어서, 상기 공통 버스는 적어도 하나의 데이타 버스를 구비하며, 상기 프로세서 수단은 오퍼랜드 레지스터 수단(operand register means)을 구비하며, 상기 프로세서 수단은, 연산 레지스터 수단에 기억된 연산 코드에 응답하여, (a) 상기 메모리 수단의 출력 및 상기 적어도 하나의 데이타 버스로부터 받은 데이타, 또는 (b) 상기 메모리 수단의 출력 및 상기 오퍼랜드 레지스터 수단에

기억된 데이타, 또는 (c) 상기 오퍼랜드 레지스터 수단내에 기억된 데이타 및 상기 적어도 하나의 데이타 버스로부터 받은 데이타에 대하여 산술적 또는 논리적 연산(a arithmetic or a logical operation)을 수행하는 것을 특징으로 하는 그래픽 버퍼.

청구항 21

제20항에 있어서, 상기 오퍼랜드 레지스터 수단은 호스트 데이타 프로세서 수단에 접속되어 그것에 의해 오퍼랜드가 로드되는 것(being loaded)을 특징으로 하는 그래픽 버퍼.

청구항 22

제20항에 있어서, 상기 연산 레지스터 수단은 호스트 데이타 프로세서 수단에 접속되어 그것에 의하여 연산 코드가 로드되는 것을 특징으로 하는 그래픽 버퍼.

청구항 23

제16항에 있어서, 상기 공통 버스는 적어도 하나의 데이타 버스를 구비하고, 상기 프로세서 수단은 오퍼랜드 레지스터 수단을 구비하며, 상기 프로세서 수단은 연산 코드에 응답하여 상기 메모리 수단의출력 및 상기 오퍼랜드 레지스터 수단에 기억된 데이타에 대한 산술적 또는 논리적 연산을 수행하며, 상기 연산코드는 상기 인에이블 또는 디스에이블 수단의 출력으로부터 제공되는 것을 특징으로 하는 그래픽 버퍼.

청구항 24

제12항에 있어서, 상기 프로세서 수단이, 그것의 동작에 의하여 영향받지 말아야 하는(not to bo affected) 상기 메모리 수단의 출력의 하나 이상의 비트(one or more bits)를 나타내는 출력을 가지는 기록 마스크 레지스터 수단(writemask register means)을 구비하는 것을 특징으로 하는 그래픽 버퍼.

청구항 25

제12항에 있어서, 상기 공통 버스에 접속되어 그 버스에 픽셀 정보를 입력하는 버퍼 수단이 구비되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 26

제12항에 있어서, 상기 공통 버스에 접속되어 그 버스로부터 픽셀 정보를 출력하는 버퍼 수단이 구비되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 27

제12항에 있어서, 상기 공통 버스가 적어도 하나의 데이타 버스를 구비하고 있고, 상기 시험 수단은 기준 레지스터 수단을 구비하고 있으며, 상기 시험 수단은 시험 레지스터 수단내에 기억된 시험 사항(a test specifier)에 응답하여, (a) 상기 적어도 하나의 데이타 버스상에 나타나는 데이타에 대한 상기 메모리 수단의 출력, 또는 (b) 상기 메모리 수단의 출력 또는 상기 적어도 하나의 데이타 버스상에 나타나는 데이타에 대하여 기준 레지스터 수단의 출력을 비교하는 것을 특징으로 하는 그래픽 버퍼.

청구항 28

제27항에 있어서, 상기 시험 수단은, 상기 적어도 하나의 데이타 버스상에 나타나는 데이타에 대하여 상기 메모리 수단의 출력을 비교하기 앞서 상기 기준 레지스터 수단의 내용을 상기 메모리 수단의 출력에 가산하는 수단(adding means)을 구비하고 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 29

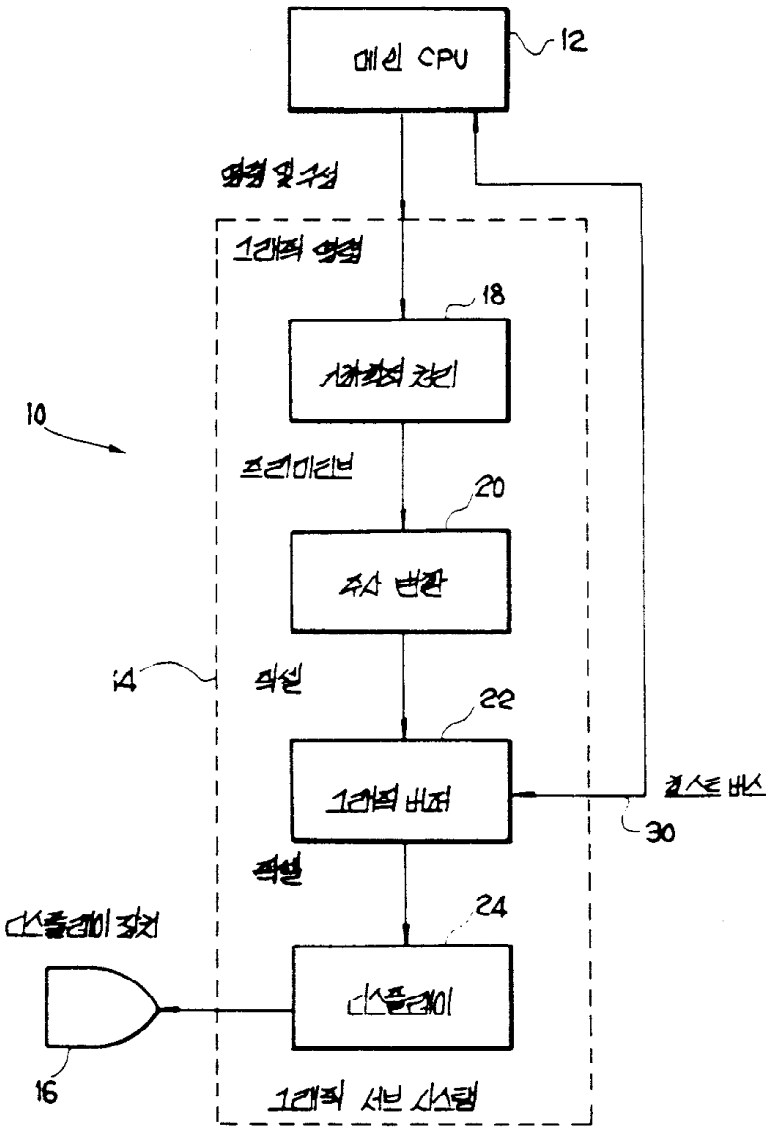
제27항에 있어서, 상기 시험 수단에는 서로 비교되야 할 비트를 나타내는 수단이 구비되어 있는 것을 특징으로 하는 그래픽 버퍼.

