(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl. [6]
G06T 1/00

(45) 공고일자 2003년01월24일
(11) 등록번호 10-0356884
(24) 등록일자 2002년10월02일

(21) 출원번호 10-1995-0005073
(22) 출원일자 1995년03월08일

(65) 공개번호 특1995-0033914
(43) 공개일자 1995년12월26일

(30) 우선권주장 08/208413 1994년03월08일 미국(US)
08/208161 1994년03월08일 미국(US)

(73) 특허권자 텍사스 인스트루먼츠 인코포레이티드
미국 텍사스주 75265 달라스 노스센트럴 익스프레스웨이 13500

(72) 발명자 로버트제이.고브
미합중국75075텍사스주플라노스카보로레인1405
칼엠.거타그
미합중국77479텍사스주슈가랜드오크몬트코트3907
제레미아이.골스톤
미합중국77479텍사스주슈가랜드메스퀴트2930
크리스토퍼제이.리드
미합중국77099텍사스주휴스톤버링게임11807
시드니더블유.폴랜드
미합중국77449텍사스주카티프린스조지22307

(74) 대리인 장수길
주성민

심사관 : 김희곤

(54) 투과기록제어를위한투과레지스터를구비한데이터프로세서

요약

투과 검출 데이터 전송 제어기를 갖는 데이터 프로세서에 대해 기술한다. 투과 레지스터는 투과 데이터를 저장한다. 소스 어드레스 제어기는 전송될 데이터의 재현을 위해 소스 어드레스를 계산한다. 비교기는 재현된 데이터를 저장된 투과 데이터에 비교하고, 전송될 데이터가 메모리에 기입되는 지를 표시한다. 비교기가 전송될 데이터가 메모리에 기입되는 것을 표시하는 경우, 데스티네이션 어드레스 제어기는 전송될 데이터를 계산된 데스티네이션 어드레스에서 메모리에 기입한다. 재현된 전송될 데이터는 비교를 위해 소스 레지스터내에 일시적으로 저장된다. 양호한 실시예에서, 전송될 데이터가 투과 데이터와 부합하는 경우에는 데이터는 메모리 내에 기입되지 않는다. 투과 레지스터는 전송될 데이터의 최

소량의 다수배를 저장할 수 있다. 전송될 데이터는 선택된 크기를 갖는 데이터 워드로 조직화된다. 선택된 크기는 전송될 데이터의 최소량의 정수배이다. 비교기는 복수의 데이터 비교기를 포함하는데, 각각의 데이터 비교기는 전송될 데이터의 최소량을 투과 데이터 중 해당 부분과 비교한다. 멀티플렉서는 이들 비교 신호를 수신하고, 선택된 데이터 크기를 표시한다. 멀티플렉서는 전송될 데이터가 투과 레지스터 범위에 적당한 선택된 크기의 데이터 워드의 다수배와 동일한 크기의 메모리에 기입되는 지에 대한 표시를 여러번 제공한다. 양호한 실시예에서, 선택된 데이터 크기는 바이트(8비트), 하프 워드(16비트), 워드(32비트) 및 더블 워드(64비트)일 수 있고, 투과 레지스터는 64비트를 저장한다.

대표도

도 1

명세서

도면의 간단한 설명

제1도는 본 발명의 영상 처리 시스템 실시예의 시스템 아키텍처를 도시한 도면.

제2도는 단일 집적 회로 멀티프로세서의 아키텍처를 도시한 도면.

제3도는 디지탈 영상/그래픽 프로세서의 인터럽트 인에이블 및 인터럽트 플래그 레지스터를 도시한 도면.

제4도는 명령 워드의 필드를 도시한 도면.

제5도는 디지탈 영상/그래픽 프로세서의 통신 레지스터를 도시한 도면.

제6도는 영상 시스템 버스 우선 순위와 동작 방법을 도시한 도면.

제7도는 멀티프로세서 집적 회로내의 크로스바 우선 순위 및 동작 방법을 도시한 도면.

제8도는 본 발명의 실시예의 전송 프로세서 구조를 도시한 도면.

제9도는 LASTPAGE 레지스터의 사용을 도시한 도면.

제10도는 외부 메모리 시퀀서의 상태도와 동작 방법을 도시한 도면.

제11도는 소스 머신 구조를 도시한 도면.

제12도는 패킷 전송 FIFO 버퍼와 캐시 버퍼 산술 동작 및 블럭의 예를 도시한 도면.

제13도는 버퍼 카운트 산술 동작 및 블럭의 예를 도시한 도면.

제14a도는 리틀 엔디언 패킷 전송 FIFO 버퍼 예와 방법을 도시한 도면.

제14b도는 빅 엔디언 패킷 전송 FIFO 예와 방법을 도시한 도면.

제15도는 프로세서의 파라메타 메모리의 사용을 도시한 도면.

제16도는 링크 리스트 패킷 전송 방법 예를 도시한 도면.

제17도는 차원 패킷 전송 방법 예를 도시한 도면.

제18도는 고정-패치 델타-가이드 패킷 전송 방법 예를 도시한 도면.

제19도는 고정-패치 오프셋-가이드 패킷 전송 방법 예를 도시한 도면.

제20도는 고정-패치 오프셋-가이드 룩업 테이블 패킷 전송 방법 예를 도시한 도면.

제21도는 리틀 엔디언 가변-패치 가이드 테이블 포맷을 도시한 도면.

제22도는 빅 엔디언 가변-패치 가이드 테이블 포맷을 도시한 도면.

제23도는 가변-패치 델타-가이드 패킷 전송 방법 예를 도시한 도면.

제24도는 가변-패치 오프셋-가이드 패킷 전송 방법 예를 도시한 도면.

제25도는 소스 투과 전송 파라메타 예를 갖는 차원 패킷 전송을 도시한 도면.

제26도는 필 파라메타 예를 갖는 차원 패킷 전송을 도시한 도면.

제27도는 차원 소스 및 고정된 패치 가이드 데스티네이션 패킷 전송 파라메타 예를 도시한 도면.

제28도는 차원 소스 및 가변 패치 가이드 데스티네이션 패킷 전송 파라메타예를 도시한 도면.

제29도는 차원 블럭 기입 패킷 전송 파라메타 예를 도시한 도면.

제30도는 패킷 전송 파라메타의 패킷 전송 옵션 필드의 엔코딩을 도시한 도면.

제31도는 패킷 전송 타이머 구조를 도시한 도면.

제32도는 중지된 패킷 전송 파라메타의 저장 방식의 예를 도시한 도면.

제33도는 중지된 패킷 서비스 파라메타의 엔코딩을 도시한 도면.

제34도는 리틀 엔디언 모드에서 8x 블럭-기입 비트 리맵핑에서의 비트 리맵핑의 예를 도시한 도면.

제35도는 리틀 엔디언 모드에서 4x 블럭-기입 비트 리맵핑에서의 비트 리맵핑의 예를 도시한 도면.

제36도는 직렬 레지스터 전송 방법 예를 도시한 도면.

제37도의 (a)-(d)는 투과 전송 동작을 도시한 도면.

제38도는 투과 전송 및 바이트 기입 로직 회로 실시예를 도시한 도면.

제39도는 고품위 텔레비전 시스템의 일예의 실시예를 도시한 도면.

제40도는 단일 디지탈 영상/그래픽 프로세서를 갖는 멀티프로세서 집적 회로를 포함하는 컬러 팩시밀리 시스템의 일예의 실시예를 도시한 도면.

< 도면의 주요 부분에 대한 부호의 설명>

1 : 호스트 처리 시스템 2 : 호스트 주변 장치

3 : 영상 장치 4 : 영상 캡쳐 제어기

5 : 비디오 RAM 6 : 비디오 RAM

7 : 비디오 팔레트 8 : 비디오 디스플레이

9 : 메모리

100 : 멀티프로세서 집적 회로

60 : 마스터 프로세서

71, 72, 73, 74 : 디지탈 영상/그래픽 프로세서

80 : 전송 제어기 90 : 프레임 제어기

202 : 프레임 제어기 203 : 긴급한 리프레시

208 : 트리클 리프레시

301 : 내부 메모리 인터페이스

302 : 외부 메모리 인터페이스

303 : 요청 큐잉 및 우선 순위화 로직

310 : 캐시, VRAM, 및 리프레시 제어기

311 : 패킷 전송 FIFO 버퍼

312 : 캐시 버퍼

320 : 소스 머신

330 : 소스 멀티플렉서 및 정렬 로직

340 : 데스티네이션 머신

341 : 데스티네이션 제어 레지스터

342 : 데스티네이션 제어기

350 : 데스티네이션 멀티플렉서 및 정렬 로직

360 : LASTPAGE 레지스터 370 : r1 상태

371 : r2 상태 372 : r3 상태

373 : rf3 상태 374 : tr3 상태

375 : r4 상태 376 : r5 상태

377 : r6 상태 378 : rf6 상태

379 : rf7 상태 380 : rf8 상태

381 : rf9 상태 382 : rhiz 상태

383 : C1, 2 상태 384 : C1, 3 상태

385 : C2, 3 상태 386 : C3 상태

387 : CS1 상태 388 : CS2 상태

389 : CP 상태 390 : CDR 상태

391 : CDW 상태 392 : C1 상태

401 : CACHEADR 레지스터 402 : ABPITCH 레지스터

403 : GTPIPE 레지스터 404 : ASTART 레지스터

405 : BSTART 레지스터 406 : CRNTAPR 레지스터

407 : BCPITCH 레지스터 408 : ACOVINT 레지스터

409 : ACVRRENT 레지스터 410 : BCOVNT 레지스터

411 : BCVRRENT 레지스터 412 : CCVRRENT 레지스터

413 : CBYTES 레지스터 414 : BVFPTR 레지스터

421, 422, 424, 425 : 멀티플렉서

420, 423 : 32 비트 가산기

427 : 어드레스 파이프라인

428 : 바이트 파이프라인

429 : 요청 파이프라인 431 : FCOVNT 레지스터

432 : CHCOVNT 레지스터 433 : 멀티플렉서 1

434 : 가산기 435 : 멀티플렉서 2

436 : 감산기 437 : 멀티플렉서 3

441 : BVFCOVNT 레지스터 442, 443 : 멀티플렉서

444 : 전 가산기 501 : 테이블 포인터

502 : 가이드 테이블 503 : 엔트리의 수 카운터

504 : 최종 패치 스타트

505 : 베이스 어드레스 레지스터

506 : 오프셋 레지스터 511 : 최소 레지스터 PTMIN

512 : 최대 레지스터 PTMAX 513 : 패킷 전송 카운터

514 : 제로 검출기 515 : 타이머 시퀀서

601 : 투과 레지스터

611, 612, 613, 614, 615, 616, 617, 618 : 8 비트 비교기

621, 622, 623, 624 : ½ 워드 AND 게이트들의 집합

631, 632 : 워드 AND 게이트 645 : 멀티플렉서

802 : TV 튜너 803, 812, 822 : 버퍼

811 : CD ROM 플레이어 821 : 케이블 시스템 버스

1301 : 모뎀 1302 : 버퍼

1303 : 비디오 RAM 1304 : 프린터 제어기

1305 : 컬러 프린터

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 디지탈 데이터 처리에 관한 것이며, 상세하게는 특히 디지탈 영상/그래픽 처리와 같은 디지탈 데이터 처리를 위한 멀티프로세서 회로, 아키텍처 및 방법에 관한 것이다.

본 발명의 실시예는 다양하게 응용될 수 있으며, 그 중 일부는 예시적인 배경으로서 본 명세서에 설명된 컴퓨터 그래픽의 분야에 관한 것이다. 비트 맵 그래픽으로서 공지된 컴퓨터 그래픽 분야에 있어서, 컴퓨터 메모리는 개별적인 픽셀 각각에 대한 데이터를 영상 내의 그 픽셀의 위치에 대응하는 메모리 위치에 저장한다. 이 영상은 디스플레이될 영상일 수도 있고, 조작, 저장, 디스플레이 또는 재 전송될 영상일 수도 있다. 비트 맵 컴퓨터 그래픽의 분야는 다이내믹 랜덤 액세스 메모리(DRAM)의 단가 하락 및 용량 증가와 마이크로프로세서의 단가 하락 및 용량 증가에 의해 상당히 이익을 얻고 있다. 소자 부품의 가격 및 성능에서의 이러한 유리한 변화는 보다 크고 복잡한 컴퓨터 영상 시스템을 경제적으로 이용가능하게 하였다.

비트 맵 그래픽의 분야는 영상 데이터 조작에 사용된 처리 형태의 진전에서 여러 단계를 거쳐 왔다. 초기에, 비트 맵 그래픽을 지원하는 컴퓨터 시스템은 모든 비트 맵 동작에 대해 시스템 프로세서를 이용하였다. 이런 형태의 시스템은 몇가지 단점을 갖고 있다. 우선, 컴퓨터 시스템 프로세서는 비트 맵 그래픽을 다루도록 특별히 설계되지 않았다. 일반적인 목적의 처리에 합당한 설계 선택은 비트맵 그래픽 시스템에는 적합하지 않다. 결과적으로 몇가지 루틴한 그래픽 작업은 느리게 동작하였다. 또한, 비트 맵 그래픽의 영상 조작에 필요한 처리가 시스템 프로세서의 계산 용량에 부담을 주기 때문에, 다른 동작들도 느려진다는 것이 곧 발견되었다.

비트 맵 그래픽 처리의 발전의 다음 단계는 전용 하드웨어 그래픽 제어기였다. 이들 장치는, 선, 타원 및 원과 같은 간단한 그림을 시스템 프로세서의 제어하에서 그릴 수 있다. 이들 장치 중 일부는 픽셀 블럭 전송(PixBlt)도 수행할 수 있다. 픽셀 블럭 전송은 영상 데이터를 메모리의 한 부분으로부터 다른 부분으로 이동시키는 메모리 이동 동작이다. 픽셀 블럭 전송은 비디스플레이 메모리에서부터 비트 맵 디스플레이 메모리로의 전송에 의해 특정 형태의 폰트의 영숫자와 같은 표준 영상 요소를 디스플레이 내에 제공하는데 유용하다. 이 기능은 또한 동일한 작은 영상을 비트 맵 디스플레이 메모리의 전체에 전송하여 타일링(tiling)하는데 사용될 수 있다. 가장 빈번하게 사용되는 그래픽 기능들 중 몇가지 기능을 수행하기 위한 내장(built-in) 알고리즘은 시스템 성능을 개선시키는 방법을 제공한다. 또한 그래픽 컴퓨터 시스템은 이러한 하드웨어 그래픽 제어기에서 구현된 몇가지 것들 이외의 다른 기능들을 바람하게 포함할 수 있다. 이들 부가적인 기능들은 시스템 프로세서에 의해 소프트웨어에서 구현될 수 있다. 이들 하드웨어 그래픽 제어기는, 전형적으로 시스템 프로세서가 비트 맵 메모리에 제한된 액세스만을 할 수 있게 한다. 이것은 시스템 소프트웨어가 하드웨어 그래픽 제어기의 고정된 기능 세트를 증대시킬 수 있는 정도를 제한시킨다.

그래픽 시스템 프로세서는 비트 맵 그래픽 처리의 발전의 또 다른 단계를 나타낸다. 그래픽 시스템 프로세서는 마이크로프로세서의 모든 속성을 갖고 비트 맵 그래픽을 위한 특별한 기능도 포함하는 프로그래머블 장치이다. 텍사스 인스트루먼츠 인코포레이트에 의해 제조된 TMS34010 및 TMS34020 그래픽 시스템 프로세서가 이 부류의 장치를 대표한다. 이들 그래픽 시스템 프로세서는 마이크로프로세서와 동일한 방식으로 저장된 프로그램에 응답하며 산술 논리부를 통한 데이터 조작, 레지스터 파일 내의 데이터 저장 및 프로그램 플로우와 외부 데이터 메모리의 제어의 능력을 포함한다. 또한, 이들 장치는 프로그램 제어 하에 동작하는 특수 목적 그래픽 조작 하드웨어를 포함한다. 이들 그래픽 시스템 프로세서의 명령 세트 내의 부수적 명령들은 특수 목적 그래픽 하드웨어를 제어한다. 이 명령들과 이 명령들을 지원하는 하드웨어는, 많은 콘텍스트에 유용한 기초 레벨의 그래픽 기능을 수행하도록 선택된다. 그러므로 그래픽 시스템 프로세서는 특정한 문제를 위해 선택된 알고리즘을 사용하는 여러가지 다른 그래픽 응용을 위해 프로그램될 수 있다. 이것은 하드웨어 제어기를 프로그램된 마이크로프로세서로 변경시킨 경우에서 제공된 것과 유사한 유용성의 증가를 제공한다. 이러한 그래픽 시스템 프로세서들은 마이크로프로세서와 동일한 방식으로 프로그램 가능한 장치이기 때문에, 이들은 독립형 그래픽 프로세서, 시스템 프로세서에 종속되거나 그래픽 제어기에 강하게 결합된 코프로세서로서 동작할 수 있다.

여러가지 분야가 보다 단가 효율적이고 강력한 그래픽 동작을 바람직하게 사용할 것이다. 이러한 분야로는, 영상 회의, 풀 모션(full motion) 비디오를 이용하는 멀티 미디어 컴퓨팅, 고품위 텔레비젼, 컬러 팩시밀리, 스마트 포토카피어(smart photocopier), 영상 인식 시스템 및 디지탈 포토그래픽을 포함한다. 이러한 분야들 각각은 독특한 문제를 나타낸다. 영상 데이터 압축(compression) 및 압축 해제(decompression)는 몇가지 이들 응용에서 공통 테마이다. 송신 대역폭의 양과 영상 및 특정한 풀 모션 비디오에 요청되는 저장 용량의 양은 막대하다. 최종 영상 품질을 허용 가능한 수준으로 유지시키는 효율적인 비디오 압축 및 압축 해제가 없다면, 이들 응용은 송신 대역폭 및 저장 용량과 관련된 가격에 의해 제약받을 것이다. 또한, 본 기술에서는, 영상 인식 등의 영상 처리 기능과 디스플레이 제어와 같은 그래픽 기능 모두를 지원할 수 있는 단일 시스템이 필요하다.

본 발명은 투과 검출 데이터 전송 제어기를 갖는 데이터 프로세서이다. 메모리는 다수의 저장 위치를 갖는다. 전송 제어기는 투과 데이터를 저장하는 투과 레지스터를 포함한다. 소스 어드레스 제어기는 메모리로부터 전송될 데이터의 재현을 위해 소스 어드레스를 계산한다. 비교기는 재현된 데이터를 저장된 투과 데이터와 비교하여, 전송될 데이터가 메모리에 기입될지의 여부를 결정한다. 비교기가 전송될 데이터가 메모리에 기입될 것임을 나타내면, 데스티네이션 어드레스 제어기는 전송되는 데이터를 메모리에 기입한다. 재현된 전송될 데이터는 비교를 위해 소스 레지스터 내에 일시적으로 저장된다. 양호한 실시예에서, 데이터가 투과 데이터와 일치한다면 데이터는 메모리 내에 기입되지 않는다.