청구항 30

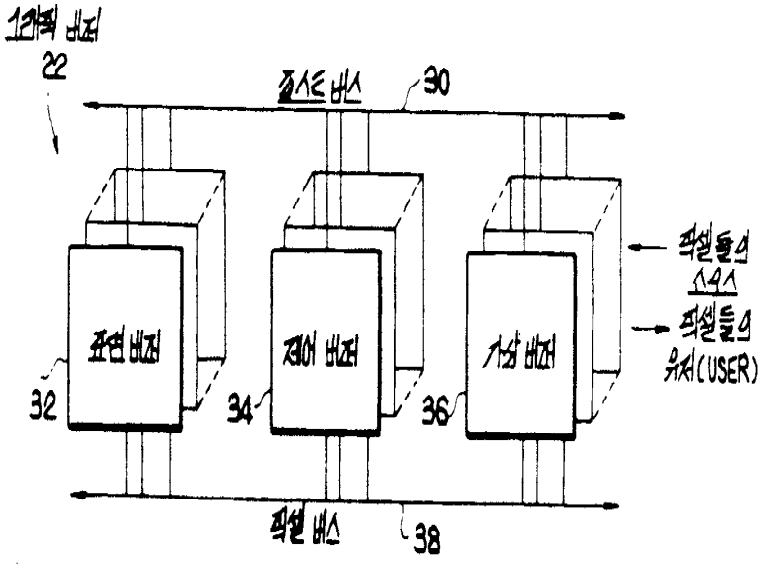
제12항에 있어서, 상기 모듈중 적어도 하나의 모듈의 기능성(functionality)이 상기 그래픽 버퍼의 외부에 있는 데이타 처리 수단의 동작에 의해 충족되는 것을 특징으로 하는 그래픽 버퍼.