투과 레지스터는 전송될 데이터의 최소량의 여러 배를 저장할 수 있다. 이 경우에, 비교기는 전송될 데이터의 최소량의 단위로 재현된 데이터를 투과 데이터와 비교한다. 다음에, 비교기는 데이터가 메모리에 기입될 지의 여부를 나타내는 복수의 표시를 제공한다. 비교기는, 전송될 데이터의 최소량 단위마다, 데이터가 메모리 내에 저장될 것인지의 여부를 나타내는 하나의 표시를 제공한다. 전송될 데이터는 선택된 크기를 갖는 데이터 워드로 조직화된다. 선택된 크기는 전송될 데이터의 최소량의 정수배이다. 비교기는 다수의 데이터 비교기를 포함하는데, 각각의 데이터 비교기는 전송될 데이터의 최소량을 투과 데이터의 해당하는 부분과 비교한다. 멀티플렉서는 이들 비교 신호와 선택된 데이터 크기의 표시를 수신한다. 멀티플렉서는 전송될 데이터가 투과 레지스터 범위에 적당한 선택된 크기의 데이터 워드의 배수와 같게 기입되는지의 여부에 대한 복수의 표시를 제공한다. 이러한 표시들은 검출된 부합성 및 선택된 데이터 크기로 형성된다. 양호한 실시예에서, 선택된 데이터 크기는 1 바이트(8비트), 하프 워드(16비트), 워드(32비트) 및 더블 워드(64비트)일 수 있다. 양호한 실시예에서, 투과 레지스터는 64비트를 저장한다.

메모리 및 데이터 전송 제어기는 데이터 프로세서, 내부 메모리 및 외부 메모리 인터페이스와 함께 하나의 반도체 칩 내에 실장될 수 있다. 양호한 실시예에서, 내부 메모리 및 외부 메모리 인터페이스는 단일 통합 어드레스 공간을 포함한다. 제1 세트의 어드레스는 내부 메모리에 충당되고, 나머지 제2 세트의 어드레스는 외부 메모리에 충당된다. 데이터 프로세서는 투과 데이터를 지정한다. 반도체 칩은 양호하게는 내부 메모리 및 데이터 전송 인터페이스에 접속된 적어도 하나의 추가 데이터 프로세스를 포함하는 마이크로프로세서이다.

본 발명의 실시예의 이러한 특징 및 다른 특징은 첨부 도면과 함께 다음에 설명될 것이다.

제1도는 본 발명에 따른 영상 및 그래픽 처리를 위해 구성된 멀티프로세서 집적 회로를 포함하는 영상 데이터 처리 시스템의 블럭도이다. 이 데이터 처리 시스템은 호스트 처리 시스템(1)을 포함한다. 호스트 처리 시스템(1)은 제1도의 데이터 처리 시스템의 호스트 시스템을 위한 데이터 처리를 제공한다. 호스트 처리 시스템(1) 내에는 프로세서, 적어도 하나의 입력 장치, 장기(long term) 저장 장치, 판독 전용 메모리, 랜덤 액세스 메모리 및 호스트 시스템 버스에 결합된 적어도 하나의 호스트 주변 기기(2)가 포함되어 있다. 그 처리 기능으로 인해, 호스트 처리 시스템(1)은 영상 데이터 처리 시스템의 기능을 제어한다.

멀티프로세서 집적 회로(100)는 제1도의 영상 데이터 처리 시스템의 영상 동작을 위한 데이터 조작 및 계산을 포함하는 대부분의 데이터 처리를 제공한다. 멀티프로세서 집적 회로(100)는 영상 시스템 버스에 양방향 결합되며 이러한 영상 시스템 버스를 통해 호스트 처리 시스템(1)과 통신한다. 제1도의 구성에서, 멀티프로세서 집적 회로(100)는 호스트 처리 시스템(1)과 독립적으로 동작한다. 그러나, 멀티프로세서 집적 회로(100)는 호스트 처리 시스템(1)에 응답한다.

제1도의 2개의 영상 시스템을 도시한다. 영상 장치(3)는 영상 입력 장치로서 기능하는 문서 스캐너, 전하 결합 장치 스캐너 또는 비디오 카메라를 나타낸다. 영상 장치(3)는, 영상을 디지탈화하여 래스터 스캔 프레임으로 형성하는 기능을 하는 영상 캡쳐 제어기(4)에 이 영상을 공급한다. 이 프레임 캡쳐 프로세스는 멀티 프로세서 집적 회로(100)로부터의

신호에 의해 제어된다. 이렇게 형성된 영상 프레임은 비디오 랜덤 액세스 메모리(5) 내에 저장된다. 비디오 랜덤 액세스 메모리(5)는 멀티프로세서 집적 회로(100)에 의한 영상 처리를 위한 데이터 전송을 허용하는 영상 시스템 버스를 경유하여 액세스될 수 있다.

제2 영상 시스템은 비디오 디스플레이를 구동시킨다. 멀티프로세서 집적 회로(100)는 디스플레이된 영상의 지정을 위해 픽셀 맵을 경유하여 비디오 랜덤 액세스 메모리(6)와 통신한다. 멀티프로세서 집적 회로(100)는 영상 시스템 버스를 경유하여 비디오 랜덤 액세스 메모리 내에 저장된 영상 데이터를 제어한다. 이 영상에 대응하는 데이터는 비디오 랜덤 액세스 메모리(6)로부터 재현(recall)되어 비디오 팔레트(7)에 공급된다. 비디오 팔레트(7)는 이 재현된 데이터를 다른 컬러 공간으로 변형시켜 픽셀당 비트수 등을 확장할 수 있다. 이러한 변환은 룩업 테이블을 통해 달성될 수 있다. 비디오 팔레트(7)는 비디오 디스플레이(8)를 구동시키기 위해 적절한 비디오 신호를 발생시킨다. 이러한 비디오 신호가 아날로그 신호인 경우, 비디오 팔레트(7)는 적합한 디지털-아날로그 변환을 포함한다. 비디오 팔레트(7)로부터 출력된 비디오 레벨 신호는 컬러, 포화도 및 취도 정보를 포함할 수 있다. 멀티프로세서 집적 회로(100)는 비디오 팔레트(7) 내에 저장된 데이터를 제어하여, 데이터 변형 처리 및 영상 프레임의 타이밍을 제어한다. 멀티프로세서 집적 회로(100)는 비디오 팔레트(7)의 제어를 통해 비디오 디스플레이 영상의 프레임 당 라인 길이 및 라인수, 동기화, 회귀, 및 블랭킹 신호를 제어할 수 있다. 중요하게, 멀티프로세서 집적 회로(100)는 그래픽 디스플레이 정보가 비디오 랜덤 액세스 메모리(6) 어디에 저장되는지를 결정하고 제어한다. 후속적으로, 비디오 랜덤 액세스 메모리(6)로부터 판독 중에, 멀티프로세서 집적 회로(100)는 비디오 랜덤 액세스 메모리(6)로부터의 판독 시퀀스, 액세스될 어드레스, 및 비디오 디스플레이(8) 상에 원하는 그래픽 영상을 발생시키는데 필요한 제어 정보를 결정한다.

비디오 디스플레이(8)는 사용자가 볼 수 있도록 지정된 비디오 디스플레이를 발생시킨다. 2가지의 광범위하게 사용되는 기술이 있다. 첫번째 기술은 각 픽셀의 색상, 색조, 휘도 및 포화도애 관련하여 비디오 데이터를 지정한다. 두번째 기술에서는, 각 픽셀에 대해 적, 청 및 녹의 색상 레벨이 지정된다. 비디오 팔레트(7)와 비디오 디스플레이(8)는 선택된 기술과 호환성이 있도록 설계되어 제조된다.

제1도는 영상 시스템 버스에 결합된 부가 메모리(9)를 도시한다. 이 부가 메모리는 부가 비디오 랜덤 액세스 메모리, 동적 랜덤 액세스 메모리, 정적 랜덤 액세스 메모리 또는 판독 전용 메모리를 포함할 수 있다. 멀티프로세서 집적 회로(100)는 메모리(9) 내에 저장된 프로그램에 의해 전체적으로 또는 부분적으로 제어될 수 있다. 이 메모리(9)는 또한 여러가지 형태의 그래픽 영상 데이터를 저장할 수 있다. 또한, 멀티프로세서 집적 회로(100)는 양호하게는 비디오 랜덤 액세스 메모리, 동적 랜덤 액세스 메모리 및 정적 랜덤 액세스 메모리용 메모리 인터페이스 회로를 포함한다. 그러므로, 시스템은 어떤 비디오 랜덤 액세스 메모리(5 또는 6) 없이도 멀티프로세서 집적 회로(100)를 사용하여 구성될 수 있다.

제1도는 송수신기(16)를 도시한다. 송수신기(16)는 영상 시스템 버스와 통신 채널 간의 번역 및 양방향 통신을 제공한다. 송수신기(16)를 이용하는 시스템의 한 예가 영상 회의이다. 제1도에 도시된 영상 데이터 처리 시스템은 영상 장치(3) 및 영상 캡쳐 제어기(4)를 이용하여 사람의 비디오 영상을 제1 위치에 형성한다. 멀티프로세서 집적 회로(100)는 비디오 압축을 제공하고 압축된 비디오 신호를 송수신기(16) 및 통신 채널을 통해 다른 위치에 있는 유사한 영상 데이터 처리 시스템에 송신한다. 송수신기(16)는 유사하게 압축된 비디오 신호를 통신 채널을 통해 원격 영상 데이터 처리 시스템으로부터 수신한다. 멀티프로세서 집적 회로(100)는 이 수신된 신호를 압축 해제하고 비디오 랜덤 액세스 메모리(6) 및 비디오 팔레트(7)를 제어하여, 비디오 디스플레이(8)에 대응하는 압축 해제된 비디오 신호를 디스플레이 한다. 이것은 영상 데이터 처리 시스템이 송수신기(6)를 이용하는 유일한 예가 아니라는 것에 유의해야 한다. 또한, 양방향 통신은 동일한 형태의 신호일 필요는 없다. 예를 들어, 인터랙티브 케이블 텔레비젼 신호에 있어서, 케이블 시스템 헤드인은 통신 채널을 통해 압축된 비디오 신호를 영상 데이터 처리 시스템에 송신한다. 영상 데이터 처리 시스템은 송수신기(16) 및 통신 채널을 통해 제어 및 데이터 신호를 케이블 시스템 헤드인에 역 전송할 수 있다.

제1도는 호스트 처리 시스템(1)을 포함하는 시스템에 구현된 멀티프로세서 집적 회로(100)를 도시한다. 본 기술에 숙련된 자들은 본 명세서로부터 멀티프로세서 집적 회로(100)가 유용한 시스템의 유일한 프로세서로서 사용될 수 있음을 알 수 있을 것이다. 이러한 시스템에서, 멀티프로세서 집적 회로(100)는 시스템의 모든 기능을 수행하도록 프로그램된다.

이 멀티프로세서 집적 회로(100)는 영상 처리용으로 사용되는 시스템에 특히 유용하다. 멀티프로세서 집적 회로(100)는 양호하게는 복수의 동일 프로세서를 포함한다. 이 프로세서들 각각은 디지탈 영상/그래픽 프로세서라고 부를 것이다. 이런 명명은 단지 설명의 편의상 이루어진 것이다. 본 발명을 실시하는 프로세서는 단일 집적 회로 또는 다수의 집적 회로 상에 개별적으로 제조된 프로세서일 수 있다. 단일 집적 회로 상에 실시된다면, 이 단일 집적 회로는 또한 디지탈 영상/그래픽 프로세서에 의해 사용되는 판독 전용 메모리 및 랜덤 액세스 메모리를 선택적으로 포함할 수 있다.

제2도는 마이크로프로세서 집적 회로(100)의 아키텍처를 도시한다. 멀티프로세서 집적 회로는 각각 복수의 섹션으로 분할된 2개의 랜덤 액세스 메모리(10 및 20); 크로스바(50); 마스터 프로세서(60); 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74); 시스템 메모리로의 액세스를 조정하는 전송 제어기(80); 및 개개의 제1 및 제2 영상 메모리에의 액세스를 제어할 수 있는 프레임 제어기(90)를 포함한다. 멀티프로세서 집적 회로(100)는 멀티미디어 컴퓨팅에서와 같이, 영상 처리 및 그래픽 동작에 유용한 고등급의 동작 병렬성을 제공한다. 영상 및 그래픽 처리 이외에도 이들 프로세서가 유용한 컴퓨팅 응용이 존재하므로, 프로세서(71, 72, 73 및 74) 영상/그래픽 프로세서로 하는 것은 단지 설명의 편의를 위한 것이다.

멀티프로세서 집적 회로(100)는 2개의 랜덤 액세스 메모리를 포함한다. 랜덤 액세스 메모리(10)는 주로 마스터 프로세서(60)에 전용된다. 이것은 2개의 명령 캐시 메모리(11과 12), 2개의 데이터 캐시 메모리(13과 14) 및 파라메타 메모리(15)를 포함한다. 이들 메모리 섹션은 물리적으로는 동일하지만 상이하게 접속되어 사용된다. 랜덤 액세스 메모리(20)는 마스터 프로세서(60) 및 각각의 디지탈 영상/그래픽 프로세서(70, 72, 73 및 74)에 의해 액세스될 수 있다. 각각의 디지탈 영상/그래픽 프로세서(70, 72, 73 및 74)는 5개의 대응하는 메모리 섹션을 갖고 있다. 이들은 명령 캐시 메모리, 3개의 데이터 메모리 및 하나의 파라메타 메모리를 포함한다. 그러므로, 디지탈 영상/그래픽 프로세서(71)는 대응하는 명령 캐시 메모리(21), 데이터 메모리(22, 23, 24) 및 파라메타 메모리(25)를 가지며, 디지탈 영상/그래픽 프로세서(72)는 대응하는 명령 캐시 메모리(26), 데이터 메모리(27, 28, 29) 및 파라메타 메모리(30)를 가지고, 디지탈 영상/그래픽 프로세서(73)는 대응하는 명령 캐시 메모리(31), 데이터 메모리(32, 33, 34) 및 파라메타 메모리(35)를 가지며, 디지탈 영상/그래픽 프로세서(74)는 대응하는 명령 캐시 메모리(36), 데이터 메모리(37, 38, 39) 및 파라메타 메모리(40)를 가진다. 랜덤 액세스 메모리(10)의 섹션과 마찬가지로, 이들 메모리 섹션은 물리적으로 동일하지만 상이하게 접속되어 사용된다. 메모리(10가 20)의 이들 메모리 섹션의 각각은 예를 들어 2K 바이트를 포함하며 전체 메모리가 5K 바이트의 멀티프로세서 집적 회로(100)가 된다.

멀티프로세서 집적 회로(100)는 복수의 개별적인 병렬 데이터 전송을 사용하여 프로세서와 메모리 간에 고속의 데이터 전송을 제공하도록 구성된다. 크로스바(50)는 이들 데이터 전송을 인에이블한다. 각각의 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 매 사이클마다 동시에 동작할 수 있는 3개의 메모리 포트를 갖는다. 명령 포트(I)는 대응하는 명령 캐시로부터 64 비트 데이터 워드를 페치할 수 있다. 로컬 데이터 포트(L)는 32 비트 데이터 워드를 그 디지탈 영상/그래픽 프로세서에 대응하는 데이터 메모리 또는 파라메타 메모리로부터 판독하거나 기입할 수 있게 한다. 글로벌 데이터 포트(G)는 32비트 데이터 워드를 데이터 메모리 또는 파라메타 메모리 또는 랜덤 액세스 메모리(20) 중 어느 하나로부터 판독하거나 기입한다. 마스터 프로세서(60)는 2개의 메모리 포트를 포함한다. 명령 포트(I)는 명령 캐시(11과 12) 중 어느 하나로부터 32 비트 명령 워드를 페치할 수 있다. 데이터 포트(C)는 랜덤 액세스 메모리(10)의 데이터 캐시(13 또는 14), 파라메타 메모리(15), 또는 랜덤 액세스 메모리(20)의 임의의 데이터 메모리, 파라미터 메모리 중 어느 하나로부터 32 비트 데이터 워드를 판독하고 또는 32 비트 데이터 워드를 기입한다. 전송 제어기(80)는 데

이터 포트(C)를 경유하여 랜덤 액세스 메모리(10 도는 20)의 섹션들 중 어느 하나에 액세스할 수 있다. 그러므로 15개의 병렬 메모리 액세스가 어떤 단일 메모리 사이클에서 요구될 수 있다. 랜덤 액세스 메모리(10과 20)는 많은 병렬 액세스를 지원하기 위해 25개의 메모리로 분할된다.

크로스바(50)는 마스터 프로세서(60), 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 및 전송 제어기(80)와 메모리(10 및 20)의 접속을 제어한다. 크로스바(50)는 로우와 행 내에 배치된 다수의 크로스포인트(51)를 포함한다. 크로스포인트(51)의 각각의 컬럼은 단일 메모리 섹션과 대응하는 범위의 어드레스에 대응한다. 프로세서는 이 프로세서에 의해 출력되는 어드레스의 최상위 비트를 통하여 메모리 섹션들 중의 하나로의 액세스를 요청한다. 프로세서에 의해 출력된 이 어드레스는 로우를 따라 이동한다. 그 어드레스를 갖는 메모리 섹션에 대응하는 크로스포인트(51)는 메모리 섹션에의 액세스를 승인 또는 거부함으로써 응답한다. 다른 프로세서가 현재의 메모리 사이클 중에 그 메모리 섹션으로의 액세스를 요청하지 않았다면, 크로스포인트(51)는 로우와 컬럼을 결합시킴으로써 액세스를 승인한다. 이것은 어드레스를 메모리 섹션에 공급한다. 메모리 섹션은 그 어드레스에서의 데이터 액세스를 허용함으로써 응답한다. 이 데이터 액세스는 데이터 판독 동작일 수도 있고 데이터 기입 동작일 수도 있다.

하나 이상의 프로세서가 동일한 메모리 섹션으로의 액세스를 동시에 요청하는 경우, 크로스바(50)는 요청하는 프로세서들 중의 단 하나에 대한 액세스만을 승인한다. 크로스바(50)의 각 컬럼 내의 크로스포인트(51)는 우선 순위 계층에 기초하여 통신하고 액세스를 승인한다. 동일한 랭크를 갖는 액세스에 대한 2개의 요청이 동시에 발생하는 경우, 크로스바(50)는 라운드 로빈에 기초하여 액세스를 승인하는데, 마지막에 승인된 액세스를 갖는 프로세서가 가장 낮은 우선 순위를 갖는다. 각각 승인된 액세스는 요청을 서비스하는데 필요로 되는 한 지속된다. 프로세서는 매 메모리 사이클마다 그 어드레스를 변경하여, 크로스바(50)는 매 사이클마다 프로세서와 메모리 섹션간의 상호 접속을 변화시킬 수 있게 한다.

마스터 프로세서(60)는 양호하게는 멀티프로세서 집적 회로(100)를 위한 주제어 기능을 수행한다. 마스터 프로세서(60)는 양호하게는 하드웨어 부동 소수점 계산부를 포함하는 32 비트의 감소된 명령 세트 컴퓨터(RISC) 프로세서이다. RISC 아키텍처에 따르면, 메모리에 대한 모든 액세스는 로드 및 저장 명령에 의해 수행되고 대부분의 정수 및 논리 동작은 레지스터 상에서 단일 사이클 내에서 수행된다. 그러나 부동 소수점 계산부가 정수 및 논리부에서 사용된 것과 동일한 레지스터 파일을 이용하는 경우에는, 동작을 수행하는데 일반적으로 몇 개의 사이클이 필요할 것이다. 레지스터 스코어보드는 올바른 레지스터 액세스 시퀀스가 유지되는 것을 보장한다. RISC 아키텍처는 영상 처리에서의 제어 기능에 적합하다. 부동 소수점 계산부는 영상 처리에 중요한 영상 로테이션 기능의 신속한 계산을 허용한다.

마스터 프로세서(60)는 명령 캐시 메모리(11) 또는 명령 캐시 메모리(12)로부터 명령 워드를 페치한다. 이와 마찬가지로, 마스터 프로세서(60)는 데이터 캐시(13) 또는 데이터 캐시(14)로부터 데이터를 페치한다. 각각의 메모리 섹션이 2K 바이트의 메모리를 포함하므로, 4K 바이트의 명령 캐시 및 4K 바이트의 데이터 캐시가 존재한다. 캐시 제어는 마스터 프로세서(60)의 통합 기능이다. 앞서 언급된 바와 같이, 마스터 프로세서(60)는 또한 크로스바(50)를 경유하여 다른 메모리 섹션들에 액세스할 수 있다.

4개의 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 각각 고도의 병렬 디지탈 신호 프로세서(DSP) 아키텍처를 갖고 있다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 데이터부, 어드레스부, 및 프로그램 흐름 제어부인 3개의 분리된 부분을 사용하여 고도의 동작 병렬성을 달성시킨다. 이들 3개의 유닛은 명령 파이프라인 내의 서로 다른 명령에 기초하여 동시에 동작한다. 또한 이들 유닛 각각은 내부 병렬성을 포함한다.

디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 다중 명령 다중 데이터 모드(MIMD)에서 독립적인 명령 스트림을 실행할 수 있다. MIMD 모드에서, 각각의 디지탈 영상/그래픽 프로세서는 독립적이거나 또는 상호 협력적일 수 있는 그것의 대응하는 명령 캐시로부터 개별적인 프로그램을 실행한다. 후자의 경우에서 크로스바(50)는 공유된 메모리와 함께 프로세서 간 통신을 가능하게 한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 동기화된 MIMD 모드에서도 동작할 수 있다. 동기화된 MIMD 모드에서, 각각의 디지탈 영상/그래픽 프로세서의 프로그램 플로우 제어부(130)는 모든 동기화된 프로세서가 진행할 준비가 되어 있을 때까지 다음 명령을 페칭하는 것을 금지한다. 이 동기화된 MIMD 모드는 디지탈 영상/그래픽 프로세서의 별도의 프로그램이 긴밀하게 결합된 동작에서 록크 단계로 실행되도록 한다.

디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 단일 명령 다중 데이터 모드(SIMD)에서 다른 데이터 상에서 동일한 명령을 실행할 수 있다. 이 모드에서, 4개의 디지탈 영상/그래픽 프로세서를 위한 단일 명령 스트림은 명령 캐시 메모리(21)로부터 나온다. 디지탈 영상/그래픽 프로세서(71)는 페칭 및 브랜칭 동작을 제어하고 크로스바(50)는 동일한 명령을 다른 디지탈 영상/그래픽 프로세서(72, 73 및 74)에 공급한다. 디지탈 영상/그래픽 프로세서(71)가 모든 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)에 대한 명령 페치를 제어하기 때문에, 디지탈 영상/그래픽 프로세서는 SIMD 모드에서 고유하게 동기화된다.

전송 제어기(80)는 멀티프로세서 집적 회로(100)용의 조합된 직접 메모리 액세스(DMA) 머신 및 메모리 인터페이스이다. 전송 제어기(80)는 지능적으로 큐(queue)하고, 우선 순위를 설정하며, 5개의 프로그래머블 프로세서의 데이터 요청 및 캐시 미스를 서비스한다. 마스터 프로세서(60) 및 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 전송 제어기(80)를 경유하여 멀티프로세서 집적 회로(100) 외부에 있는 메모리 및 시스템에 액세스한다. 데이터 캐시 또는 명령 캐시 미스는 전송 제어기(80)에 의해 자동적으로 조종된다. 캐시 서비스(S) 포트는 이러한 캐시 미스를 전송 제어기(80)에 전송한다. 캐시 서비스 포트(S)는 프로세서로부터 정보를 판독하지만 메모리로부터는 판독하지 않는다. 마스터 프로세서(60) 및 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 링크된 리스트 패킷 전송으로서 전송 제어기(80)로부터 데이터 전송을 요청할 수 있다. 이러한 링크된 리스트 패킷 전송은 다차원 블럭의 정보를 멀티프로세서 집적 회로(100) 내에 있을 수도 있고 멀티프로세서 집적 회로(100) 외부에 있을 수도 있는, 소스 메모리 어드레스 및 데스티네이션 메모리 어드레스들 사이에 전송되게 할 수 있다. 또한, 전송 제어기(80)는 그 데이터를 보유하도록 주기적 리프레시를 요청하는 동적 랜덤 액세스 메모리(DRAM)를 위한 리프레시 제어기를 포함하는 것이 바람직하다.

프레임 제어기(90)는 멀티프로세서 집적 회로(100)와 외부 영상 캡쳐 및 디스플레이 시스템 사이의 인터페이스이다. 프레임 제어기(90)는 캡쳐 및 디스플레이 장치를 제어하고, 이 장치들과 메모리 사이의 데이터의 이동을 자동적으로 관리한다. 이 목적을 위해, 프레임 제어기(90)는 2개의 독립적인 영상 시스템에 대해 동시 제어를 제공한다. 프레임 제어기(90)의 응용은 사용자에 의해 제어되지만, 이들은 전형적으로 영상 캡쳐용의 제1 영상 시스템과 영상 디스플레이용의 제2 영상 시스템을 포함한다. 이들 영상 시스템은 통상적으로 프레임 그래버(grabber) 또는 프레임 버퍼 저장에 이용되는 독립적인 프레임 메모리들을 포함할 것이다. 프레임 제어기(90)는 양호하게는 리프레시 및 시프트 레지스터 제어를 통해 비디오 동적 랜덤 액세스 메모리(VRAM)를 제어하도록 동작한다.

멀티프로세서 집적 회로(100)는 대규모 영상 처리용으로 설계된다. 마스터 프로세서(60)는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 동작을 결집시키고, 이들이 생성한 결과를 해석하는 내장 제어를 제공한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 픽셀 분석 및 조작에 잘 맞는다. 픽셀이 데이터에서는 높지만 정보에서는 낮은 것으로 간주되는 경우, 전형적인 응용에서, 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 픽셀을 잘 조사하여 원시 데이터(raw data)를 정보로 전환시킨다. 다음에 이 정보는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 또는 마스터 프로세서(60)에 의해 분석될 수 있다. 크로스바(50)는 프로세서 간 통신을 중재한다. 크로스바(50)는 멀티프로

세서 집적 회로(100)가 공유 메모리 시스템으로서 구현될 수 있도록 한다. 메시지 통과는 이 아키텍처에서 통신의 주 형태로 될 필요는 없다. 그러나, 메시지는 공유 메모리를 통해 통과될 수 있다. 본 실시예에서, 각각의 디지탈 영상/그래픽 프로세서, 크로스바(50)의 대응 섹션 및 메모리(20)의 대응 섹션은 동일한 폭을 가진다. 이것은 동일 핀 아웃을 유지하면서 디지탈 영상/그래픽 프로세서 및 대응 메모리의 부가 또는 제거를 모듈하게(modularly) 수용함으로써 아키텍처 신축성을 허용한다.

한 실시예에서, 멀티프로세서 집적 회로(100)의 모든 부분은 0.6 ㎛의 형상 크기를 사용하여 상보적 금속 산화물 반도체(CMOS) 내에 형성되는 단일 집적 회로상에 배치된다. 멀티프로세서 집적 회로(100)는 256개의 핀을 갖는 핀 그리드 어레이 팩키지 내에 적합하게 구성된다. 입력 및 출력들은 예를 들어 트랜지스터-트랜지스터 논리(TTL)의 논리 전압과 호환성이 있다. 멀티프로세서 집적 회로(100)는 약 300 만개의 트랜지스터를 포함하고, 50 MHz의 클럭 속도를 이용한다.

제3도는 인터럽트 인에이블 레지스터 INTEN(110)과 인터럽트 플래그 레지스터(INTFLG)의 필드 정의를 도시한다. "r" 로 표시된 비트는 장래 사용을 위해 보류되고 " -" 로 표시된 비트는 본 양호한 실시예에서는 구현되지 않지만 다른 실시예들에서 사용될 수 있다. 인터럽트는 좌측에서 우측으로 우선 순위가 매겨진다. 각각의 인터럽트 소스는, 인터럽트 인에이블 레지스터 INTEN(110)의 대응 인에이블(E) 비트에 " 1" 을 설정함으로써 개별적으로 인에이블될 수 있다. 인터럽트 플래그 레지스터 INTFLG(115)의 인터럽트 소스 비트는 우측에서 좌측으로 우선 순위의 내림차순으로, 즉, 항상 인에이블되는 애뮬레이션 인터럽트 ETRAP; XY 패치 인터럽트; 작업 인터럽트; 패킷 전송 비지 인터럽트(packet transfer busy interrupt) PT13; 패킷 전송 에러 인터럽트 PTERROR; 패킷 전송 성공 인터럽트 PTEND; 마스터 프로세서(60) 메시지 인터럽트 MPMSG; 디지탈 영상/그래픽 프로세서(71) 메시지 인터럽트 DIGPOMSG; 디지탈 영상/그래픽 프로세서(72) 메시지 인터럽트 DIGPIMSG; 디지탈 영상/그래픽 프로세서(73) 메시지 인터럽트 DIGP2MSG; 디지탈 영상/그래픽 프로세서(74) 메시지 인터럽트 DIGP3MSG 순으로 이루어진다. 비트 31-28은 8개의 디지탈 영상/그래픽 프로세서를 포함하는 멀티프로세서 집적 회로(100)의 구현에서의 4개의 부가적인 디지탈 영상 그래픽 프로세서로부터 메시지 인터럽트용으로 지정된다.

인터럽트 인에이블 레지스터 INTEN(110)의 " W" 비트(비트 0)는 인터럽트 플래그 레지스터 INTFLG(115)로의 기입을 제어한다. 이 비트는 통상적으로 애뮬레이션 인터럽트가 인에이블되는지의 여부를 제어한다. 양호한 실시예에서 애뮬레이션 인터럽트가 디스에이블될 수 없기 때문에, 인터럽트 인에이블 레지스터 INTEN(110)에 이 인터럽트용의 인에이블 비트가 있을 필요는 없다. 인터럽트 인에이블 레지스터 INTEN(110)의 비트 0은 인터럽트 플래그 레지스터 INTFLG(115)의 수행을 수정한다. 인터럽트 인에이블 레지스터 INTEN(110)의 " W" 비트가 " 1" 일 때, 인터럽트 플래그 레지스터 INTFLG(115)로의 소프트웨어 기입은 비트 " 1" 로만 설정될 수 있다. 이러한 상황하에서, 인터럽트 플래그 레지스터 INTFLG(115)의 임의의 비트에 " 0" 을 기입하고자 하는 시도는 효과를 갖지 못한다. 이 " w" 비트가 " 0" 일 때, 인터럽트 플래그 레지스터 INTFLG(115)의 임의의 비트에 " 1" 을 기입하는 것은 그 비트를 " 0" 으로 클리어하게 된다. 인터럽트 플래그 레지스터 INTFLG(115)의 임의의 비트에 " 0" 을 기입하고자 하는 시도는 효과를 갖지 못한다. 이것은 인터럽트 플래그 레지스터 INTFLG(115) 내의 개개의 인터럽트 플래그가 다른 것들의 상태에 영향을 주지 않고 클리어되게 한다. 본 실시예에서는, 이들 플래그가 하드웨어에 의해 클리어되지 않기 때문에, 각각의 인터럽트 서비스 루틴은 그것의 대응하는 인터럽트 플래그를 적절히 클리어시킨다. 이에 대한 한가지 예외로서, 에뮬레이션 인터럽트 ETRAP은 항상 인에이블되어 있기 때문에 하드웨어에 의해 클리어된다. 특정한 인터럽트 소스가 소프트웨어 기입 동작이 이것을 클리어하려고 하는 것과 동시에 인터럽트 플래그 레지스터 INTFLG(115) 내에 비트를 설정하고자 시도하는 경우, 논리는 비트가 설정되게 한다.

ETRAP 인터럽트 플래그(인터럽트 플래그 레지스터 INTFLG(115)의 비트 0)는 분석 로직 또는 ETRAP 명령으로부

터 설정된다. 이 인터럽트는 디스에이블될 수 없기 때문에 정상적으로는 즉시 서비스되지 않으나, 인터럽트 서비스는 크로스바(50)를 통한 메모리 경합과 같은 파이프라인 스톨 상황(pipeline stall condition)이 해결될 때까지 대기한다. ENTRAP 인터럽트 플래그는, 인터럽트가 서비스될 때 하드웨어에 의해 클리어되는 인터럽트 플래그 레지스터 INTFLG(115) 내의 유일한 인터럽트 비트이다.

XY PATCH 인터럽트 플래그(인터럽트 플래그 레지스터 INTFLG(115)의 비트 11)는, 글로벌 어드레스부(610) 및 로컬 어드레스부(620)가 XY 어드레싱을 수행하도록 조합한 것을 이용할 때 소정의 상황 하에서 설정된다. XY 패치 어드레싱은 소정의 상황에서 인터럽트를 발생시킬 수 있다. XY 패치 어드레싱을 요청하는 명령 워드는, 이러한 인터럽트가 발생될 수 있는지의 여부와 허용된 인터럽트가 지정된 패치 내부 또는 외부의 어드레스에 대해 이루어지는 지의 여부를 표시한다.

TASK 인터럽트 플래그(인터럽트 플래그 레지스터 INTFLG(115)의 비트 14)는 마스터 프로세서(60)로부터 커맨드 워드를 수신할 때 설정된다. 이 인터럽트는 디지탈 영상/그래픽 프로세서(71)가 그것의 TASK 인터럽트 벡터를 로드하게 한다. 이 인터럽트는 선택된 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)가 예를 들어 마스터 프로세서(70)의 제어 하에서 작업을 스위칭하게 한다.

패킷 전송 비지 인터럽트 플래그 PTB(인터럽트 플래그 레지스터 INTFLG 115)의 비트 17)은, 큐 활성 비트가 " 1" 일 때 소프트웨어가 통신 레지스터 COMM(120)의 패킷 전송 비트에 1을 기입하는 경우 설정된다. 이것은 패킷 전송이 이전의 전송이 종료하였다는 것을 체크하지 않고서 이루어지게 한 이전의 패킷 전송이 아직 큐 되어 있으면 이 인터럽트 플래그가 설정된다. 이것은 통신 레지스터 COMM(120)의 설명과 관련하여 이하 더 설명될 것이다.

패킷 전송 에러 인터럽트 플래그 PTERROR(인터럽트 플래그 레지스터 INTFLG(115)의 비트(18)는, 전송 제어기(80)가 디지탈 영상/그래픽 프로세서에 의해 제공된 패킷 전송을 수행하는 동안 오류 상태를 만나는 경우에 설정된다. 패킷 전송 종료 인터럽트 플래그 PTEND(인터럽트 플래그 레지스터 INTFLG 115의 비트 19)는, 디지탈 영상/그래픽 프로세서의 링크 리스트의 종료를 만날 때, 또는 전송 제어기(80)에 종료시 요청하는 디지탈 영상/그래픽 프로세서를 인터럽트하라고 지시하는 패킷 전송을 종료할 때, 전송 제어기(80)에 의해 설정된다.

마스터 프로세서 메시지 인터럽트 플래그 MPMSG(인터럽트 플래그 레지스터 INTFLG(115)의 비트 20)은, 마스터 프로세서(60)가 영상/그래픽 프로세서에 메시지-인터럽트를 보낼 때 설정된다.

인터럽트 플래그 레지스터 INTFLG(115)의 비트 27-24는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)로부터의 메시지 인터럽트를 로그(log)한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)는 메시지를 자신에게 보낼 수 있고, 인터럽트 플래그 레지스터 INTFLG(115)의 대응하는 비트를 통해 자신을 인터럽트할 수 있다. 디지탈 영상/그래픽 프로세서 0 메시지 인터럽트 플래그 DIGPOMSG(인터럽트 플래그 레지스터 INTFLG(115)의 비트 24)는, 디지탈 영상/그래픽 프로세서(71)가 디지탈 영상/그래픽 프로세서에 메시지 인터럽트를 보낼 때 설정된다. 유사한 방식으로, 디지탈 영상/그래픽 프로세서 1 메시지 인터럽트 플래그 DIGP/MSG(인터럽트 플래그 레지스터 iNTFLG(115)의 비트 25)는, 디지탈 영상/그래픽 프로세서(72)가 메시지 인터럽트를 보낼 때 설정되고, 디지탈 영상/그래픽 프로세서 2 메시지 인터럽트 플래그 DIGP2MSG(인터럽트 플래그 레지스터 INTFLG(115)의 비트 26)은, 디지탈 영상/그래픽 프로세서(73)가 메시지 인터럽트를 보낼 때 설정되며, 디지탈 영상/그래픽 프로세서 3 메시지 인터럽트 플래그 DIGP3MSG(인터럽트 플래그 레지스터 INTFLG(115)의 비트 27)은 디지탈 영상/그래픽 프로세서(74)가 메시지 인터럽트를 보낼 때 설정된다. 앞서 설명한 바와 같이, 인터럽트 플래그 레지스터 INTFLG(115)의 비트 31-28은 8개의 디지탈 영상/그래픽 프로세서를 포함하는 멀티 프로세서 집적 회로(100)의 구현에서의 4개의 부가적인 디지탈 영상/그래픽 프로세서로부터의 메시지 인터럽트용으로 보류된다.

인에이블된 인터럽트가 발생할 때, 소형 상태 머신(small state machine)일 수 있는 인터럽트 의사-명령(pseudo-instruction)부는 명령 레지스터-어드레스 단계에서 다음의 의사 명령 세트를 파이프라인으로 주입한다.

*(A14 - =16) = SR

*(A14 + 12) = PC

BR = *vectadd; vectadd의 2개의 LS 비트 = "11', S, G 및 L을 로드

*(A14 + 8) = IPA

*(A14 + 4) = IPE

이들 의사 명령은 각각 PS1, PS2, PS3, PS4 및 PS5로서 참조된다. 서브루틴 IPRS로부터의 명령 포인터-리턴(instruction pointer-return)은 이 시퀀스에 의해 세이브되지 않는다. 인터럽트 서비스 루틴이 어떤 브랜치를 수행한다면, 서브루틴 IPRS로부터의 명령 포인터-리턴은 인터럽트 서비스 루틴에 의해 푸시된 후, 복귀 이전에 복원되어야 한다. 벡터 페치는 전체 프로그램 카운터 PC(701)의 로드이고 서브루틴 IPRS로부터의 명령 포인터-리턴은 보호된다는 점에 유의한다. 이것은 프로그램 카운터 PC(701)의 S, G 및 L 비트가 로드되게 하기 때문에, 모든 인터럽트 벡터의 3개의 최하위 비트는 "0" 으로 된다. 이 설명의 한가지 예외는 재설정 이후에 페치된 작업 벡터는 루핑을 디스에이블하기 위해 "L" 비트(프로그램 카운터 PC(701))의 "L" 비트 01)를 가져야만 한다는 것이다.

인터럽트 플래그 레지스터 INTFLG(115)로 대표되는 어떤 인터럽트에 대한 인터럽트 서비스 루틴의 시작점 각각의 어드레스를 디지탈 영상/그래픽 프로세서 인터럽트 벡터라고 한다. 이들 어드레스는 소프트웨어에 의해 발생되고, 표 1에 나타낸 고정 어드레스에서 인터럽트된 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)각각에 대응하는 파라메타 메모리(25, 30, 35 및 40)에 데이터로서 로드된다. 인터럽트 의사-명령 PS3은 대응하는 파라메타 메모리(25, 30, 35 또는 40) 내의 표시된 어드레스에 저장된 32비트 어드레스를 취하여 이것을 프로그램 카운터 PC(701)에 저장시킨다. 인터럽트 의사-명령부(770)는 인터럽트 인에이블 레지스터를 통해 인에이블된 최우선 순위 인터럽트에 기초하여 대응 파라메타 메모리에 대한 어드레스를 계산한다. 인터럽트 의사-명령부(770)는, 각각의 디지탈 영상/그래픽 프로세서에 대한 특정 어드레스를 발생시키기 위해 통신 레지스터 COMM(120)으로부터의 디지탈 영상/그래픽 프로세서 번호를 포함시키도록 동작한다. 인터럽트 의사-명령(PS4 및 PS5)은 이 브랜치를 인터럽트 서비스 루틴에 이어주는 지연 슬롯이다.