도면

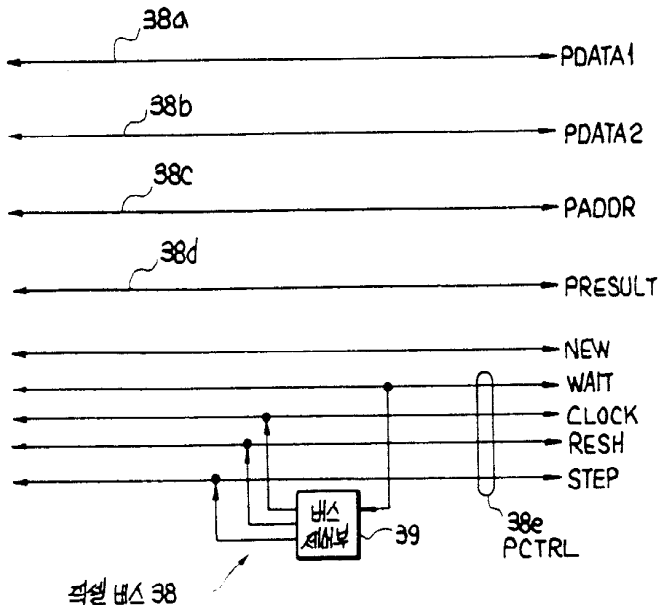
도면1



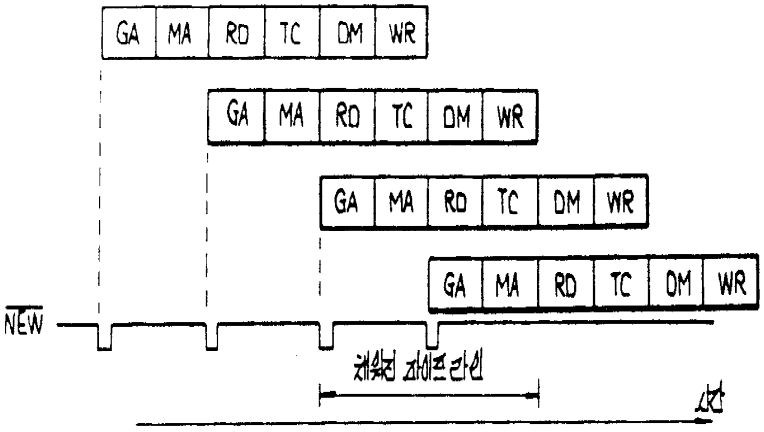
도면2



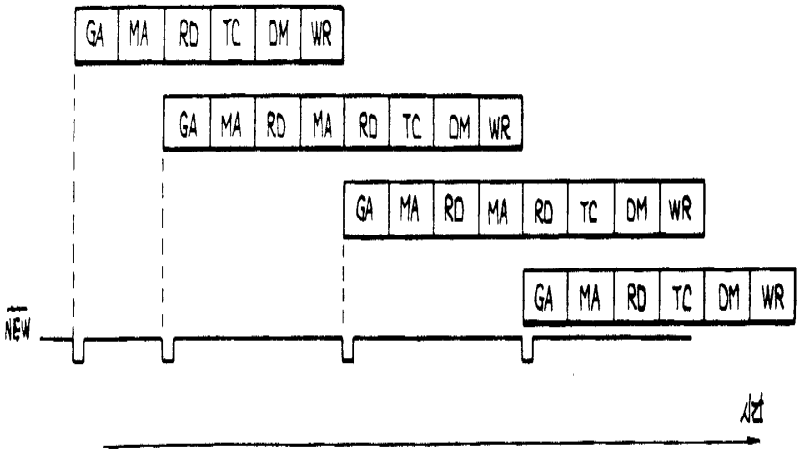
도면3



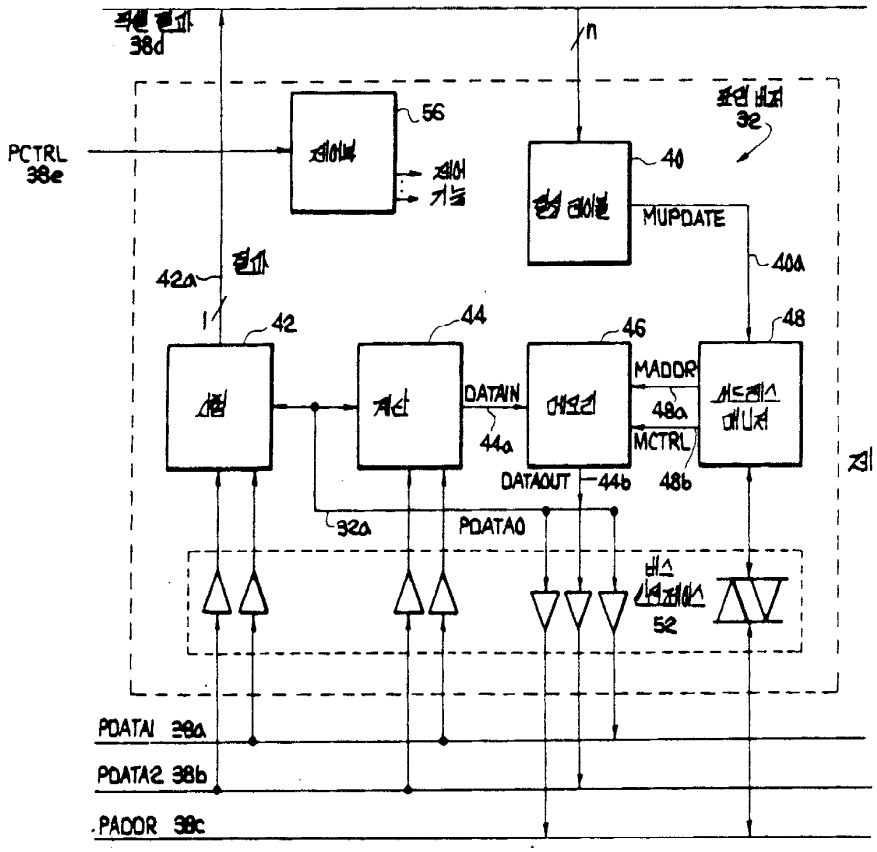
도면4A



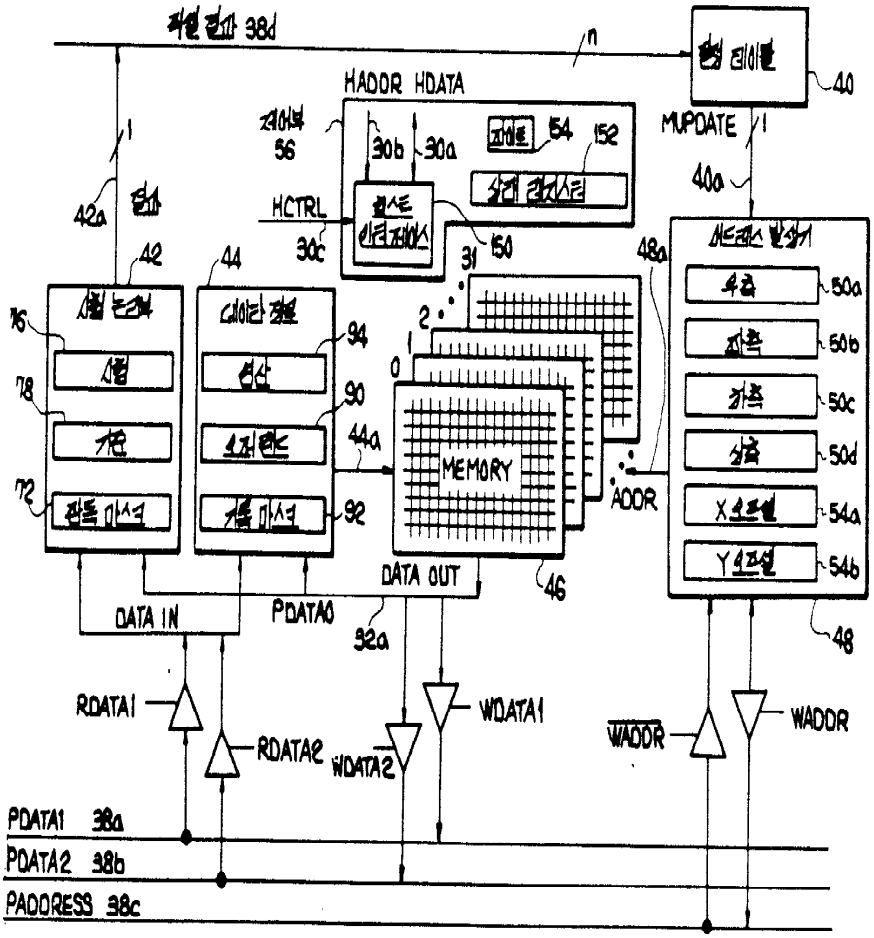
도면4B



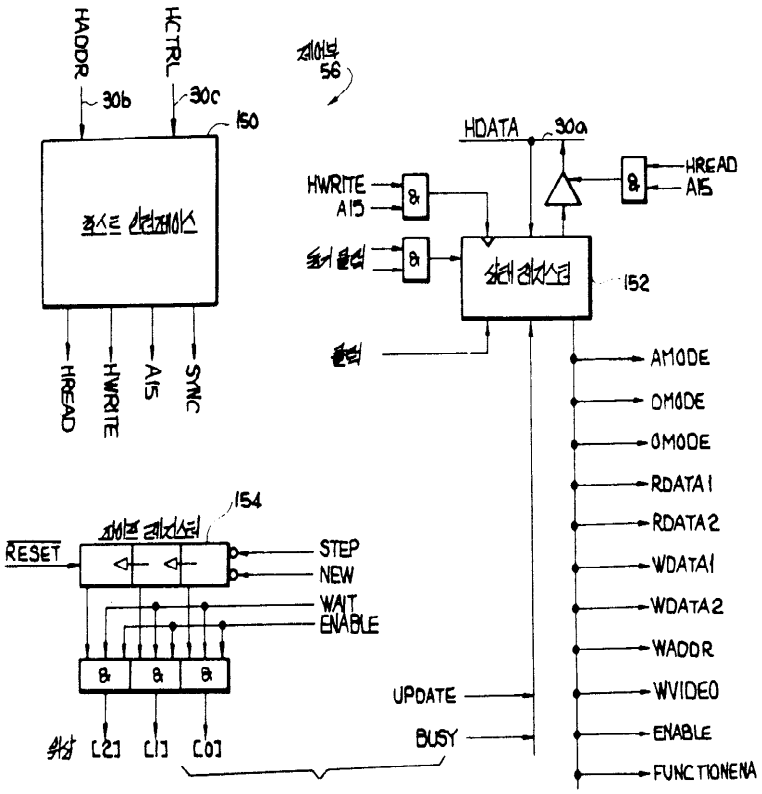
도면5A


38d
n
PCTRL
38e
56
40
32
MUPDATE
40a
42a
42
44
46
48
DATAIN
44a
MADDR
48a
MCTRL
48b
DATAOUT
44b
32a
PDATA0
52
PDATA1 38a
PDATA2 38b
PADDR 38c