INTF6LG 비트	인터럽트 이름	어드레스
31	DIGP7 메시지를 위해 보류됨	0100#1FC
30	DIGP6 메시지를 위해 보류됨	0100#1F8
29	DIGP5 메시지를 위해 보류됨	0100#1F4
28	DIGP4 메시지를 위해 보류됨	0100#1F0
27	DIGP3 메시지	0100#1EC
26	DIGP2 메시지	0100#1E8
25	DIGP1 메시지	0100#1E4
24	DIGP0 메시지	0100#1E0
23	예비	0100#1DC
22	예비	0100#1D8
21	예비	0100#1D4
20	마스터 프로세서 메시지	0100#1D0
19	패킷 전송 성공	0100#1CC
18	패킷 전송 오류	0100#1C8
17	패킷 전송 사용 중(busy)	0100#1C4
16	예비	0100#1C0
15	예비	0100#1BC
14	TASK 인터럽트	0100#1B8
13	예비	0100#1B4
12	예비	0100#1B0
11	XY 패칭	0100#1AC
10	보류	0100#1A8
9	보류	0100#1A4
8	보류	0100#1A0
7	보류	0100#19C
6	보류	0100#198
5	보류	0100#194
4	보류	0100#190
3	보류	0100#18C
2	예비	0100#188
1	예비	0100#184
0	에뮬레이션	0100#180

각각의 어드레스에서 " #" 은 통신 레지스터 COMM(120)으로부터 얻어진 디지탈 영상/그래픽 프로세서 번호로 대체된다.

인터럽트 서비스루틴의 최종 4개의 명령은 다음의 (32비트 데이터, 시프트되지 않은 인덱스) 연산을 포함하여야 한다.

SR = *(A14 ++ = 4)

BR = *(A14 ++ = 7)

BR = *(A14 ++ = 5)

BR = *(A14 ++ = 5)

이들 명령은 RETI1, RETI2, RETI3 및 RETI4로 각각 참조된다. 다른 동작들도 원한다면 이들과 병행하여 코딩될 수 있지만, 이러한 동작들 중 어느 것도 상태 레지스터(211)를 수정해서는 안된다.

새로운 작업이 디지탈 영상/그래픽 프로세서 상에서 실행되면 인터럽트 상태가 세이브될 수 있으며, 새로운 작업을 완료한 후에는 원래 상태로 복원될 수 있다. 인터럽트 인에이블 레지스터 INTEN(10)에 의해 제어되는 기입 모드는, 이것이 세이브 또는 복원 동작동안 어떠한 인터럽트도 미싱하지 않고 수행되게 한다. 이것은 다음과 같은 명령 시퀀스에 의해 달성될 수 있다. 첫째 DINT 명령을 통해 인터럽트를 디스에이블한다. 다음에는 인터럽트 인에이블 레지스터 INTEN(110) 및 인터럽트 플래그 레지스터 INTFLG(115)를 세이브한다. 인터럽트 인에이블 레지스터 INTEN(110)의 " W" 비트(비트 0)를 " 0" 으로 설정하고 Hex " FFFFFFFF" 를 인터럽트 플래그 레지스터 INTFLG(115)에 기입한다. 인에이블링 인터럽트를 포함할 수 있는 새로운 작업을 실행한다. 새로운 작업의 종료 후에 원래의 작업을 리커버한다. 첫째, DINT 명령을 통해 인터럽트를 디스에이블한다. 인터럽트 인에이블 레지스터 INTEN(110)의 " W" 비트를 " 1" 로 설정한다. 메모리로부터 인터럽트 플래그 레지스터 INTFLG(115)의 상태를 복원한다. 다음에, 메모리로부터 인터럽트 인에이블 레지스터 INTFLG(110)의 상태를 복원한다. 최종적으로, EINT 명령을 통해 인터럽트를 인에이블한다.

각각의 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 다른 디지탈 영상/그래픽 프로세서와 마스터 프로세서(60)에 명령 워드를 전송할 수 있다. 레지스터는 레지스터 A15 (글로벌 어드레스부의 제로값 레지스터)를 데스티네이션으로 하는 레지스터 대 레지스터 이동(register to register move)은, 지정된 프로세서로의 명령 워드 전송을 개시한다. 이러한 레지스터 대 레지스터 전송은 아래에 설명되는 바와 같이, 데이터부(110)의 동작 및 로컬 데이터 포트(144)를 통한 액세스에 의해 단일 명령으로 조합될 수 있다. 이러한 명령 워드는 특별 명령 워드 신호가 수반되는 글로벌 데이터 포트(148)를 통해 크로스바(50)로 전송된다. 이것은 마스터 프로세서(60) 및 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)가 멀티프로세서 집적 회로(100)의 다른 프로세서들과 통신하게 한다.

제4도는 이들 명령 워드의 필드 정의를 개략적으로 도시한다. 양호한 실시예에서, 커맨드 워드는 글로벌 데이터 포트(148)를 통해 전송된 데이터와 동일한 32 비트 길이를 갖는다. 각각의 커맨드 워드의 최하위 비트는, 그 커맨드 워드가 어드레스되는 하나 이상의 프로세서와 다른 회로를 정해준다. 각각의 수신 회로는, 이 비트들이 그 커맨드 워드가 그 회로에 지정된다는 것을 표시한 경우에만 수신된 커맨드 워드에 응답한다. 각각의 커맨드 워드의 비트 3-0은 각각 디지탈 영상/그래픽 프로세서(74, 73, 72 및 71)를 지정한다. 비트 7-4는 양호한 실시예에서 사용되지 않지만, 8개의 디지탈 영상/그래픽 프로세서를 갖는 멀티프로세서 집적 회로(100)에 사용하기 위해 보류된다. 비트 8은 명령 워드가 마스터 프로세서(60)에 어드레스된다는 것을 표시한다. 비트 9는 커맨드 워드가 전송 제어기(80)에 지정된다는 것을 표시한다. 비트 10은 커맨드 워드가 프레임 제어기(90)에 지정된다는 것을 표시한다. 모든 회로가 모든 커맨드 워드를 모든 다른 회로에 보내도록 허용되지는 않는다는 점에 유의한다. 예를 들어, 시스템 레벨 커맨드 워드는 디지탈 영상/그래픽 프로세서로부터 다른 디지탈 영상/그래픽 프로세서 또는 마스터 프로세서(60)에 보내질 수 없다. 마스터 프로세서(60)만이 커맨드 워드를 전송 제어기(80) 또는 프레임 제어기(90)에 보낼 수 있다. 어느 회로가 어느 커맨드 워드를 어느 다른 회로에 보낼 수 있는지에 대한 제한은 각각의 커맨드 워드 필드의 설명과 관련하여 아래에 설명될 것이다.

커맨드 워드의 " R" 비트(비트 31)는 재설정 비트이다. 마스터 프로세서(60)가 이 커맨드 워드를 임의의 디지탈 영상/그래픽 프로세서로 발생할 수도 있고, 디지탈 영상/그래픽 프로세서가 이 커맨드 워드를 자신에게 발행할 수도 있다. 양호한 실시예에서, 디지탈 영상/그래픽 프로세서는 다른 디지탈 영상/그래픽 프로세서를 재설정할 수 없다. 재설정 시퀀스에 대한 다음의 설명에서, 어드레스 내의 각 디지트 " #" 는 커맨드 레지스터 COMM(120)의 비트 1-0 내에 저장된 디지탈 영상/그래픽 프로세서 번호로 대체되어야 함에 유의한다. 지정된 디지탈 영상/그래픽 프로세서는 재설정 커맨드 워드를 수신하면, 먼저 자신의 정지 래치(halt latch)를 설정하고 재설정 요청 신호를 전송 제어기(80)에 보낸다. 전송 제어기(80)는 재설정 승인 신호를 디지탈 영상/그래픽 프로세서에 보낸다. 재설정 디지탈 영상/그래픽 프로세서의 재설정은 이 재설정 승인 신호를 전송 프로세서(80)로부터 수신할 때까지 더 이상의 동작을 수행하지 않는다. 재설정 승인 신호를 수신하면, 디지탈 영상/그래픽 프로세서는 다음의 동작 시퀀스를 시작한다. 즉, 아직 설정되지 않은 정지 래치를 설정한다; 통신 레지스터 COMM(120)의 " F" , " P" , " Q" 및 " S" 비트를 " 0" 으로 클리어한다 (이들 비트의 사용은 이하 설명됨); 어드레스부(20)에 의해 보류 중인 임의의 메모리 액세스를 클리어한다; 임의의 명령 캐시 서비스 요청을 재설정한다; 다음과 같은 명령

BR = [u. ncvz] A14 < < 1

A14 = Hex " 0100#7FO"

을 명령 레지스터-실행 스테이지 IRE(752)에 로딩하는데, 좌측으로 1비트 시프트된 스택 포인터 A14의 내용을 변경이 방지된 네가티브, 캐리, 오버플로우 및 제조 상태 비트와 스택 포인터 A14의 로드와 병렬로 스택 포인터 A14를 재설정하도록 설정된 " R" 비트로 프로그램 카운터 PC(701)에 무조건적으로 로딩한다; 다음과 같은 명령

*(PBA + Hex " FC") = PC

을 명령 레지스터-어드레스 스테이지 IRA로 로딩한다 -이 명령은 어드레스 PBA 및 Hex " FC" 의 합으로 표시된 어드레스에 프로그램 카운터 PC(701)의 내용을 저장함-; 인터럽트 의사-명령부(70)를 다음의 로드 인터럽트 의사-명령 PS3으로 설정한다; 작업 인터럽트를 표시하는 인터럽트 플래그 레지스터 INTFLG(115)의 비트 14를 설정한다; 인터럽트 플래그 레지스터 INTFLG(115)의 비트 0을 클리어시켜 에뮬레이터 트랩 인터럽트 ETRAP를 클리어한다; 루프 제어 레지스터 LCTL의 비트 11, 7 및 3을 클리어하여 모든 3개의 루프를 디스에이블한다.

디지탈 영상/그래픽 프로세서에 의한 실행은 마스터 프로세서(60)가 비정지 커맨드 워드(unhalt command word)를 전송할 때 시작한다. 일단 실행이 시작되면, 디지탈 영상/그래픽 프로세서는, 어드레스 Hex " 0100#7FC" 로 프로그램 카운터 PC(701) 내에 저장된 어드레스를 세이브하고 -이것은 좌측으로 한자리 시프트된 스택 포인터 A14의 이전 내용과 프로그램 카운터 PC(701)의 제어 비트(비트 2-0)의 현재값을 세이브함-; 어드레스 Hex " 0100#7FO" 을 스택 포인터 A14에 로드하고; 작업 인터럽트 벡터로 프로그램 카운터 PC(701)을 로드하며 -제어 비트 2-0는 " 000" 임-; 어드레스 Hex " 0100#7FO" 에 제어 비트 2-0을 포함하는 명령 레지스터-어드레스 스테이지 IPA의 내용을 저장하고; 어드레스 Hex " 0100#7FO" 에 제어 비트 2-0을 포함하는 명령 레지스터-실행 스테이지 IPE의 내용을 저장하며; 작업 인터럽트에 의해 주어진 어드레스에서 프로그램 실행을 시작한다. 재설정 이후의 스택 상태는 표 2에 표시되어 있다.

어드레스	내용
Hex "0100#7FC"	좌측으로 한자리 시프트되는 재설정 이전의 스택 포인터 레지스터 A14
Hex "0100#7F8"	재설정 이전의 명령 레지스터-어드레스 스테이지 IRA
Hex "0100#7F4"	재설정 이전의 명령 레지스터-실행 스테이지 IRE

명령 레지스터-어드레스 스테이지 IRA 및 명령 레지스터 실행 스테이지 IRE의 이전 상태는 제어 비트 2-0을 포함한다. 스택 포인터 A14는 이제 어드레서 Hex "0100#7FO" 을 포함한다.

커맨드 워드의 "H" 비트(비트 30)는 정지 비트이다. 마스터 프로세서(60)가 이 커맨드 워드를 임의의 디지탈 영상/그래픽 프로세서에 발행할 수도 있고, 디지탈 영상/그래픽 프로세서가 이 커맨드 워드를 자신에 발행할 수도 있다. 양호한 실시예에서, 디지탈 영상/그래픽 프로세서는 다른 디지탈 영상/그래픽 프로세서를 정지시킬 수 없다. 지정된 디지탈 영상/그래픽 프로세서가 이 커맨드 워드를 수신하면, 디지탈 영상/그래픽 프로세서는 정지 레지스터를 설정하고 파이프라인을 스톨한다. 그 후 디지털 영상/그래픽 프로세서는 무한 크로스바 메모리 경합에 있는 것처럼 행동한다. 아무것도 재설정되지 않으며, 어떠한 인터럽트도 발생 또는 인식되지 않는다. 디지탈 영상/그래픽 프로세서가 커맨드 워드를 보냄으로써 그 자체를 정지시킬 때, 정지 커맨드 워드를 보내는 명령 이후의 2개의 명령은 그것의 명령 파이프라인 내에 있음에 유의한다. 정지 커맨드 워드를 발행하는 명령 이후의 제1 명령의 어드레스 파이프라인 스테이지는, 명령 파이프라인의 성질로 인해 그 어드레스 스테이지를 이미 실행했을 것이다. 이러한 정지 상태는 마스터 프로세서(60)로부터 비정지 커맨드 워드를 수신함으로써만 반전(reverse)될 수 있다.

정지 조건은 그 상태가 변경되지 않기 때문에 디지탈 영상/그래픽 프로세서 내의 전력 소비를 감소시킨다. 디지탈 영상/그래픽 프로세서가 이 모드에 있는 동안 클럭을 중지시키면, 전력이 더 감소된다.

커맨드 워드의 "U" 비트(비트 29)는 비정지 비트이다. 이 커맨드 워드는 마스터 프로세서(60)에 의해서만 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 중 하나 이상에 발행될 수 있다. 비정지 커맨드 워드는 데스티네이션 디지탈 영상/그래픽 프로세서의 정지 래치를 클리어시킨다. 그러면, 디지탈 영상/그래픽 프로세서는 아무것도 발생하지 않은 것처럼 정지 이후의 코드 실행을 재개한다. 이것은 하드웨어 또는 커맨드 워드 재설정 이후에 디지탈 영상/그래픽 프로세서를 시작하기 위한 양호한 방식이다. 비정지 커맨드 워드의 실행시에, 데스티네이션 디지탈 영상/그래픽 프로세서는 작업 인터럽트 벡터에 의해 주어진 어드레스에서 코드 실행을 시작한다. "U" 비트는 단일 커맨드 워드의 "H" 비트보다 우선 순위를 갖는다. 그러므로, "H" 비트 및 "U" 비트 세트를 갖는 단일 커맨드 워드의 수신은 비정지 커맨드를 실행시킨다. 마스터 프로세서(60)로부터의 비정지 커맨드 및 디지탈 영상/그래픽 프로세서 자체에 의해 전송된 정지 커맨드 워드를 동시에 수신하는 경우, 마스터 프로세서(60)의 비정지 커맨드 워드에 우선 순위가 부여된다는 점에 유의한다. "R" 비트는 "U" 비트보다 우선 순위를 갖는다. 그러므로 "R" 비트 및 "U" 비트 세트를 갖는 마스터 프로세서(60)로부터의 단일 커맨드 워드의 수신은 디지탈 영상/그래픽 프로세서가 정지 조건으로 재설정되게 한다.

커맨드 워드의 "I" 비트(비트 28)는 명령 캐시 플러시(flush) 비트이다. 마스터 프로세서(60)는 이 커맨드 워드를 임의의 영상/그래픽 프로세서에 발생시킬 수 있고, 또는 디지탈 영상/그래픽 프로세서는 이러한 커맨드를 그 자신에 발생시킬 수 있다. 양호한 실시예에서 디지탈 영상/그래픽 프로세서는 다른 디지탈 영상/그래픽 프로세서에 의해 명령 캐시 플러시를 명령할 수 있다. 이 커맨드 워드를 수신하는 지정된 디지탈 영상/그래픽 프로세서는 그 명령 캐시를 플러시한다. 명령 캐시 플러시는 캐시 태그 값 필드를 태그 레지스터의 자체 수로 설정시키고, 모든 이들의 현재 비트를 클리어

하며, LRU 비트를 태그 레지스터의 자체 수로 설정시킨다.

커맨드 워드의 " D" 비트(비트 27)는 데이터 캐시 플러시를 표시한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 데이터 캐시를 이용하지 않으므로, 이 커맨드 워드는 디지탈 영상/그래픽 프로세서에 가해지지 않고 그들에 의해 무시된다. 마스터 프로세서(60)는 그 데이터 캐시 메모리(13 및 14)를 플러시하기 위해 이 커맨드 워드를 자신에게 보낼 수 있다.

커맨드 워드의 " K" 비트(비트 14)는 작업 인터럽트를 표시한다. 마스터 프로세서(60)는 이 커맨드 워드를 임의의 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)에 보낼 수 있지만, 디지탈 영상 그래픽 프로세서는 이 커맨드 워드를 다른 디지탈 영상/그래픽 프로세서 또는 마스터 프로세서(60)에 보낼 수 없다. 작업 커맨드 워드의 수신 시에, 커맨드 워드로 지정된 임의의 디지탈 영상/그래픽 프로세서는, 인터럽트 인에이블 레지스터 INTEN(110)에 의해 인에이블된 경우 작업 인터럽트를 취한다.

커맨드 워드의 " G" 비트(비트 13)는 메시지 인터럽트를 표시한다. 임의의 디지탈 영상/그래픽 프로세서는 이 메시지 인터럽트를 임의의 다른 디지탈 영상/그래픽 프로세서 또는 마스터 프로세서(60)에 보낼 수 있다. 이러한 커맨드 워드로 지정된 임의의 디지탈 영상/그래픽 프로세서는 그것의 메시지 인터럽트 플래그를 설정하고, 메시지 인터럽트가 인터럽트 인에이블 레지스터 INTEN(110)의 비트 20을 통해 인에이블된 경우 메시지 인터럽트를 취한다. 양호한 실시예에서, 이 커맨드 워드는 전송 제어기(80)에 보내지지 않는다.

디지탈 영상/그래픽 프로세서가 그 자신에 커맨드 워드를 발행하여 " H" 비트를 통해 그 자신을 정지하거나 " I" 비트를 통해 그 명령 캐시를 플러시하는 경우, 이 커맨드 워드는 커맨드를 실행하기 위해, 대응하는 디지탈 영상/그래픽 프로세서 지정 비트를 설정시켜야 한다. 이것은 일관성을 위한 것이고, 앞으로의 커맨드 워드 기능 확장을 허용하기 위한 것이다.