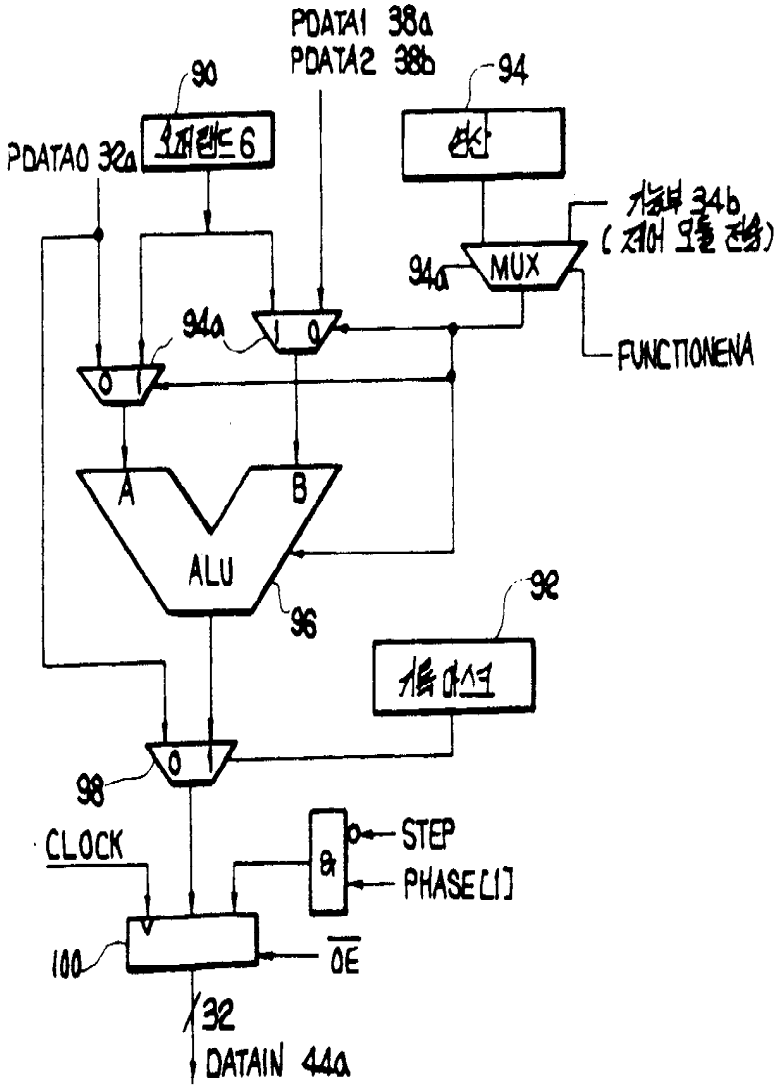
도면5B


HADDR HDATA
MUPDATE
HCTRL
MEMORY
ADDR
DATA OUT
DATA IN
PDATA0
RDATA1
RDATA2
WDATA1
WDATA2
WADDR
WADDR
PDATA1 38a
PDATA2 38b
PADDRESS 38c

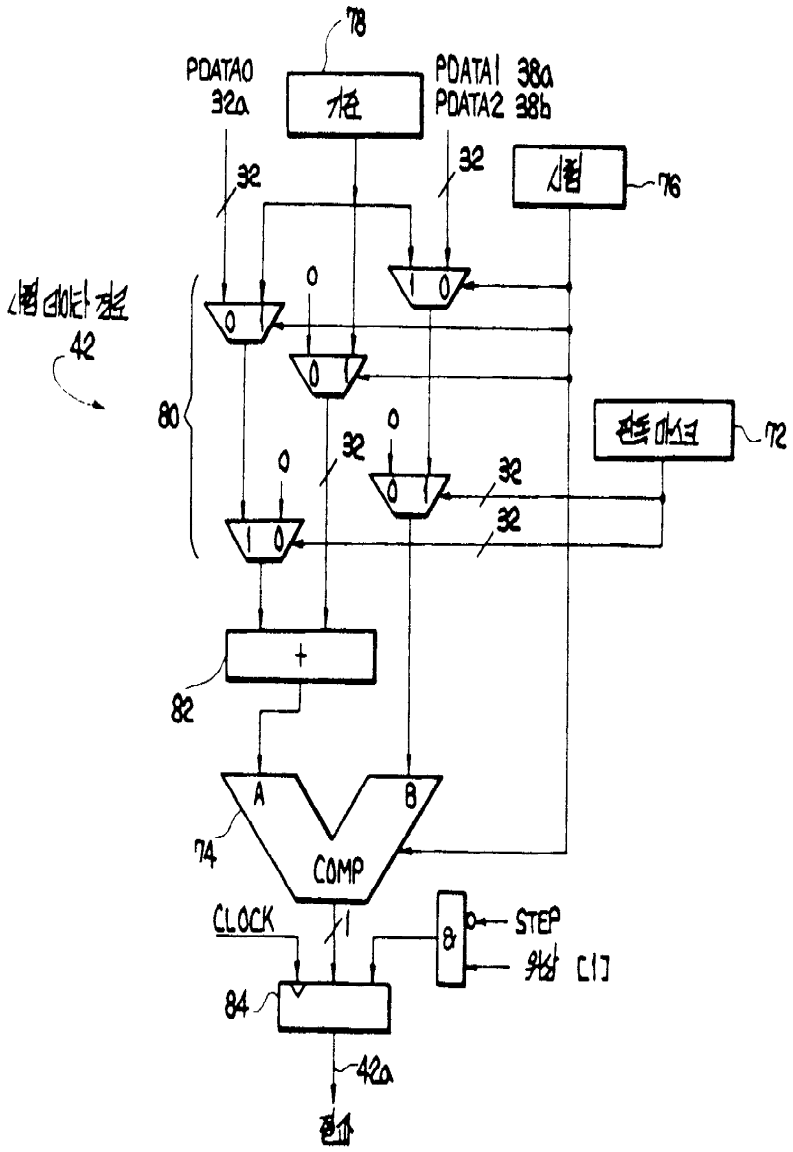
도면5C


HADDR
HCTRL
30b
30c
150
제어부
56
HDATA
30a
HWRITE
A15
HREAD
A15
152
HREAD
HWRITE
A15
SYNC
AMODE
DMODE
OMODE
RDATA1
RDATA2
WDATA1
WDATA2
WADDR
WVIDEO
ENABLE
FUNCTIONENA
154
RESET
STEP
NEW
WAIT
ENABLE
UPDATE
BUSY
[2]
[1]
[0]

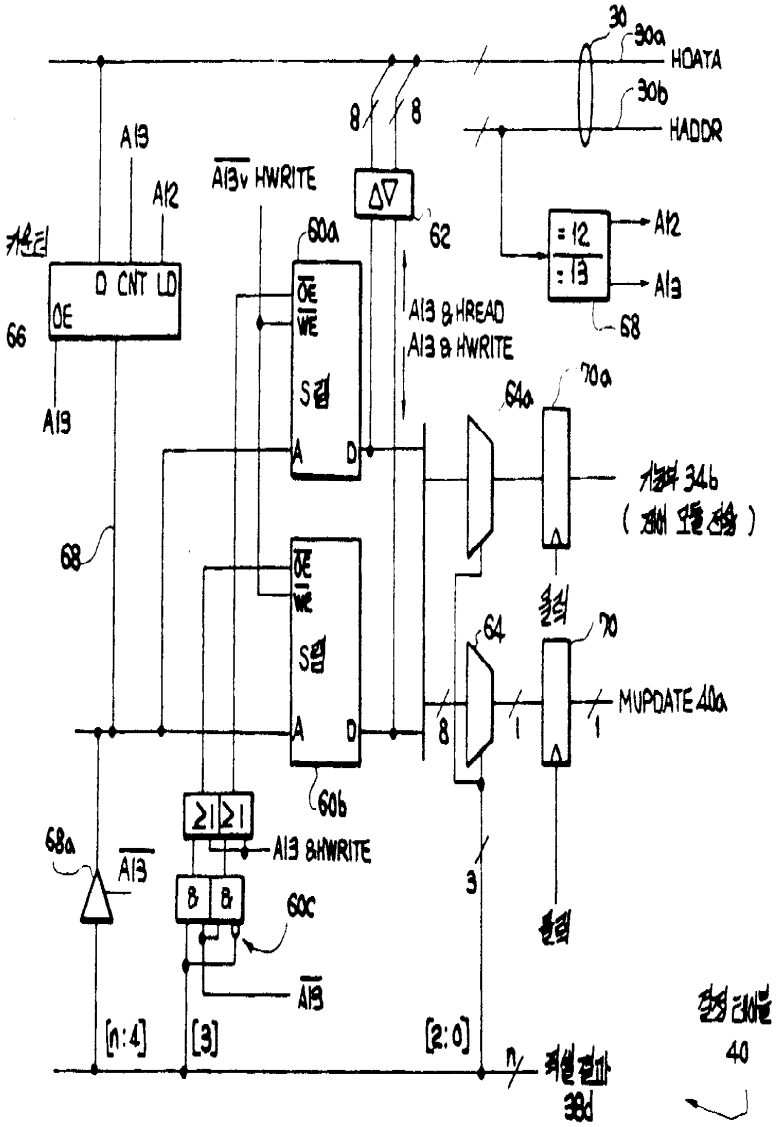
도면50


PDATA1 38a
PDATA2 38b
90
94
PDATA0 32a
MUX
94a
FUNCTIONENA
A
B
ALU
96
92
98
CLOCK
STEP
PHASE[1]
100
OE
32
DATAIN 44a

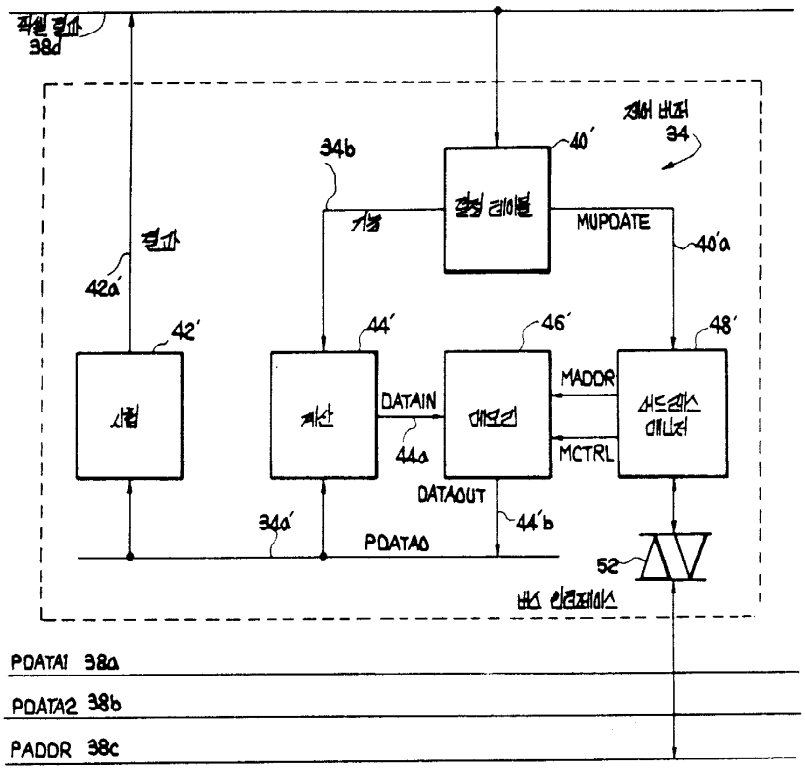
도면5E


78
PDATA0
32a
가산
PDATA1 38a
PDATA2 38b
32
32
시험
76
시험 데이타 경로
42
80
0
1
32
32
32
32
타겟 레지스터
72
+
82
A
B
74
COMP
CLOCK
1
8
STEP
84
42a

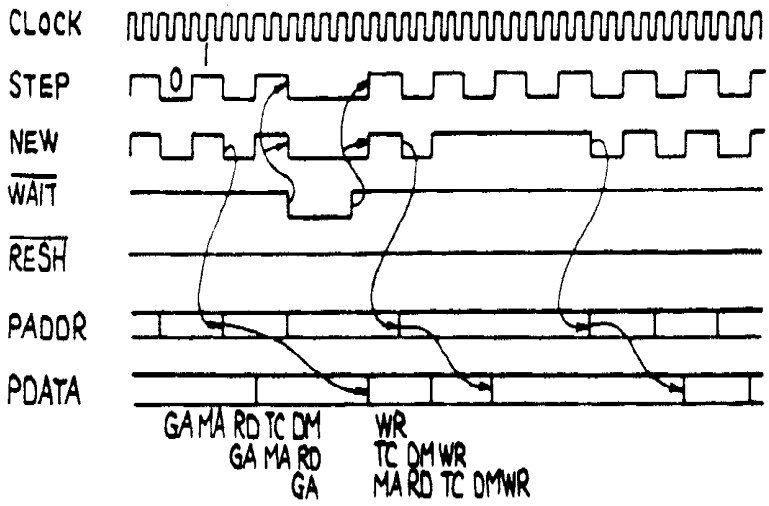
도면5F


30
30a
HDATA
30b
HADDR
8
8
A13
A12
A13v HWRITE
62
=12
=13
A12
A13
68
A13 & HREAD
A13 & HWRITE
66
D CNT LD
OE
A13
60a
OE
WE
S램
A
D
64a
70a
68
64
70
MUPDATE 40a
60b
68a
≥1
≥1
A13 & HWRITE
8
8
60c
3
[n:4]
[3]
[2:0]
n
40