제5도는 통신 레지스터 COMM의 필드 정의를 개략적으로 도시한다. " F" , " S" , " Q" 및 " P" 비트(비트 31-28)는 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74) 및 전송 제어기(80)로부터의 패킷 전송의 통신에 이용된다. " F" 및 " S" 비트는 정상적인 판독/기입 비트이다. " P" 비트는 " S" 비트가 " 0" 인 경우 또는 동시에 " 0" 으로 클리어되는 경우에만 기입될 수 있다. " Q" 비트는 판독 전용이다. 패킷 전송은 디지탈 제어기(80)에 의한 데이터 이동을 위해 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)에 의해 요청된다. 이들 데이터 이동은 멀티프로세서 집적 회로(100) 내부에 있는 메모리(11-14 및 21-40)만을 포함할 수도 있고 내부 메모리 및 외부 메모리 둘 다를 포함할 수도 있다. 패킷 전송은 링크-리스트 구조로서 저장되어, 각각의 디지탈 영상/그래픽 프로세서에 대해 한번에 하나의 패킷 전송만이 활성화될 수 있다. 요청하는 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)에 대응하는 파라메타 메모리(25, 30, 35 또는 40) 내의 전용 어드레스에서의 링크-리스트 포인터는 액티브 링크-리스트의 시작을 지시한다, 링크-리스트내의 각각의 엔트리는 다음 리스트 엔트리에 대한 포인터를 포함한다.

패킷 전송을 초기화하는 것은 다음의 단계를 포함한다. 먼저, 디지탈 영상/그래픽 프로세서는 대응 파라메타 메모리 내로 원하는 패킷 전송 파라메타를 설정한다. 다음에, 디지탈 영상/그래픽 프로세서는 대응 파라메타 메모리 내의 선정된 어드레스 Hex " 0100#0FC" 에서의 링크-리스트의 제1 링크의 어드레스를 저장하는데, 여기서 " #" 은 디지탈 영상/그래픽 프로세서의 번호로 대체된다. 통신 레지스터 COMM(120)의 " P" 비트(비트 28)를 " 1" 로 설정하는 것은 패킷 전송의 전송 제어기(80)를 경보시킨다. 디지탈 영상/그래픽 프로세서는 " F" 비트9비트 31)을 " 1" 로 설정함으로써 높은 우선 순위를 요청하거나 " F" 비트를 " 0" 으로 클리어시킴으로써 낮은 우선 순위를 요청할 수 있다.

전송 제어기(80)는 " P" 비트가 설정될 때를 인식하여 " F" 비트의 상태에 기초하여 패킷 전송에 우선 순위를 할당한다. 전송 제어기(80)는 " F" 비트를 클리어하고 " Q" 비트를 설정하여 패킷 전송이 큐 상태에 있다는 것을 표시한다. 다음에 전송 제어기(80)는 대응 파라메타 메모리 내의 선정된 어드레스 Hex " 0100#OFC" 에 액세스하고, 링크 리스트에 기초하여 패킷 전송을 서비스한다. 패킷 전송의 종료시에, 전송 제어기(80)는 " Q" 비트를 " 0" 으로 클리어하여 큐가 더 이상 액티브하지 않다는 것을 표시한다. 디지탈 영상/그래픽 프로세서는 패킷 전송이 완료됐다는 것을 표시하기 위해 이 비트를 주기적으로 판독할 수 있다. 다르게는, 패킷 전송 자체가, 전송 제어기(80)에 패킷 전송의 완료시 요청하는 디지탈 영상/그래픽 프로세서를 인터럽트하라고 명령할 수 있다. 이 경우에, 전송 제어기(80)는 인터럽트 플래그 레지스터 INTFLG(115)에서, 패킷 전송 종료 인터럽트 비트 PTEND를 비트 19로 설정함으로써 디지탈 영상/그래픽 프로세서에 인터럽트를 보낸다. 전송 제어기(80)가 패킷 전송을 서비스하는데 있어서 에러가 발생될 경우, 전송 제어기(80)는 인터럽트 플래그 레지스터 INTFLG(115) 내의 패킷 전송 에러 인터럽트 비트 PTERROR을 비트 18로 설정함으로써 디지탈 영상/그래픽 프로세서에 인터럽트를 보낸다. 디지탈 영상/그래픽 프로세서는 표 1에 명시된 위치에 저장된 적절한 인터럽트 벡터와 적절한 인터럽트 서비스 루틴을 갖는다.

디지탈 영상/그래픽 프로세서는, 전송 제어기(80)가 이전의 요청을 서비스하는 동안에 다른 패킷을 요청할 수 있다. 이 경우에 디지탈 영상/그래픽 프로세서는 " P" 비트를 " 1" 로 설정하며, " Q" 비트는 " 1" 이다. 이 경우, 전송 제어기(80)는 인터럽트 플래그 레지스터 INTFLG(115)를 비트 17로 설정함으로써 디지탈 영상/그래픽 프로세서에 패킷 전송 비지 인터럽트 PTB를 보낸다. 다음으로, 전송 제어기(80)는 " P" 비트를 " 0" 으로 클리어시킨다. 요청하는 디지탈 영상/그래픽 프로세서의 인터럽트 서비스 루틴은 제1 패킷 전송이 큐 상태에 있는 동안 제2 패킷 전송을 중지할 수 있고, 패킷 전송을 취소하거나 소정의 다른 정정 동작을 취할 수 있다. 이러한 특징은 디지탈 영상/그래픽 프로세서가 통신 레지스터 COMM(120)의 " Q" 비트를 먼저 체크하지 않고서 패킷 전송을 제공할 수 있게 한다.

디지탈 영상/그래픽 프로세서는 " S" 비트를 " 1" 로 설정함으로써 패킷 전송의 서비스를 중지할 수 있다. 전송 제어기(80)는 " S" 비트가 " 1" 일 때를 검출한다. 패킷 전송이 큐 상태에 있는 동안 이것이 발생하면, 전송 제어기(80)는 " Q" 비트를 " P" 비트로 카피하고 " Q" 비트를 클리어한다. 이것은 일반적으로 " P" 비트를 " 1" 로 설정할 것이다. 다음에, 요청하는 디지탈 영상/그래픽 프로세서 내의 소프트웨어는 " S" 및 " P" 비트의 상태를 변경시킬 수 있다. 전송 제어기(80)은 중지된 패킷 전송의 링크-리스트 내의 그 위치를 메모리 내에 보유한다. 전송 제어기(80)이 " S" 비트가 " 0" 이고 " P" 비트가 동시에 " 1" 이라는 것을 결정하면, 중지된 패킷 전송은 재개된다.

통신 레지스터 COMM(120)의 " 동기 비트" 필드 (비트 15-8)는 동기화된 다중 명령인 다중 데이터 모드에서 사용된다. 이것은 동기화된 다중 명령인 다중 데이터 모드를 인에이블하는 록크 명령 LCK 및 이 모드를 디스에이블하는 언록크 명령 UNLCK에 의해 한정된 임의의 명령에 대해 동작한다. 비트 11-8은 명령 페칭이 디지탈 영상/그래픽 프로세서(74, 73, 72 및 71)와 동기화되는지를 각각 표시한다. 이들 비트들 중 임의의 비트 내의 " 1" 은, 대응하는 디지탈 영상/그래픽 프로세서가 이전의 명령의 실행을 종료하였다는 것을 표시할 때까지 디지탈 영상/그래픽 프로세서가 명령 페치를 지연한 것을 표시한다. 이 디지탈 영상/그래픽 프로세서가 동기화될 다른 디지탈 영상/그래픽 프로세서는 그들의 통신 레지스터 COMM(120) 내의 대응 비트를 유사하게 설정할 것이다. 디지탈 영상/그래픽 프로세서가 동기화된 다중 명령인 다중 데이터 모드에 있을 때 자신에 대응하는 " 동기 비트" 가 설정될 필요는 없지만, 이것은 해를 주지 않는다. 비트 15-12는 8개의 디지탈 영상/그래픽 프로세서로의 가능한 확장을 위해 보류된다는 점에 유의한다.

통신 레지스터 COMM(120)의 " DIGP#" 필드 (비트 2-0)는 멀티프로세서 집적회로(100) 상의 각각의 특정한 디지탈 영상/그래픽 프로세서에 특정된다. 이들 비트는 판독 전용이며, 이들 비트에 기입하고자 하는 어떠한 시도도 실패한다. 이것은 동일하지 않은 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 부분만이다. 비트 1-0는 표 3에 표시된 특정한 디지탈 영상/그래픽 프로세서를 식별하는 2개의 비트 코드와 고정 배선(hardwire)된다.

COMM 필드		병렬 프로세서
1	0	
0	0	DIGP0 (71)
0	1	DIGP1 (72)
1	0	DIGP2 (73)
1	1	DIGP3 (74)

통신 레지스터 COMM(120)의 비트 2는 8개의 디지탈 영상/그래픽 프로세서를 갖는 멀티프로세서 집적 회로(100)에 사용하기 위해 보류된다. 본 실시예에서, 이 비트는 모든 4개의 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)에 대해 " 0" 으로 고정 배선된다.

통신 레지스터 COMM(120)의 이 부분은 특정한 디지탈 영상/그래픽 프로세서를 식별하는 기능을 한다. 디지탈 영상/그래픽 프로세서의 식별 번호는 통신 레지스터 COMM(120)을 7(Hex " 0000007")과 AND 함으로써 추출될 수 있다. 예를 들어, 명령 " DO=COMM87" 이 이것을 행한다. 이 명령은 통신 레지스터 COMM(120)의 비트 2-0 내의 데이터만을 복귀시킨다. 이 명령은 8개의 디지탈 영상/그래픽 프로세서를 갖는 실시예에 적합하다. 각각의 디지탈 영상/그래픽 프로세서에 대응하는 데이터 메모리 및 파라메타 메모리의 어드레스는 그 디지탈 영상/그래픽 프로세서의 식별에 의존하기 때문에, 식별 번호는 소프트웨어가 이들 대응 메모리에 대한 어드레스를 계산할 수 있게 한다. 이러한 식별 번호를 사용하면, 프로그램을 실행하는 특정한 디지탈 영상/그래픽 프로세서에 독립적인 소프트웨어를 기입할 수 있게 된다.

멀티프로세서 집적 회로(100)는 리틀 엔디언 또는 빅 엔디언 데이터 포맷 중 어느 하나에서 동작할 수 있다. 다르게 명시되지 않는다면, 도면과 표는 빅 엔디언 포맷의 동작을 나타낸다. 내부 레지스터와 외부 데이터 버스에 대한 비트 넘버링은 항상 비트 0이 우측에 있는 리틀 엔디언 관례를 따를 것이다. 64 비트 더블 워드 내의 바이트들은, 리틀 엔디언 모드에서는 우측에서부터 어드레스되고, 빅 엔디언 모드에서는 좌측에서부터 어드레스된다. 항상 32 비트 동작을 이용하여 내부 레지스터에 액세스함으로써 혼동을 피할 수 있다. 다른 실시예에서, 보다 넓거나 보다 좁은 버스폭이 고려된다. 16비트의 정수배인 버스폭이 꽤 유용한 것으로 여겨진다.

전송 제어기(80)는 멀티프로세서 집적 회로(100)의 마스터 프로세서(60)와 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74), 및 비디오 RAM(5 및 6)과 같은 외부 메모리와 메모리(9) 사이의 인터페이스를 제공한다. 전송 제어기(80)는 프로세서에 의해 요청된 메모리 동작뿐만 아니라 몇가지 자체 메모리 동작을 수행한다. 이들은 아래에 설명될 것이다. 전송 제어기(80)는 메모리(11, 12, 13, 14, 15, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39 및 40)로의 액세스를 제공하는 크로스바(50) 모두와 64비트 버스 접속을 한다. 그러므로 전송 제어기(80)는 모든 명령 캐시, 데이터 메모리 및 파라메타 메모리 모두에 액세스할 수 있다. 두번째 64 비트 버스는 영상 시스템 버스에 접속하고 오프 칩(off-chip) 액세스를 제공한다.

전송 제어기(80)는 외부 호스트가 그것의 홀드/홀드-승인 메카니즘을 이용하여 영상 시스템 버스를 통해 외부 메모리에 액세스할 수 있게 한다. 전송 제어기(80)는 캐시 미스와 패킷 전송 요청을 서비스하고, 외부 DRAM/VRAM의 리프레시를 수행하며, VRAM 기초 디스플레이/캡쳐 버퍼를 업데이트 하기 위해 프레임 제어기(90)에 의해 요청된 직렬 레지스터 전송 사이클을 수행한다. 메모리(22, 23, 24, 27, 28, 29, 32, 33, 34, 37, 38 및 39)에 관련된 전송 제어기

(80)의 메모리 동작은, 통상적으로 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 또는 마스터 프로세서(60)로부터의 패킷 전송 요청에 응답하여 발생한다. 패킷 전송은 온-칩 및/또는 오프칩 메모리 사이에 데이터를 전송하는 매우 융통성있는 방법을 제공한다. 또한, 전송 제어기(80)는 직접 외부 액세스(DEA) 사이클을 사용하여 프로세서들에 데이터를 직접 제공할 수 있다. 직접 외부 액세스 사이클은 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)가 오프칩 메모리에 액세스할 수 있게 하고, 마스터 프로세서(60)가 외부 메모리에 액세스할 때 그것의 데이터 캐시 메모리를 바이-패스(by-pass)할 수 있게 한다. 전송 제어기(80)는 다양한 요청들에 우선 순위를 부여하고, 필요에 따라서는 패킷 전송 요청들 간에 외부 메모리 인터페이스를 시간-분배한다. 마스터 프로세서(60), 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74), 프레임 제어기(90) 및 호스트 처리 시스템(1)으로부터의 모든 요청은 전송 제어기(80)에 의해 평가되고 다음에 고정된 우선 순위화 방식에 기초하여 서비스된다. 동일한 우선 순위의 다중 요청이 보류 중일 때, 전송 제어기(80)는 라운드 로빈 방식(round robin basis)으로 이들을 서비스한다.

전송 제어기(80)는 프로세서들로부터의 여러가지 다른 형태의 요청을 취급하여야 한다. 최적 시스템 성능을 보증하기 위해, 이들 요청은 그들의 긴급성 및 중요성에 따라 우선 순위가 부여된다. 전송 제어기(80)는 이러한 상이한 우선 순위들 상에서 동작하기 때문에, 크로스바(50) 상에서의 그 자신의 우선 순위는 사이클 마다 변화할 수 있다.

제6도는 영상 시스템 버스 상의 전송 제어기(80)의 동작의 우선 순위를 도시한다. 이것은 계층(hierarchy)(200)으로서 표시된다. 다중 요청이 동일한 우선순위로 수신될 때, 전송 제어기(80)는 이들 사이에서 라운드 로빈할 것이다. 이것은 제6도에 도시되어 있다. 어떠한 프로세서도 그 패킷 전송에 대해 단 하나의 실제 우선 순위를 가질 수 있음에 유의한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 높고 낮은 우선 순위 패킷 전송으로 한정되고, 마스터 프로세서(60)는 긴급한 높고 낮은 우선 순위의 패킷 전송 요청을 제공할 수 있다.

최우선 순위(201)는 호스트 처리 시스템(1)(HREQ)에 의한 외부 버스 요청의 서비스이다. 전송 프로세서(80)는 호스트 요청 라인 상의 신호에 응답하여, 영상 시스템의 제어를 호스트 처리 시스템(1)에 넘겨준다.

다음으로 낮은 우선 순위(202)는 프레임 제어기(90)로부터의 메모리 요청의 서비스이다. 다음으로 낮은 우선 순위(203)는 긴급한 동적 랜덤 액세스 메모리(DRAM) 리프레시 요청의 서비스이다. 이후 더 설명되는 바와 같이, 이들 긴급한 DRAM 리프레시 요청은 DRAM 리프레시 요청 형태의 선정된 백 로그(backlog)가 형성될 때 일어난다.

다음의 우선 순위(204)는 마스터 프로세서(60) 명령 캐시 메모리(11 및 12), 테이터 캐시 메모리(12 및 13) 및 인터럽트가 디스에이블될 때 마스터 프로세서(60)의 긴급한 패킷 전송 요청의 서비스이다. 캐시에 관련한 전송 제어기(80)의 모든 동작은 각 프로세서 내에 포함된 연관 캐시 논리에 의해 자동적으로 요청된다. 요청된 데이터는 그것의 외부 메모리 위치로부터 요청하는 프로세서의 데이터 캐시 메모리 내의 적절한 서브-블럭으로 이동된다. 또한, 전송 제어기(80)는 마스터 프로세서(60)에 의해 요청될 때 마스터 프로세서(60)에 대한 데이터 캐시 메모리(13 및 14)의 더티(dirty) 서브 블럭을 세이브한다. 긴급한 패킷 전송 요청은 마스터 프로세서(60)에 의해서만 요청될 수 있고 패킷 전송 요청내의 특정 비트의 설정을 포함한다. 이것은 이하 더 설명될 것이다. 마스터 프로세서(60)가 인터럽트를 인에이블하는 경우, 이들 데이터 전송은 보다 낮은 우선 순위를 갖는다는 점에 유의한다.

다음으로 낮은 레벨의 우선 순위는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 또는 마스터 프로세서(60)의 명령 캐시 메모리(11 및 12)의 명령 캐시 요청 또는 직접 외부 액세스(DEA) 요청, 및 인터럽트가 인에이블될 때의 마스터 프로세서(60)의 긴급한 패킷 전송 요청의 서비스를 포함한다. 이 레벨에서의 동시의 액세스 요청은 요청하는 프로세서에 따라 라운드 로빈 방식으로 취급된다. 캐시 서비스 요청 및 동일한 프로세서에 대한 집적 외부 액세스 요청은 토큰을 다음 프로세서로 통과시키기 전에 서비스된다.

다음 레벨의 우선 순위(206)는 높은 우선 순위의 패킷 전송이다. 제6도에 도시된 바와 같이, 여러가지 프로세서로부터의 패킷 전송은 라운드 로빈 방식으로 처리된다. 마스터 프로세서(60) 또는 디지탈 영상/그래픽 프로세서들(71, 72, 73 또는 74) 중의 하나는 그 프로세서가 요청된 데이터를 대기하는 경우 통상적으로 높은 우선 순위 패킷 전송을 이용한다. 전송 제어기(80)는 다음 레벨의 우선 순위(207)를 형성하는 낮은 우선 순위 패킷 전송에 대해 유사한 라운드 로빈을 이용한다. 마스터 프로세서(60) 또는 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74) 중의 하나는, 그 프로세서가 요청된 데이터를 대기하지 않는 경우, 통상적으로 낮은 우선 순위의 패킷 전송을 이용한다. 이하 더 설명되는 바와 같이, 요청하는 프로세서는 패킷 전송이 긴급하고 높은 우선 순위인지 또는 낮은 우선 순위인지를 표시한다.

각각의 요청의 우선 순위에 대한 원리가 이하 설명된다. 호스트 처리 시스템(1)과 같은 외부 장치는 그것이 원한다면 즉각적인 액세스를 할 수 있어야 한다. 외부 장치는 아래에 더 설명되는 REQ[1:0] 핀을 모니터함으로써 요청되는 경우에 높은 우선 순위 요청을 위해 전송 제어기(80)에 다시 영상 시스템 버스를 내줄 수 있다. 프레임 제어기(90) 요청은 시간이 임박한 VRAM 전송 사이클이 비디오 디스플레이 또는 캡쳐에 대한 중단없이 일어날 수 있도록 두번째로 높은 우선 순위를 받는다. 호스트 요청 사이클 및 프레임 제어기(90) 요청은 단속적으로만 발생하기 때문에, 높은 우선 순위를 필요로 하는 긴급한 DRAM 리프레시는 그들보다 낮은 우선 순위를 부여받는다. 다음의 우선 순위 레벨은 마스터 프로세서(60)의 인터럽트가 디스에이블될 때 발생하는 마스터 프로세서(60)의 캐시 서비스, 직접 외부 액세스 및 긴급 우선 순위 전송 사이클에 대한 것이다. 이것은 마스터 프로세서(60)의 인터럽트 서비스 루틴(통상적으로 인터럽트를 디스에이블함)이 가능한 한 신속하게 수행되게 하여 시스템 성능을 최대화한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 캐시 서비스 및 직접 외부 액세스 요청은 우선 순위에 있어서 다음이다. 프로세서는 요청이 서비스될 때까지 아이들 상태에 있기 때문에 이들이 신속하게 서비스되는 것이 중요하다. 또한, 마스터 프로세서(60)는 시스템 내의 다른 곳으로부터의 인터럽트 요청을 서비스하지 않기 때문에, 마스터 프로세서(60) 캐시 서비스와, 직접 외부 액세스 요청 및 긴급한 패킷 전송은 인터럽트가 인에이블되는 경우 이 우선 순위에 있게 된다. 높은 우선 순위의 패킷 전송은 요청하는 프로세서가 전송을 종료하도록 데이터를 대기하고 있거나 전송 제어기(80)가 외부 버스 대역폭을 최적화하기 위해 크로스바 액세스를 위한 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 보다 높은 우선 순위를 취할 필요가 있다는 것을 함축한다. 낮은 우선 순위 패킷 전송은 그들에게 매우 낮은 우선 순위가 주어지도록 프로세서가 데이터를 대기하고 있지 않는다는 것을 함축한다. 고려된 실시예에서 최저 우선 순위는 트리클 리프레시 사이클(trickle refresh cycle)에 부여된다. 이들은 외부 버스가 아이들이고 리프레시 백 로그가 0이 아닌 때에만 수행된다. 이것은 백 로그를 낮추어 주고, 높은 우선 순위의 긴급한 리프레시가 나중에 요청될 가능성을 감소시킨다.

전송 제어기(80)는 서로 다른 프로세서들로부터의 동일한 우선 순위의 다중 요청을 수신할 때마다, 그들 간에 라운드 로빈을 형성할 것이다. 라운드 로빈은 고정 사이클의 우선 순위 방식이다. 이것은 어떠한 프로세서도 라운드 로빈으로부터 제거되지 않고 사이클 내에서의 프로세서의 순서도 변경되지 않는다는 것을 의미한다. 특정한 프로세서의 요청이 완료될 때, 라운드 로빈 토큰은 체인 내에서 보류 중인 요청을 갖는 다음 프로세서로 넘겨질 것이다. 이것은 다른 프로세서들로부터의 동일한 우선 순위의 요청들이 서비스될 필요가 있을 때 어느 한 프로세서가 전송 제어기(80)를 독점하는 것을 방지한다.

전송 제어기(80)의 크로스바 우선 순위는 전송 제어기(80)가 서비스하고 있는 요청의 레벨에 따라 동적으로 변화한다. 이것은 제7도에 도시된 계층(210)에 도시되어 있다. 전송 제어기(80)는 다음과 같은 우선 순위가 할당된다. 전송 제어기(80)는 긴급한 우선 순위의 패킷 전송 요청, 캐시 서비스 요청, 직접 외부 액세스 요청을 서비스할 때, 또는 그것의 파이프라인을 플러시(flush)할 때, 마스터 프로세서(60)의 우선 순위(212)보다 높은 우선 순위(211)를 가지고 동작한다. 전송 제어기(80)는 긴급한 DRAM 리프레시 요청, 프레임 제어기(90)로부터의 요청, 호스트 인터페이스 요청 또는 소프트 재설정을 수신할 때마다 그것의 파이프라인을 플러시한다. 이것은 완료 대기 중인 외부 사이클에서 시작하지 않을 수도 있다. 파이프라인 플러싱은 경우에 따라서만 발생하고 짧은 시간 사이클 동안만 마스터 프로세서(60)를 로크 아웃(lock out) 한다.

전송 제어기(80)에는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 우선 순위(214)보다 높지만 높은 우선 순위의 패킷 전송을 위한 마스터 프로세서(60) 보다는 아래 순위인 우선 순위(213)가 주어진다. 이것은 마스터 프로세서(60)를 로크 아웃하지 않고 전송 제어기(80)에 최대로 가능한 우선 순위를 준다. 일반적으로, 마스터 프로세서(60)는 제어기로서 사용될 것이기 때문에, 연장된 사이클동안 이것을 로크 아웃하면 바람직하지 못한 시스템 관계가 유발될 수 있다.

전송 제어기(80)는, 낮은 우선 순위의 패킷 전송을 수행하고 있는 동안에는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 라운드 로빈의 우선 순위(214)보다 낮은 우선 순위(215)를 갖는다. 이것은 패킷 전송이 낮은 우선 순위를 가질 때 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)로부터 크로스바 대역폭을 스틸링(stealing)하는 것을 방지한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 로컬 메모리 액세스들은 서로 다른 메모리들을 향한 것이며 간섭할 수 없다는 점에 유의한다. 그러므로, 이들 로컬 메모리 액세스는 라운드 로빈에서 동일한 스테이지를 갖는다.

보다 높은 우선 순위 요청이 전송 제어기(80)에 의해 수신될 때마다, 이것은 새로운 요청의 크로스바 우선 순위에 있는 현재의 동작을 완료하거나 중지할 것이다. 이것은 시스템 차단이 일어나지 않게 한다. 그러므로, 예를 들어, 높은 우선 순위 패킷 전송 요청이 받아들여질 때, 낮은 우선 순위의 패킷 전송의 중지가 높은 우선 순위 레벨에서 발생할 것이다.

전송 제어기(80)는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 명령 캐시 미스와 마스터 프로세서(60) 명령 및 데이터 캐시 미스를 자동적으로 서비스한다. 다중 캐시 서비스 요청이 수신될 때, 전송 제어기(80)는 제6도에 도시한 바와 같은 라운드 로빈 방식으로 이들에 우선 순위를 부여한다. 전송 제어기(80)는 그것의 캐시 미스가 서비스되었을 때 요청 프로세서에 신호한다. 마스터 프로세서(60)는, 그 명령과 데이터 캐시 모두가 라운드 로빈 상의 순번 내에서 서비스되게 할 수 있다.

디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 명령 캐시는 4개의 128 바이트(16-명령) 서브-블럭을 각각 갖는 4개의 블럭으로 구성된 일방향 세트 연관 캐시(one-way set associative cache)(완전 연관)이다. 다중-방향 세트 연관 캐시는 다른 실시예들에서 고려된다. 디지탈 영상/그래픽 프로세서에서 캐시 미스가 발생하면, 그것의 프로그램 플로우 제어부는 전송 제어기(80)에 신호하여, 캐시 미스 서비스를 요청한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)는 명령이 배치되어야 하는 캐시 블럭을 결정하고, 그 어드레스는 물론 이 정보를 전송 제어기(80)에 전달한다. 다음에 전송 제어기(80)는 외부 메모리로부터 완전한 서브-블럭(128 블럭)을 페치하여, 요청하는 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)의 적절한 캐시 서브-블럭 내에 그것을 배치한다. 다음에, 전송 제어기(80)는 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)에 요청이 서비스되어 그 프로세서가 자신의 프로그램 실행을 계속할 수 있다는 것을 알린다.

마스터 프로세서(60) 캐시는 각각 4 블럭으로 구성된 4 방향 세트 연관 캐시이다. 각각의 블럭은 4개의 64 바이트 서브-블럭을 포함한다. 마스터 프로세서(60)는 자신의 명령 캐시 또는 데이터 캐시, 또는 둘 다에 서비스를 요청할 수 있다. 마스터 프로세서(60) 명령 캐시 서비스 요청은, 전송 제어기(80)에 의해 페치된 서브-블럭의 크기가 단지 64 바이트인 것만을 제외하면, 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)와 동일하게 처리된다.

마스터 프로세서(60)의 데이터 캐시 메모리(13 및 14)는, 전송 제어기(80)가 그 내용을 외부 메모리에 다시 기입하도록 요청될 수 있다는 점에서 명령 캐시(11 및 12)와는 다르다. 마스터 프로세서(60)의 데이터 캐시 미스에 있어서, 전

송 제어기(80)는 마스터 프로세서(60)의 명령 캐시와 같이 64 바이트 서브-블럭을 페치한다. 그러나, 매칭 태그 어드레스가 발견되지 않고 모든 블럭이 사용되어 마스터 프로세서(60) 내에서 블럭 미스가 발생하는 경우, 마스터 프로세서(60)는 우선 전송 제어기(80)가 그 블럭이 대체되기 전에 " 최근에 가장 적게 사용된" 블럭 내의 임의의 더티 서브-블럭을 재기입하도록 요청할 것이다. 더티 서브 블럭의 재기입은 모두 라운드 로빈 우선 순위의 단일 마스터 프로세서(60)의 " 순번(turn)" 내에서 일어날 수 있다. 또한, 전송 제어기(80)는 특별한 캐시 명령에 응답하여 더티 서브-블럭을 다시 기입하라고 요청될 수 있다.

전송 제어기(80)는 마스터 프로세서(60) 및 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)로부터의 모든 직접 외부 액세스(DEA) 요청을 처리하는 것을 담당한다. 직접 외부 액세스 사이클은, 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)가 외부 메모리 내의 데이터에 직접 액세스하게 하고 마스터 프로세서(60)가 자신의 데이터 캐시를 바이패스하게 한다. 직접 외부 액세스의 액세스들은 높은 우선 순위를 부여받기 때문에, 바이트, 하프-워드(16 비트), 워드(32 비트), 또는 더블-워드(64 비트) 길이일 수 있는 단일 액세스로 제한된다. 이것은, 하나의 프로세서가 다수의 직접 외부 액세스 사이클로 외부 버스를 독점하여 다른 프로세서들의 직접 외부 액세스 요청 및 캐시 미스가 서비스되지 못하게 하는 것을 방지한다. 직접 외부 액세스 사이클은 프로그램 가변 또는 오프칩 레지스터와 같은 단일 오프칩 메모리 위치에 대한 고속의 액세스가 요청될 때 사용될 예정이다.

디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)는 그들의 파라메타 메모리(25, 30, 35 및 40), 및 그들의 데이터 메모리(22, 23, 24, 26, 28, 29, 32, 33, 34, 37, 38 및 39)에 정상적으로 액세스할 수 있다. Hex " 02000000" 과 그 이상의 어드레스에 대한 액세스는 자동적으로 직접 외부 액세스 요청이 전송 제어기(80)에 보내지게 한다. 캐시/직접 외부 액세스 라운드 로빈 내에서 요청하는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 순번에 도달하면, 이 요청이 서비스될 것이다. 캐시 미스와 직접 외부 액세스 요청이 둘 다 보류 중인 경우에는, 이들이 하나의 순번에서 서비스될 수 있다. 마스터 프로세서(60)의 파라메타 메모리(15)와 같은 크로스바를 통해 액세스될 수 없는 온-칩 메모리 영역에 대한 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)의 액세스는 전송 제어기(80)에 대한 직접 외부 액세스 요청으로 변환될 것이다. 그러나 이 직접 외부 액세스는 오류를 발생시킬 것이다. 오류를 발생시키는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 직접 외부 액세스는 오류가 있는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 캐시 사이클과 동일하게 처리된다.

마스터 프로세서(60)는, 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)와 약간 다른 방식으로 직접 외부 액세스 사이클을 사용한다. 마스터 프로세서(60)는 정상적으로 데이터 캐시 메모리(13 및 14)를 통해 외부 메모리에 액세스한다. 마스터 프로세서(60)을 데이터 캐시 메모리(13 및 14)를 바이 패스하고 액세스 메모리에 직접 액세스하기 위해 직접 외부 액세스를 사용한다. 직접 외부 액세스 사이클은 특별한 메모리 로드 또는 저장 명령을 사용하여 명백하게 특정된다. 마스터 프로세서(60)가, (디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 명령 캐시 메모리(21, 26, 31 및 36) 등의) 마스터 프로세서(60)에 액세스할 수 없는 온-칩 어드레스에 대해 이러한 특별한 로드 또는 저장 명령을 수행하기를 시도한다면, 동작은 전송 제어기(80)에 의해 처리될 수 없는 직접 외부 액세스 요청으로 변환될 것이다. 온-칩 레지스터 또는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 데이터 메모리(22, 23, 24, 27, 28, 29, 32, 33, 34, 37, 38 및 39)와 같은 액세스 가능한 온-칩 메모리 영역에 대한 특별한 메모리 로드 또는 저장 동작은 정상 로드 또는 저장 동작으로 변환될 것이고, 직접 외부 액세스 요청은 전송 제어기(80)를 통해 발생할 것이다.

제8도는 전송 제어기(80)의 하이 레벨 블럭도를 도시한다. 각각의 주 블럭에 대한 설명이 아래에 제공된다. 전송 제어기(80)는 내부 메모리 인터페이스(301), 외부 메모리 인터페이스(302), 요청 큐잉 및 우선 순위화 회로(303), 캐시, VRAM 및 리프레시 제어기(310), 패킷 전송 선입 선출(FIFO) 버퍼(311), 캐시 버퍼(312), 소스 레지스터(321) 및 소스 제어 논리(322)로 이루어진 소스 머신(320), 소스 멀티플렉서 및 정렬 논리(330), 데스티네이션 레지스터(341)

및 데스티네이션 제어 논리(342)로 이루어진 데스티네이션 머신(340), 및 데스티네이션 멀티플렉서 및 정렬 논리(350)를 포함한다.

전송 제어기(80)는 멀티프로세서 집적 회로(100)에 의한 외부 메모리로의 모든 액세스를 관할한다. 외부 메모리 인터페이스는 여러가지 다른 형태 및 크기의 메모리 장치 및 주변 장치를 수용하도록 설계된다. 액세스되는 메모리의 형태는 동적으로 결정되어, 각각의 메모리 사이클의 타이밍이 액세스되는 장치에 대해 최적화될 수 있게 한다.

다음은 외부 메모리 인터페이스(302)를 통한 외부 데이터 액세스용으로 사용되는 멀티프로세서 집적 회로(100)의 리스트이다.

어드레스 버스 A[31:0], 이것은 멀티프로세서 집적 회로(100)로부터의 32 비트 바이트 어드레스를 외부 메모리에 제공한다. 이 어드레스는 DRAM 액세스용으로 멀티플렉스될 수 있다.

어드레스 시프트 선택 AS[2:0], 이들 입력은 전송 프로세서(80)에 의해 컬럼 어드레스에 주어지는 시프트의 양을 결정한다. (아무것도 포함하지 않는) 8개의 시프트 양은 표 4a 및 4b에 표시된 바와 같이 지원된다. 표 4a는 어드레스 버스 비트 a[31:16] 상의 출력을 나타내고 표 4b는 어드레스 버스 비트 A[15:0] 상의 출력을 나타냄에 유의한다.

[표 4a]

	A[31:16]															
AS[2:0]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0 0 0	31	30	20	28	27	26	25	24	23	22	21	20	19	18	17	16
0 0 1	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
0 1 0	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
0 1 1	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6
1 0 0	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5
1 0 1	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4
1 1 0	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3
1 1 1	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2

[표 4b]

	A[15:0]															
AS[2:0]	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 0 0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 0 1	7	6	5	4	3	2	1	0	X	X	X	X	X	2	1	0
0 1 0	6	5	4	3	2	1	0	X	X	X	X	X	X	2	1	0
0 1 1	5	4	3	2	1	0	X	X	X	X	X	X	X	2	1	0
1 0 0	4	3	2	1	0	X	X	X	X	X	X	X	X	2	1	0
1 0 1	3	2	1	0	X	X	X	X	X	X	X	X	X	2	1	0
1 1 0	2	1	0	X	X	X	X	X	X	X	X	X	X	2	1	0
1 1 1	1	0	X	X	X	X	X	X	X	X	X	X	X	2	1	0

로우 어스레스 시간(row address time)동안, 어드레스 버스 A[31:0]은 정상 어드레스 값을 출력한다. 칼럼 어드레스 시간동안, 어드레스 버스 A[31:0]은 표 4a 및 4b에 표시한 바와 같이 어드레스 시프트 선택 AS[2:0]에 따라 시프트 된 어드레스 값을 출력한다. " X" 라고 표기된 어드레스 라인 출력은 구동되지 않고, 고 임피던스 상태로 설정됨에 유의한다.

버스 크기 선택 BS[1:0]. 이것은, 64비트 폭보다 작은 데이터 버스에 대해 다이내믹 버스 사이징을 허용해 준다. 이 입력들의 엔코딩은 표 5에 나타나 있다.

[표 5]

BS[1:0]		버스 크기
0	0	8 비트
0	1	16 비트
1	0	32 비트
1	1	64 비트

출력 클록 CLKOUT. 이 클록 출력은 외부 논리가 멀티프로세서 집적 회로(100)의 동작과 동기화될 수 있게 한다.

칼럼 타이밍 선택 CT[1:0], 이 입력은 전송 제어기(80)에 의해 부여되는 현재 메모리 사이클의 타이밍을 결정해 준다. 이 입력의 엔코딩은 표 6에 도시되어 있다.

[표 6]

CT[1:0]		메모리 타이밍
0	0	파이프라인된 1 사이클/칼럼
0	1	비 파이프라인된 1 사이클/칼럼
1	0	비 파이프라인된 2 사이클/칼럼
1	1	비 파이프라인된 3 사이클/칼럼

칼럼 어드레스 스트로브$\overline{CAS}$ [7:0]. 이런 출력은 DRAM/VRAM의 $\overline{CAS}$ 입력을 구동한다. 개별적 바이트 액세스를 허용하기 위해 8개의 스트로브가 제공된다. 엔디언 모드와 무관하게, $\overline{CAS}$ [0]은 D[7:0] 상의 데이터 전송에 대응하고, $\overline{CAS}$ [1]은 D[15:8] 상의 전송에 대응하는 방식으로 계속된다. 투과 전송시, 이런 신호들은 바이트 기입 스트로브로 사용된다.

데이터 버스 D[63:0]. 이 입력/출력 버스는 멀티프로세서 집적 회로(100)의 내외로 메모리 사이클당 최대 64 비트를 액세스할 수 있게 한다.

데이터 버퍼 출력 인에이블$\overline{DREN}$. 이 출력은 데이터 송수신기를 켜는데 사용된다.

데이터 방향 인디케이터(indicator) $\overline{DDIN}$. 이 출력은 데이터 송수신기에 방향 인디케이터를 제공한다.

특수 함수 핀 DSF1과 DSF2. 이 출력들은 특수 VRAM 함수를 선택하는 데에 사용된다.

폴트$\overline{FAULT}$ 이 입력은 멀티프로세서 회로(100)에게 메모리 오류가 발생했음을 알려준다.

페이지 크기 선택 PS[2:0]. 이 입력은 멀티프로세서 집적 회로(100)에게 현재 액세스되는 메모리의 페이지 크기를 표지해 준다. 이런 입력의 이용은 아래에 더 자세히 설명된다.

로우 어드레스 스트로브$\overline{RAS}$. 이 입력은 DRAM/VRAM들의 $\overline{RAS}$ 입력을 구동한다.

레디 READY. 이 입력은 완료된 메모리 사이클에 대해 외부 장치가 준비되었다는 것을 표지한다. 이것은, 전송 제어기(80)가 대기 상태(wait states)를 메모리 사이클에 삽입하는 데에 이용된다.

재시도$\overline{RETRY}$. 이 입력은 멀티프로세서 집적 회로(100)에게 메모리가 여유가 없어, 전송 제어기(80)가 메모리 사이클을 다시 시작해야만 함을 나타낸다.

로우 래치(row latch) $\overline{RL}$. 이 출력은 유효한 32 비트 어드레스가 어드레스 버스 상에 존재한다는 것을 나타낸다.