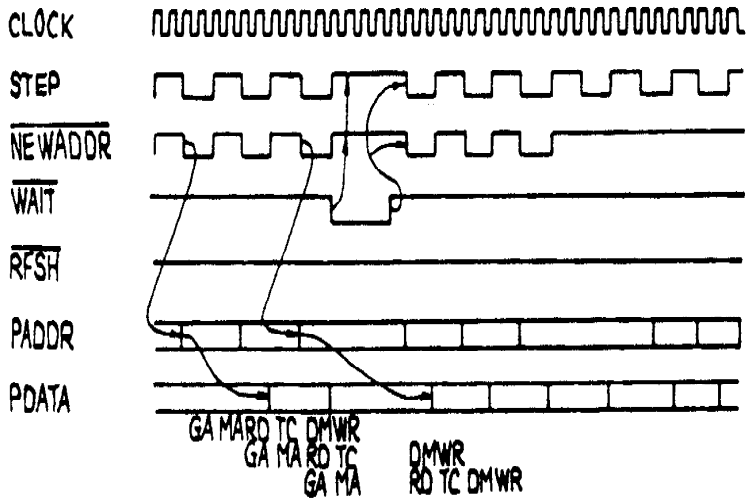
도면6


34b
40'
34
MUPDATE
40'a
42'a
42'
44'
46'
48'
DATAIN
44'a
MADDR
MCTRL
DATAOUT
44'b
34'a
PDATA0
52
PDATA1 38a
PDATA2 38b
PADDR 38c