상태 코드 STATUS[4:0]. 이 출력은 현재 전송 제어기 메모리 사이클의 타입과 기원에 대한 상세한 정보를 제공한다. 로우 액세스의 시작으로부터, 칼럼 액세스의 출발에 이르기까지, STATUS[4:0] 출력은 표 7에 도시된 대로 엔코딩된다. 이 엔코딩은 아래에 설명되는$\overline{UTIME}$ 과 함께, 외부 장치가 멀티프로세서 집적 회로(100)의 메모리 타이밍에 대응하는 메모리 타이밍을 발생시킬 수 있게 한다.

[표 7]

STATUS[4:0]	활동
0 0 0 0 0	정규 판독
0 0 0 0 1	정규 기입

00010	리프레시
00011	보류
00100	주변 장치 패킷 전송 판독
00101	주변 장치 패킷 전송 기입
00110	보류
00111	보류
01000	보류
01001	블록 기입
01010	보류
01011	보류
01100	보류
01101	로드 칼라 레지스터
01110	보류
01111	보류
10000	프레임 0 완전 시프트 레지스터 판독
10001	프레임 0 완전 시프트 레지스터 기입
10010	프레임 0 분할 시프트 레지스터 판독
10011	프레임 0 분할 시프트 레지스터 판독
10100	프레임 1 완전 시프트 레지스터 판독
10101	프레임 1 완전 시프트 레지스터 판독
10110	프레임 1 분할 시프트 레지스터 판독
10111	프레임 1 분할 시프트 레지스터 판독
11000	보류
11001	보류
11010	보류
11011	보류
11100	패킷 전송 완전 시프트 레지스터 판독
11101	패킷 전송 완전 시프트 레지스터 기입
11110	보류
11111	보류

로우 시간 상태 코드에 대한 세부 사항은 다음과 같다. 정상 판독 코드는 패킷 전송, 캐시 미스, 직접 외부 액세스 요청에 의해 발생되는 임의의 판독 사이클에 대한 출력이다. 정상 판독 코드는 패킷 전송, 데이터 캐시 재기입 또는 직접 외부 액세스 요청에 의해 발생되는 임의의 정상 기입 사이클에 대한 출력이다. 리프레시 코드는 트리클 리프레시 사이클 동안, 그리고 리프레시 제어기로부터의 긴급 리프레시 요청에 의해 발생되는 버스트 리프레시 사이클 동안 출력된다.

주변 장치 패킷 판독과 기입은 아래에 더 설명될 것이다. 주변 장치 패킷 전송 판독 코드는 주변 장치 패킷 전송의 결과

로 발생되는 메모리 판독 사이클 동안 출력된다. 이는 순차적인 칼럼 액세스에 의한 메모리로부터의 데이터 판독이 그 전송을 시발시킨 주변 장치에 의해 래치되어야 한다는 것을 나타낸다. 주변 장치 패킷 전송 기입 코드는 주변 장치 패킷 전송의 결과 발생하는 메모리 기입 사이클동안 출력된다. 이는 멀티프로세서 집적 회로(100)가 순차적인 칼럼 액세스 동안 자신의 데이터 버스를 고 임피던스 상태에 놓이게 하여, 주변 장치가 메모리에 저장될 데이터와 함께 버스를 구동할 수 있게 해 준다는 것을 나타낸다.

블록 기입은 특별 VRAM 사이클이다. 블록 기입 코드는 블록 기입 사이클 동안 블록 기입 액세스 모드를 이용한 패킷 전송에 의해 발생되어 VRAM으로 출력된다. 로드 칼라 레지스터 코드는 자신의 액세스 모드로서 특정화된 블록 기입을 소유하는 패킷 전송의 칼라 레지스터 로드 부분 동안의 출력이다. 이 사이클은 시스템 VRAM의 칼라 레지스터 내에 데이터를 배치시키는 데에 이용된다. 이 칼라 레지스터 데이터는 블록 기입 사이클동안에 메모리에 기입된다.

패킷 전송 시프트 레지스터 판독 코드는, 직렬 레지스터 전송 액세스 모드를 사용하는 패킷 전송의 소스 사이클동안 출력된다. 이 사이클은 시스템 VRAM 상에서 판독 메모리 대 레지스터 전송(read memory to register transfer)을 수행한다. 패킷 전송 시프트 레지스터 기입 코드는 직렬 레지스터 전송 액세스 모드를 특정화시켜주는 패킷 전송의 데스티네이션 싸이클동안 출력된다. 이 사이클은 시스템 VRAM 상에서 기입 레지스터 대 메모리 전송을 수행한다.

프레임 사이클은 프레임 제어기(90)에 요청되는 메모리 액세스이다. 시프트 레지스터 판독 및 기입은 특별한 직렬 레지스터 VRAM 사이클이다. 1 판독 전송 코드의 프레임 0은 프레임 제어기(90)에 의해 요청되는 각각의 완전 판독 전송 사이클동안 출력된다. 이 사이클은 시스템 VRAM 상에서 완전 판독 전송 사이클을 수행한다. STATUS[2] 신호는, 프레임 0 요청 사이클동안에는 " 0" 이며, 프레임 1 요청 사이클동안에는 " 1" 이다. 프레임 0 또는 1 기입 전송 코드는 프레임 제어기(90)에 의해 요청된 완전 기입 전송 사이클동안 출력된다. 이 사이클은 시스템 VRAM 상에서 완전 기입 전송 사이클을 수행한다. STATUS[2] 신호는, 프레임 0 요청 사이클 동안에는 " 0" 이며, 프레임 1 요청 사이클동안에는 " 1" 이다. 프레임 0 또는 1 분할 판독 전송 코드는 프레임 제어기(90)가 분할 판독 전송 사이클을 요청할 때 출력된다. 이 사이클은 VRAM 메모리의 로우로부터, 직렬 레지스터의 절반으로 전송한다. STATUS[2] 신호는 프레임 0 요청 사이클동안에는 " 0" 이고, 프레임 1 요청 사이클동안에는 " 1" 이다. 프레임 0 또는 1 분할 기입 전송 코드는, 프레임 제어기(90)에 의해 요청된 분할 기입 전송 주기동안 출력된다. 이 주기는 VRAM 직렬 레지스터의 절반을 메모리 어레이로 전송한다. STATUS[2] 신호는 프레임 0 요청 사이클동안에는 " 0" 이고, 프레임 1 요청 사이클동안에는 " 1" 이다.

칼럼 액세스의 시작에서 로우 액세스의 시작에 이르기까지, STATUS[4:0] 출력은 표 8에 도시된 대로 엔코딩된다. 이 정보는 전송 제어기(80)를 통해 멀티프로세서 집적 회로(100)에 의해 출력되어, 메모리 사이클을 요청하는 모듈을 표지해준다. 이 신호들은 시스템 분석과 디버그 작업을 위해 제공되며, 외부 메모리 시스템에는 필요치 않다.

[표 10]

STATUS[4:0]	활동
0 0 0 0 0	DIGP0 낮은 우선 순위의 패킷 전송
0 0 0 0 1	DIGP0 높은 우선 순위의 패킷 전송
0 0 0 1 0	DIGP0 캐시
0 0 0 1 1	DIPG0 DEA
0 0 1 0 0	DIGP1 낮은 우선 순위의 패킷 전송
0 0 1 0 1	DIGP1 높은 우선 순위의 패킷 전송
0 0 1 1 0	DIGP1 캐시
0 0 1 1 1	DIPG1 DEA
0 1 0 0 0	DIGP2 낮은 우선 순위의 패킷 전송
0 1 0 0 1	DIGP2 높은 우선 순위의 패킷 전송
0 1 0 1 0	DIGP2 캐시
0 1 0 1 1	DIGP2 DEA
0 1 1 0 0	DIGP3 낮은 우선 순위의 패킷 전송
0 1 1 0 1	DIGP3 높은 우선 순위의 패킷 전송
0 1 1 1 0	DIGP3 캐시
0 1 1 1 1	DIGP3 DEA
1 0 0 0 0	MP 낮은 우선 순위의 패킷 전송
1 0 0 0 1	MP 높은 우선 순위의 패킷 전송
1 0 0 1 0	MP 긴급 패킷 전송 인터럽트 인에이블됨
1 0 0 1 1	MP 긴급 패킷 전송 인터럽트 디스에이블됨
1 0 1 0 0	보류
1 0 1 0 1	보류
1 0 1 1 0	MP 명령 캐시 인터럽트 인에이블됨
1 0 1 1 1	MP 명령 캐시 인터럽트 디스에이블됨
1 1 0 0 0	MP DEA 인터럽트 인에이블됨
1 1 0 0 1	MP DEA 인터럽트 디스에이블됨
1 1 0 1 0	MP 데이터 캐시 인터럽트 인에이블됨
1 1 0 1 1	MP 데이터 캐시 인터럽트 디스에이블됨
1 1 1 0 0	프레임 메모리 0
1 1 1 0 1	프레임 메모리 1
1 1 1 1 0	리프레시
1 1 1 1 1	보류

전송/출력 인에이블$\overline{TR}$. 전송 제어기(80)로부터의 이 출력은 DRAM 출력 드라이버와 VRAM 시프트 레지스터 전송 사이클을 인에이블시킨다.

사용자-타이밍 선택$\overline{UTME}$. 이 입력은 전송 제어기가$\overline{RAS}$와$\overline{CAS}$[7:0]의 타이밍을 변조시켜서, 사용자가 그들 자신의 메모리 타이밍을 생성할 수 있게 한다. 또한, 이 입력은 멀티프로세서 집적 회로(100)가 동작할 엔디언 모드를 결정하기 위해서 재설정 시에도 이용된다.

기입 인에이블$\overline{WE}$. $\overline{WE}$는 정상적으로는 기입 또는 기입 전송 사이클이 진행되고 있음을 메모리에 지시해 주는 출력이다. 내부 메모리 테스트 모드 내에서, WE는 내부 메모리로의 기입을 위해 외부 장치에 구동되는 입력이다.

멀티프로세서 집적 회로(100)에 의해 발생되는 각각의 외부 메모리 사이클은 페이지-모드 사이클을 제외하고 최소한 5 머신 상태 동안 지속된다. 머신 상태는 한 클록 사이클 길이이며, CLKOUT의 하강 에지에서 시작한다. 각각의 메모리 사이클은 어드레스 서브사이클 및 데이터 서브사이클의 두 부분을 갖는다. 페이지 모드 사이클은, 액세스가 한 어드레스 서브사이클과 멀티 데이터 서브사이클을 갖는 형식의 확장이 된다.

어드레스 서브사이클은 외부 메모리 사이클의 제1 머신 상태와 함께 시작하고, 넷 이상의 머신 상태 길이가 된다. 액세스를 위한 어드레스와 상태 코드는 이 시간에서는 출력이 된다. 이 사이클의 이 부분은, VRAM과 DRAM의 로우 어드레스가 이 시간에서 래치되기 때문에, 로우 어드레스 시간이라고 부른다.

어드레스 버스 A[31:0]은 현재 액세스되는 64 비트 워드의 시작 바이트를 지시하는 32 비트 어드레스를 출력시킨다. 액세스는 시작 바이트와 전송될 데이터량에 따라 1 내지 8 바이트의 어떤 값도 가능하다. 이 어드레스는 멀티프로세서 집적 회로(100)의 외부 메모리 공간을 디코딩하는 데에 이용된다. 외부 디코드 논리는, 액세스되고 있는 장치 또는 장치들의 어드레스 시프트 선택 AS[2:0]을 통한 타입, 칼럼 타이밍 선택 CT[1:0]을 통한 속도, 페이지 크기 선택 PS[2:0]을 통한 페이지 크기, 버스 크기 선택 BS[1:0]을 통한 데이터 버스 폭의 표시를 위해, 멀티프로세서 집적 회로(100)에 다수의 신호를 반송해 준다.

이 정보는 어드레스 서브사이클의 길이뿐만 아니라, 데이터 서브 사이클의 길이, 어드레싱, 갯수를 결정하는 데에 사용된다. 어드레스와 상태는$\overline{RL}$ 또는$\overline{RAS}$ 로 래치될 것이다.

어드레스 서브사이클은 수행 중인 액세스 타입이 요청하는 바대로, 4개의 머신 상태를 초과하며, 머신 상태의 정수배 개수만큼 자동적으로 확장될 것이다. 또한 대기 상태 삽입에 의해 확장될 수도 있다. 이 서브사이클은 최소한 1 머신 상태 길이이며, 즉각적으로 어드레스 서브사이클을 따라간다. DRAM과 VRAM을 위한 칼럼 어드레스는 이 시점에 출력이 되며, 데이터는 멀티프로세서 집적 회로(100)와 외부 메모리 사이에서 전송된다. 메모리 사이클 중 이런 부분은 보통 칼럼 어드레스 시간이라고 부른다.

데이터 버스 D[63:0]은 멀티프로세서 집적 회로(100)와 외부 메모리 사이에서 데이터를 전송한다. 데이터는 기입 사이클동안 구동되거나, 판독 사이클동안 래치된다. 버스 상의 유효 데이터 위치는 멀티프로세서 집적 회로(100)의 엔디언 모드, 전송되는 데이터의 양, 그리고 메모리 폭에 의해 결정된다.

이 시간 동안의 칼럼 어드레스 출력은 32 비트 바이트 어드레스의 시프트된 버젼이 된다. 어드레스 버스 A[31:0] 상의 어드레스 정렬은, 어드레스 서브사이클 동안에 멀티프로세서 집적 회로(100)에 입력된 어드레스 시프트 선택 버스 A[31:0] 값에 의해 결정된다.

데이터 서브사이클의 길이는, 정규적으로는, 어드레스 서브사이클동안에 멀티프로세서 집적 회로(100)에 입력된 칼럼 타이밍 선택 CT[1:0]에 의한 결정에 따라, 1, 2 또는 3 머신 상태가 된다. 더 긴 액세스 시간을 요청하는 장치들은 어드레스나 데이터 서브사이클 중 어느 하나의 대기 상태를 삽입할 수 있다.

현재 메모리 액세스가 로우 시간에서의 페이지 크기 선택 PS[2:0] 입력에 기초한 이전 액세스와 동일한 방향이고 동일한 메모리 페이지 내에 있을 때마다, 전송 제어기(80)는 페이지-모드 사이클을 이용할 것이다. 페이지-모드 사이클은 다수의 데이터 서브사이클이 후속하는 하나의 어드레스 서브사이클로 구성된다. 데이터가 동일 메모리 페이지 내에서만 연속적일 필요는 없다.

어드레스 및 데이터 서브사이클 동안에, 멀티프로세서 집적 회로(100)는 STATUS[4:0] 핀에 대한 상태 코드를 출력한다. 이 상태 코드들은 실행 중인 외부 사이클에 관한 정보를 제공한다. 어드레스 서브사이클의 로우 시간 동안, STATUS[4:0] 핀은 실행 중인 사이클 타입을 표지해 주기 위해 표 7에 도시된 대로 코드를 출력시킨다. 로우 시간 상태 코드는 $\overline{RM}$ 또는 $\overline{RAS}$ 신호에 의해 래치될 수 있으며, 외부 논리에 의해 메모리-뱅크 디코딩을 수행하거나, 특별 하드웨어 특성을 발휘하게 하는 데에 이용된다. 데이터 서브사이클의 칼럼 시간동안, STATUS[4:0] 핀 상의 정보는 사이클 및 요청하는 프로세서에 관한 세부 사항을 제공하도록 변화한다. 이런 활동 코드들이 표8에 나타나 있다. 이 코드들은 사이클 자체의 유형에 대해서는 어떠한 정보도 제공하지 않으며 따라서 주로 시스템 디버그를 촉진시키는 데에 사용된다.

전송 제어기(80)가 외부 메모리와 적절히 통신하기 위해서는, 전송 제어기가 액세스하는 메모리 타입이 무엇인지 알아야만 한다. 이는, 멀티프로세서 집적 회로(100)의 어드레스 시프트 선택 AS[2:0], 버스 크기 선택 BS[1:0], 칼럼 시간 선택 CT[1:0], 그리고 페이지 크기 선택 PS[2:0] 입력을 이용하여 로우 시간에서 메모리 타입을 식별함으로써 성취된다. 멀티프로세서 집적 회로(100)는 외부 메모리 어드레스와 로우 시간 상태 코드들을 출력한 후, 이런 입력들을 샘플링하여 메모리 타입을 결정한다. 이는 어드레스를 디코딩하기 위한 외부 논리 시간을 제공하여, 메모리 식별 신호를 그들의 적절한 레벨로 구동한다. 선택된 메모리 타입은 다음 어드레스 서브사이클까지 유효하게 유지된다.

멀티프로세서 집적 회로(100)가 DRAM과 SRAM을 지원하기 때문에, 반드시 어드레스 버스 상에서 멀티플렉스된 로우와 칼럼 어드레스를 제공해야 한다. 멀티프로세서 집적 회로(100)는 언제나 로우 시간에서 완전 32 비트 바이트 어드레스를 출력한다. 칼럼 시간에서는, 칼럼 어드레스를 이미 DRAMS에 의해 래치된 로우 어드레스에 정렬되도록 버스 상의 어드레스를 시프트해야만 한다. 칼럼 어드레스를 적절히 정렬하기 위해서는, 어레이 크기가 장치의 로우/칼럼 어드레스 비트의 수를 결정하므로, 멀티프로세서 집적 회로(100)는 반드시 자신이 액세스하는 DRAM의 어레이 크기를 알아야만 한다. 이는 어드레스 시프트 선택 AS[2:0] 입력을 이용하여 이루어진다. 외부 논리는 로우 시간에서 멀티프로세서 집적 회로(100)에 의해 출력된 어드레스 출력을 디코딩하며, 어드레스 시프트 선택 AS[2:0] 입력에 3 비트 시프트 코드를 제공한다. 전송 제어기(80)는 이 값을 샘플링하고 래치하며, 이 값을 이용하여, 어드레스가 동적 메모리에서 요청되는 대로 멀티플렉스되어야 하는지 또는 정적 메모리 및 주변 기기와 함께 사용되기 위해 멀티플렉스되지 않고 유지되어야 하는지를 결정한다. 표 4a 및 4b는 샘플링된 어드레스 시프트 선택 AS[2:0] 값이 칼럼 시간에서 출력된 어드레스에 어떻게 영향을 끼치는지를 나타내고 있다. 이 값이 "000" 이면, 시프트되지 않은 32 비트 어드레스가 각각의 순차적인 "칼럼" 액세스에 대해 출력된다. 이 값이 0이 아닌 경우, 순차적인 칼럼 어드레스는 표 4a 및 4b에 나타나 있는대로 시프트될 것이다. 시프트 값은 8 내지 14 비트의 범위이며, 이는 64K×N 내지 256M×N 어레이 크기에 대응하는 8 내지 14 어드레스 핀을 가진 동적 메모리가 지원된다는 것을 의미한다. 어드레스 버스 A[2:0]은 언제나 시프트된 양에 관계없이 바이트 어드레스 비트 0 내지 2를 출력한다는 것에 유의한다. 이는 동적 버스 사이징을 지원하기 위해 수행된다.