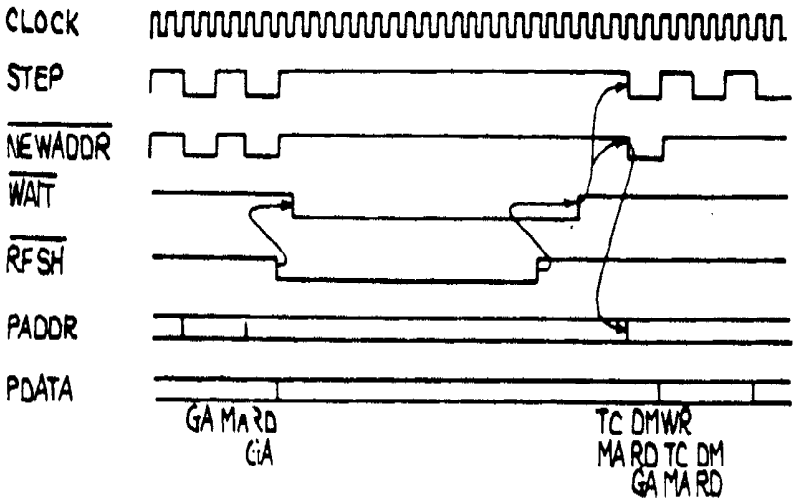
도면7A


CLOCK
STEP
0
NEW
WAIT
RESH
PADDR
PDATA
GA MA RD TC DM
GA MA RD
GA
WR
TC DM WR
MA RD TC DMWR

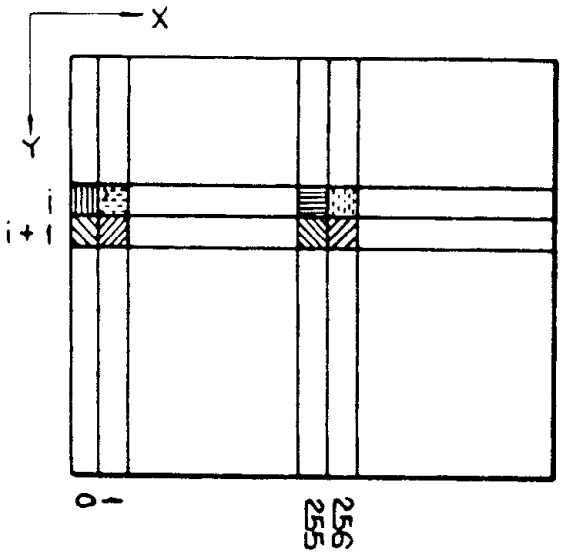
도면7B



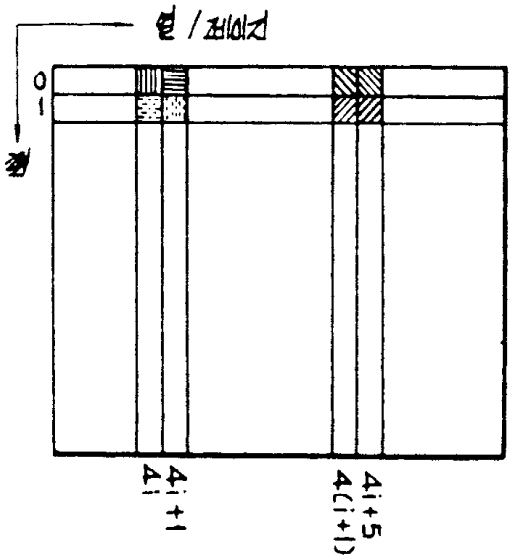
도면7C



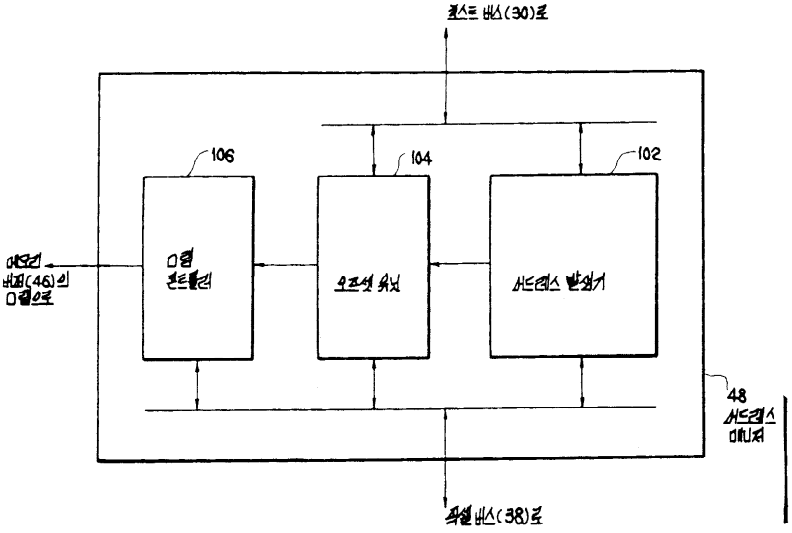
도면8A



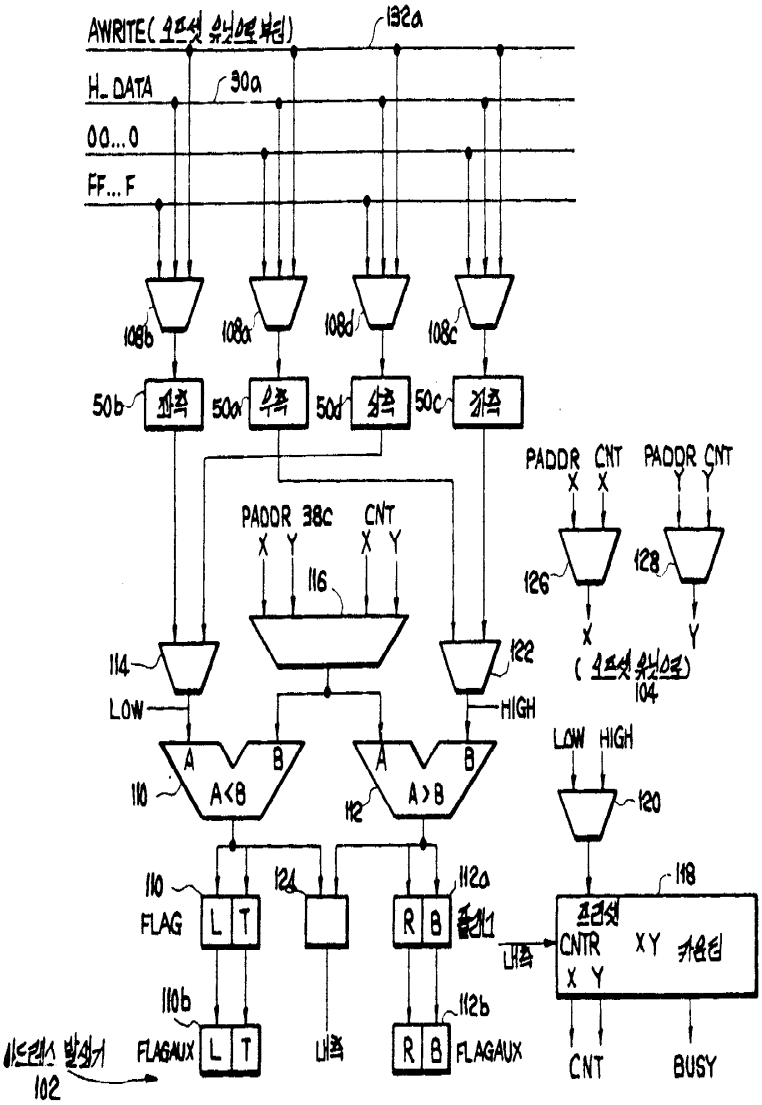
도면8B



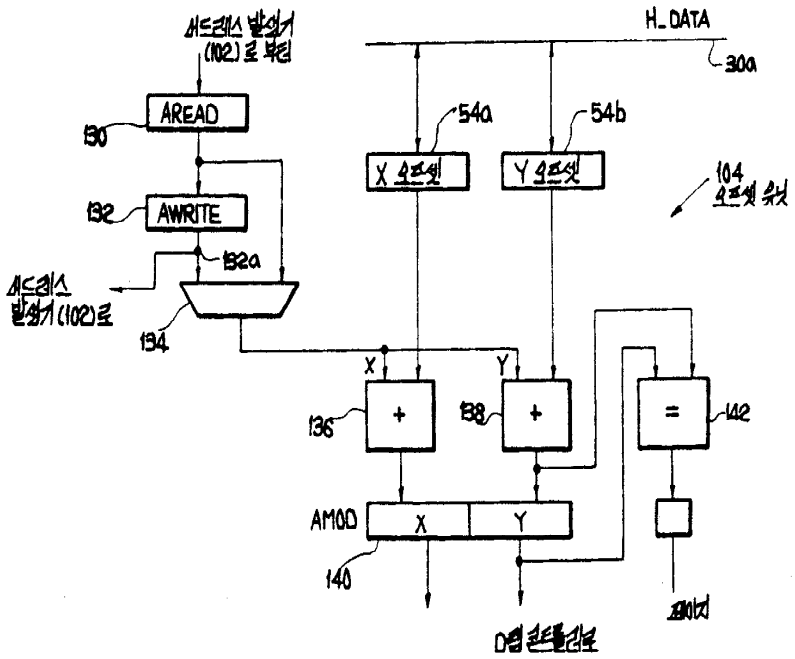
도면9A



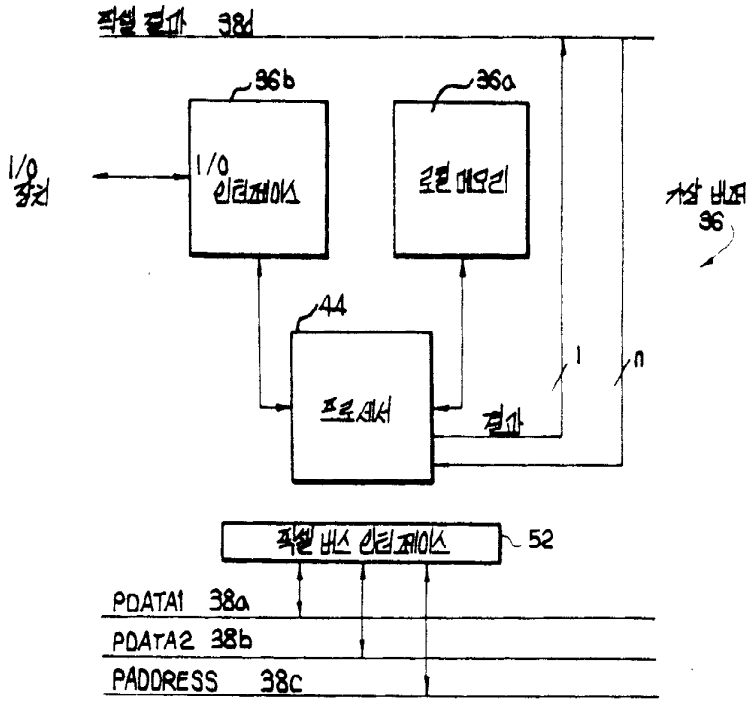
도면9B


AWRITE(오프셋 유닛으로부터)
132a
H_DATA
30a
00...0
FF...F
108b
108a
108d
108c
50b
좌측
50a
우측
50d
상측
50c
감측
PADDR 38c
CNT
X Y
X Y
116
114
122
LOW
HIGH
A
B
A<B
A>B
110
112
PADDR CNT
X X
PADDR CNT
Y Y
126
128
X
Y
(오프셋 유닛으로)
104
LOW HIGH
120
118
110
FLAG
L T
124
112a
R B
플래그1
조건선
CNTR XY 카운팅
X Y
내측
110b
112b
어드레스 발생기
102
FLAGAUX
L T
내측
R B
FLAGAUX
CNT
BUSY

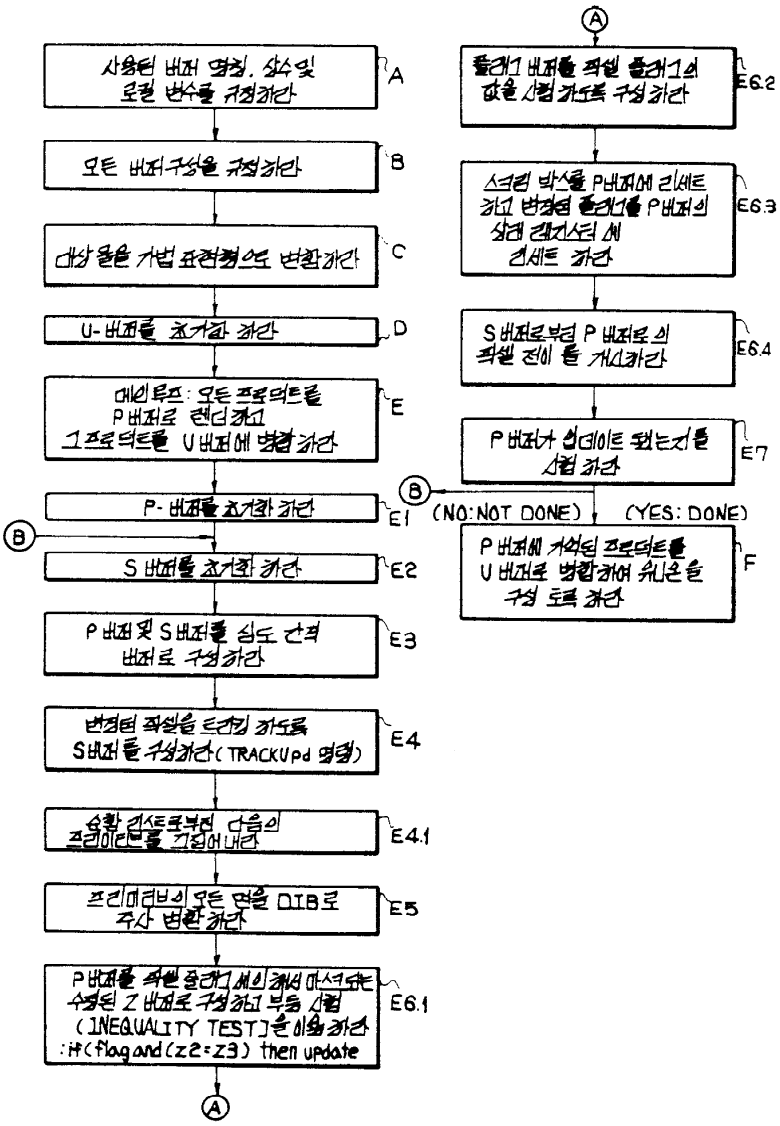
도면9C


H_DATA
30a
AREAD
130
AWRITE
132
132a
54a
54b
X 오프셋
Y 오프셋
104
134
X
Y
136
138
142
AMOD
140

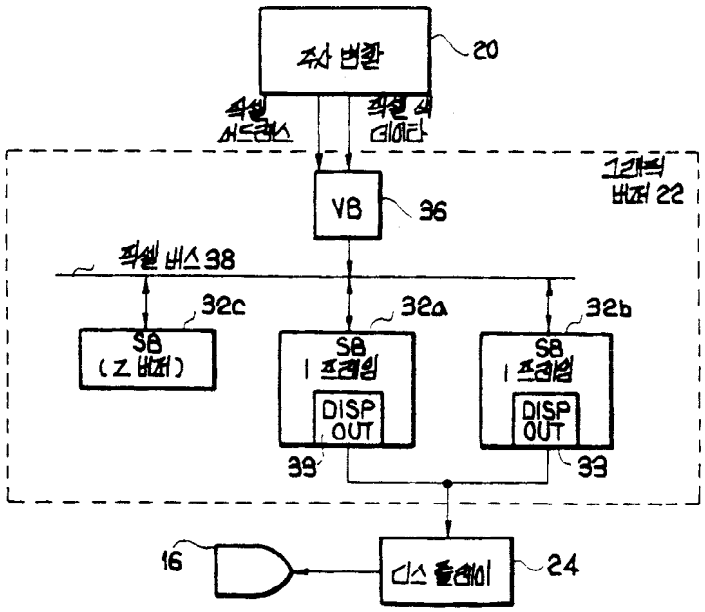
도면10


38d
36b
36a
I/O
인터페이스
로컬 메모리
36
44
프로세서
52
PDATA1 38a
PDATA2 38b
PADDRESS 38c

도면11


사용된 버퍼 명칭, 상수 및 로컬 변수를 규정한다
A
모든 버퍼구성을 규정한다
B
대상 물을 가법 표현형으로 변환한다
C
U-버퍼를 초기화 한다
D
메인루프 : 모든 프로덕트를 P버퍼로 렌더링하고 그 프로덕트를 U버퍼에 병합한다
E
P-버퍼를 초기화 한다
E1
S 버퍼를 초기화 한다
E2
P 버퍼 및 S 버퍼를 심도 근접 버퍼로 구성한다
E3
변경된 픽셀을 트래킹 하도록 S버퍼를 구성한다 (TRACKUpd 명령)
E4
활성 리스트로부터 다음의 프리미티브를 가져와 내린다
E4.1
프리미티브의 모든 면을 DIB로 주사 변환한다
E5
P버퍼를 픽셀 플래그 세이 해서 마스크되는 수정된 Z 버퍼로 구성하고 부등 시험 (INEQUALITY TEST)을 이용한다 : if(flag and (z2<z3) then update
E6.1
플래그 버퍼를 픽셀 플래그의 값을 시험 하도록 구성한다
E6.2
스크린 박스를 P버퍼에 리세트 하고 변경된 플래그를 P버퍼의 상관 레지스터 에 리세트 한다
E6.3
S 버퍼로부터 P 버퍼로 의 픽셀 전이 를 개시한다
E6.4
P 버퍼가 업데이트 됐는지를 시험 한다
E7
(NO: NOT DONE)
(YES: DONE)
P 버퍼에 저장된 프로덕트를 U 버퍼로 병합하여 유니온 을 구성 도록 한다
F

도면12


중앙 변환
20
픽셀 어드레스
픽셀 데이타
그래픽 버퍼 22
VB
36
픽셀 버스 38
32c
32a
32b
SB (Z 버퍼)
SB 프레임
DISP OUT
SB 프레임
DISP OUT
39
33
16
디스플레이
24