예를 들어, 1M×4 DRAM이 64 비트폭의 구성을 가지는 데이터 버스에 접속된 것으로 가정하자. 메모리는 로우와 칼럼 어드레스 각각 10 비트를 요청한다. 멀티프로세서 집적 회로(100)의 어드레스 비트의 어드레스 시프트 선택 AS[2:0]은 바이트 어드레스를 나타내며, 메모리 뱅크가 64 비트폭이고 개별 바이트가 $\overline{CAS}$ [7:0]스트로브에 의해 제어되기 때문에 무시될 수 있다. 이는 메모리가 비트 A[3]으로 시작하는 20개의 연속적인 어드레스 비트를 필요로 한다는 것을 의미한다. 표 4a 및 4b를 보면, " 011" 의 어드레스 시프트 선택 AS[1:0] 값이 10 비트의 멀티플렉스된 어드레스 10 비트를 제공한다는 것을 알 수 있다. 시작 어드레스 비트 A[3]이 칼럼 시간에서 어드레스 핀 A[13]에 대응하기 때문에, DRAM 어드레스 핀은 A[13]에서 시작하는 멀티프로세서 집적 회로(100)의 어드레스 핀에 접속된다.

전송 제어기(80)는 가능할 때는 언제든지 페이지-모드 사이클을 수행하기 때문에, 새로운 페이지에서 로우 액세스를 수행할 수 있도록, 현재 액세스 중인 메모리의 페이지 또는 로우 경계를 통과하는 시기를 알아야 한다. 현재의 액세스에 관한 페이지 크기는 외부 논리에 의해 페이지 크기 선택 PS[2:0] 핀 상에 3비트 값에 의해 배치된다. 전송 제어기(80)는 로우 시간에 이 핀들을 샘플링하며, 샘플링된 값을 이용하여, 변화하는 어드레스 비트 중 어느 것이 페이지 교체를 표지하는지를 결정한다. 샘플링된 값은 다음 로우 액세스 때까지 전송 제어기(80)에 의해 보존된다. 멀티 뱅크가 인터리빙될 수도 있기 때문에, 페이지 크기 선택 PS[2:0] 상에 표지된 페이지 크기가 반드시 어드레스 시프트 선택 AS[2:0]에 표지된 시프트 양에 대응하지는 않는다.

외부 메모리 액세스가 발생할 때마다, 전송 제어기(80)는 내부 LAST PAGE 레지스터(360) 내의 21개의 최상위 어드레스 비트를 기록한다. 그 다음, 각각의 순차적인 칼럼 액세스의 어드레스가 이 값과 비교된다. 이러한 처리가 제9도에 도시되어 있다. 페이지 크기 선택 PS[2:0]에서 입력된 값은, 비교시, LASTPAGE 레지스터(360)의 6개의 최하위 비트를 선택적으로 무시하는 데에 사용된다. 페이지 크기 선택 PS[2:0]-1의 값과 동일한, 비트수 아래의 LASTPAGE 레지스터 비트는 비교에서 무시되었다. 15개의 최상위 비트는 항상 비교되고, 다음 어드레스의 11개의 최하위 비트는 항상 무시된다. LASTPAGE 레지스터(360)의 인에이블된 비트와 다음 메모리 어드레스 사이에서 어떠한 정합도 발견되지 않으면, 페이지는 변경되고 다음 메모리 액세스는 로우 어드레스 사이클과 더불어 시작할 것이다. 페이지 크기 선택 PS[2:0] = " 000" 인 경우, 페이지 모드는 디스에이블되며 임의의 순차적인 사이클은 다음 로우 액세스와 함께 시작할 것이다.

예를 들어, 액세스되는 메모리가 다른 뱅크와 인터리빙되지 않고 64 비트 데이터 버스로 접속되는 16개의 1M×4 DRAM으로 구성되었다고 하자. 각각의 메모리 장치는 210 비트의 로우 크기를 갖고 있다.

하나의 로우 내의 임의의 위치는 DRAM의 페이지 모드 사이클동안 액세스될 수 있으므로, 이러한 구성의 페이지 크기는 " 210 위치/페이지 ×8 바이트/위치 = 8K 바이트/페이지" 가 될 것이다. 표 9가 보여 주듯이, 이는 페이지 크기 선택 PS[2:0] 값 " 011" 에 대응한다. 전송 제어기(80)가 로우 시간에서 페이지 크기 선택 PS[2:0] = " 011" 을 샘플링할 때, 페이지 경계가 통과되었는지를 결정하기 위해, 순차적인 액세스의 비트 31-13만이 점검될 것이다. 이는 이러한 특별한 구조의 DRAM의 로우 어드레스 비트 및 이 뱅크 디코드 비트에 대응함에 유의한다. 8개의 가능한 페이지 크기 선택 PS[2:0] 값에 대해 비교된 어드레스 비트와 페이지 크기가 표 9에 나타나 있다.

[표 9]

PS[2:0]	페이지 크기	비교된 비트
0 0 0	페이지 모드 아님	31- 0
0 0 1	2 K	31-11
0 1 0	4 K	31-12
0 1 1	8 K	31-13
1 0 0	16 K	31-14
1 0 1	32 K	31-15
1 1 0	64 K	31-16
1 1 1	128 K	31-17

LASTPAGE 레지스터(360)는 연관된 유효 비트(361)를 가진다. 이 유효 비트(361)는 LASTPAGE 레지스터(360)에 저장된 데이터가 유효(valid)로 처리되는지 또는 무효(invalid)로 처리되는지를 표지해 준다. LASTPAGE 레지스터(360)는, 재설정, 호스트 액세스, 폴트 및 재시도 후에, 또는 액세스의 방향 변화(즉, 판독-기입) 후에 무효로 취급된다. 이는 로우 액세스가 항상 순차적인 액세스 전에 발생하도록 강제한다. 또한 LASTPAGE 레지스터(360)는 리프레시 사이클, 프레임 제어기(90)가 요청하는 사이클, 및 패킷 전송으로 발생된 직렬 레지스터 전송 사이클 전후 모두에서 무효로 취급되어, 이들이 항상 단일 비페이지 모드 사이클로서 발생하게 한다. 주변 장치 패킷 전송의 특별한 경우, 전송은 주변 장치 전송 상태 코드가 출력될 것을 보장하도록, 항상 로우 액세스와 함께 시작한다. LASTPAGE 레지스터(360) 내의 데이터는 새로운 상태 코드가 출력될 수 있도록, 전송 완료 후에는 무효로 취급된다. 그러나 전송동안, LASTPAGE 레지스터(360)는 주변 장치 전송이 가능할 때는 언제든지 페이지-모드 주기를 이용할 수 있도록 정상적으로 작동할 것이다.

버스 크기 선택 BS[1:0] 핀은 로우 시간에서 샘플링되어, 현재 액세스의 버스 크기를 결정한다. 전송 제어기(80)는 표 5에 나타난 대로 8, 16, 32, 또는 64 비트의 버스 크기를 지원한다. 버스 크기의 설정은 전송 제어기(80)가 각각의 칼럼 액세스동안 전송할 수 있는 최대 바이트 개수를 결정해 준다. 요청되는 바이트 개수가 버스 크기를 초과하는 경우, 전송 제어기(80)는 자동적으로 멀티 액세스를 수행하여 전송을 완료할 것이다. 선택된 버스 크기는 데이터 중 어느 부분이 전송에 이용될지를 결정해 준다. 64 비트 메모리에 대해, 전체 버스가 전송에 유용하다. 32 비트 메모리에 대해서는, 데이터 버스 D[63:32] 핀이 빅 엔디언 모드에서 사용되고, 데이터 버스 D[31:0] 핀이 리틀 엔디언 모드에서 사용된다. 빅 엔디언 및 리틀 엔디언 각각에 대해서, 16비트 버스는 데이터 버스 D[63:48] 핀과 데이터 버스 D[15;0] 핀을 사용하고, 8 비트 버스는 데이터 버스 D[63:56] 핀과 데이터 버스 D[7:0] 핀을 사용한다. 어떤 크기의 버스가 사용되든 상관없이, 전송 제어기(80)는 언제나 버스의 적절한 부분에 데이터를 정렬시키며, 적절한 $\overline{CAS}$ 스트로브가 유효한 바이트만이 전송되도록 보장하는 작용을 하게 스트로브를 활성화시킨다.

전송 제어기(80)는 4개의 기초 메모리 타이밍 세트를 지원한다. 이는 대기 상태를 사용하지 않고서도 칼럼 액세스당 하나, 둘 또는 세개의 클록 사이클을 선택할 수 있게 한다. 모든 타입이 그 본질상 DRAM과 유사하지만, 각각의 타입은 DRAM 또는 SRAM의 특정 타입에 적합하도록 만들어져 있다. 사용되는 타이밍은 표 6에 나타나 있는 것과 같은 칼럼 타이밍 선택 CT[1:0] 입력에 의해 결정된다. 외부 디코드 논리는, 어드레스를 디코딩한 후 이 핀들을 적절한 레벨로 구동하여 현재 액세스되는 메모리 타입을 결정한다. 파이프라인되지 않은 1 사이클/칼럼 타이밍은 아주 빠른 액세스 시간을 갖는 장치에 사용되도록 디자인된다. 파이프라인된 1 사이클/칼럼 타이밍은 액세스가 파이프라인되었다는 것을 제외하고는, 1 사이클/칼림 타이밍과 유사하다. 한 어드레스는 1 사이클 동안 출력되며, 이 어드레스에 상응하는 데이터

는 다음 사이클 동안 액세스 된다. 이 타이밍은 파이프라인된 페이지 모드 사이클을 지원하는 DRAM/VRAM에서 사용되지만, 동기화된 SRAM 장치에서 사용되기도 한다. 파이프라인되지 않은 2개의 사이클/칼럼 타이밍 사이클은 SRAM과 고속 DRAM 장치에 대한 칼럼 액세스 시간의 2 클록 사이클을 제공한다. 이는 로우 어드레스 시간에 1 사이클을 부가해 준다. 파이프라인되지 않은 3개의 사이클/칼럼 주기는 DRAM과 다른 느린 장치에서 사용된다. 칼럼 액세스 시간의 3개 클록 사이클을 제공하는 데에 더하여, 로우 액세스 시간에 2 클록 사이클을 부가한다. 선택된 칼럼 타이밍은 전체 페이지에 대해, 즉 다음 로우 액세스가 발생할 때까지 유효하게 유지된다. 발생전 어드레스에 기초하여 적절한 칼럼 타이밍 선택 CT[1:0]을 제공하는 것은 외부 장치의 책임이다.

블록 기입과 로드 칼러 레지스터 사이클 동안, 버스 크기 선택 BS[1:0] 핀은 다른 목적으로 사용된다. 블록 기입은 64비트 버스에 대해서만 지원되기 때문에 버스 크기 정보는 필요없게 된다. 대신, 버스 크기 선택 BS[1:0]은 어드레스된 메모리가 지원하는 블록 기입 타입을 표지하는 데에 사용된다. 이 사이클 동안의 BS[1:0] 값은 표 23에 나타나 있다. 블록 기입은 아래에 자세히 논의된다.

제8도에 도시된 실시예에서, 멀티프로세서 집적 회로(100) 외부 메모리 사이클은 전송 제어기(80)의 외부 메모리 인터페이스(302)에 의해 발생된다. 외부 메모리 인터페이스(302)는 메모리 인터페이스 신호의 천이(transition)를 제어하는 상태의 시퀀스를 발생시키는 복합 상태 머신을 포함한다. 발생된 상태와 그들의 시퀀스는 수행되는 사이클의 형태, 액세스되는 메모리의 칼럼 타이밍, 수행될 다음 액세스, 폴트와 같은 내부 또는 외부 사건 등에 의해 변화한다. 실시예가 상태 머신을 포함하지만, 이 분야 기술에 익숙한 자는 이 기능이 ROM 안에 고정 프로그램을 갖는 마이크로 컨트롤러 또는 프로그래밍 가능한 마이크로프로세서에 의해 수행될 수도 있다는 것을 알 수 있을 것이다.

제10도는 외부 메모리 인터페이스(302)에 대한 방법 실시예를 정의하는 완전 상태 다이어그램을 보여준다. 매우 큰 수의 상태 및 상태 천이가 존재하긴 하지만, 그들의 시퀀스는 기본적으로 수행되는 메모리 액세스를 위해 선택된 칼럼 타이밍에 의존한다. 부가적으로, 상태는 로우 시간 상태와 칼럼 시간 상태의 두 그룹으로 나누어진다.

로우 시간 상태는 각각의 메모리 액세스의 어드레스 서브사이클 또는 " 로우 시간" 을 구성한다. 외부 메모리 인터페이스(302)는 새 페이지 액세스가 시작될 때마다, 이 상태들로 진입하게 된다. 어드레스되는 메모리 타입이 결정되는 것은 이 상태들 동안이 된다. 매 로우 액세스마다 최소한 4개의 로우 상태가 발생할 것이다. 그들은 다음과 같이 정의된다.

모든 메모리 액세스에 대한 시작 상태는 r1 상태(370)이다. r1 상태(370) 동안, 전송 제어기(80)는 로우 어드레스 A[31:0]과 사이클 타입 STATUS[4:0]을 출력하고, 모든 제어 신호를 그들의 비활성 상태로 구동한다.

모든 메모리 액세스에 공통적인 다음 상태는 r2 상태(371)이다. r2 상태(371)동안, 전송 제어기(80)는 $\overline{RL}$을 하이 상태로 하고, 데이터 전송 방향에 따라$\overline{DDIN}$ 을 구동한다. r2 상태(371) 동안, 전송 제어기(80)는 어드레스 시프트 선택 AS[2:0], 버스 크기 선택 BS[1:0], 칼럼 타이밍 선택 CT[1:0], 페이지 크기 선택 PS[2:0] 및 $\overline{UTIME}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 블록 기입과 로드 칼라 레지스터를 포함하는 판독과 기입을 위해 r3 상태(372)로 진입한다. r3 상태(372)동안, 전송 제어기 (80)는 $\overline{DBEN}$, DSF1, DSF2, $\overline{TRG}$ 및$\overline{WE}$를 그들의 활성 로우 시간 레벨로 구동하며,$\overline{FAULT}$, READY 및$\overline{RETRY}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 리프레시만을 위해 rf3 상태(373)로 진입한다. rf3 상태(373) 동안, 전송 제어기(80)는 모든$\overline{CAS}$ [7:0] 스트로브를 표명하고,$\overline{FAULT}$, READY 및$\overline{RETRY}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 시프트 레지스터 전송 사이클만을 위해 tr3 상태(374)로 진입한다. tr3 상태(374)는 r3(372)과 외부적으로 동일하다.

외부 메모리 인터페이스(302)는 칼럼 타이밍 선택 CT[1:0] = " 11" 일 때 3 사이클/칼럼 액세스 동안만 r4 상태(375)로 진입한다. 어떠한 신호 천이도 발생하지 않는다. 전송 제어기(80)는 $\overline{RETRY}$ 입력을 샘플링한다.

r5 상태(376)는 칼럼 타이밍 선택 CT[1] = " 1" 일 때 2와 3 사이클/칼럼 액세스에 공통적이다. r5 상태(376) 동안, 전송 제어기(80)는 $\overline{RAS}$ 출력을 로우 상태로 구동하며 $\overline{RETRY}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 리프레시를 제외한 모든 액세스에 대해 r6 상태(377)로 진입한다, r6 상태(377) 동안, 전송 제어기(80)는 $\overline{RAS}$ 가 로우 상태가 아닌 경우 로우 상태로 구동하고, $\overline{DBEN}$, DSF1, DSF2, $\overline{TRG}$ 및 $\overline{WE}$ 를 그들의 적절한 칼럼 시간 레벨로 구동한다. 전송 제어기(80)는 READY 및 $\overline{RETRY}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 리프레시 사이클에 대해서만 rf6 상태(378)에 진입한다. rf6 상태(378) 동안, 전송 제어기(80)는 $\overline{RAS}$ 를 로우 상태로 구동하고, $\overline{RETRY}$ 입력을 샘플링한다.

rf7 상태(379)는 칼럼 타이밍 선택 CT[1] = " 1" 일 때 2 및 3 사이클/칼럼 리프레시에 공통적이다. rf7 상태(379) 동안, 전송 제어기(80)는 STATUS[4:0]에 프로세서 활동 코드(processor activity code)를 출력하고 $\overline{RETRY}$ 입력을 샘플링한다.

외부 메모리 인터페이스(302)는 칼럼 타이밍 선택 CT[1:0] = " 11" 일 때, 3 사이클/칼럼 리프레시 동안만 rf8 상태(380)에 진입한다. 어떠한 신호 천이도 발생하지 않는다. 전송 제어기(80)는 $\overline{RETRY}$ 입력을 샘플링한다.

rf9 상태(381)는 모든 리프레시에 대한 최종 상태이다. rf9 상태(381)동안, 전송 제어기(80)는 STATUS[4:0] 상에 프로세서 활동 코드를 출력하고, $\overline{RETRY}$ 입력을 샘플링한다.

rhiz 상태(382)는 고 임피던스 상태이다. 외부 메모리 인터페이스(302)는 외부 버스 호스트 요청동안 상태(382)로 진입한다. 외부 메모리 인터페이스(302)는 버스 릴리스(bus release)를 통한 영상 시스템의 복귀 때까지 상태(382)를 반복한다.

칼럼 시간 상태는 각각의 메모리 액세스의 데이터 서브사이클 또는 " 칼럼 시간" 을 포함한다. 모든 데이터 전송은 이 상태동안 일어난다. 외부 메모리 인터페이스(302)는 칼럼 상태의 두 가지 기초 시퀀스로 들어간다. 이 시퀀스들은 파이프라인된 그리고 파이프라인되지 않은 1 사이클/칼럼 액세스에 대한 것과 2 및 3 사이클/칼럼 액세스에 대한 것이다. 페이지 모드 동작동안, 멀티 칼럼 액세스가 발생하여, 외부 메모리 인터페이스(302)가 단일 페이지 모드 액세스 과정동안 이 상태 시퀀스를 여러 번 반복할 수 있게 한다.

c1,2 상태(383)는 칼럼 타이밍 선택 CT[1:0] = " 10" 일 때 2 사이아클/칼럼 액세스에 대한 제1 칼럼 상태가 된다. c1,2 상태(383)동안, 전송 제어기(80)는 어드레스 버스 A[31:0]에 칼럼 어드레스를 출력하고, STATUS[4:0]에 프로세서 활동 코드를 출력한다. 전송 제어기(80)는, 이 때가 기입 사이클이면 데이터 출력을 구동하며, 적절한 $\overline{CAS}$ [7:0] 스트로브를 표명(assert)한다.

c1,3 상태(384)는 칼럼 타이밍 선택 CT[1:0] = " 11" 일 때 3 사이클/칼럼 액세스에 대한 제1 칼럼 상태이다. c1,3 상태(384)동안, 전송 제어기(80)는 어드레스 버스 A[31:0]에 칼럼 어드레스를 출력하고, STATUS[4:0]에 프로세서 활동 코드를 출력한다. 그리고 이 때가 기입 사이클이면 데이터 출력을 구동한다.

c2,3 상태(385)는 칼럼 타이밍 선택 CT[1:0] = " 11" 일 때 2 및 3 사이클/칼럼 액세스에 대한 제2 칼럼 상태이다. c2,3 상태(385)동안, 전송 제어기(80)는 적절한 $\overline{CAS}$ [7:0] 스트로브를 표명한다.

c3 상태(386)는 칼럼 타이밍 선택 CT[1]=" 1" 일 때 2 및 3 사이클/칼럼 액세스에 대한 최종 칼럼 상태가 된다. c3 상태(386)동안, 전송 제어기(80)는 READY 입력을 샘플링하고, 현재 사이클이 판독 사이클이면 입력 데이터를 래치한다.

cs1 상태(387)는 칼럼 타이밍 선택 CT[1] = " 0" 일 때 1 사이클/칼럼 판독에 대한 시작 칼럼 상태이다. cs1 상태(387)동안, 전송 제어기(80)는 어드레스 버스 A[31:0] 상에 칼럼 어드레스를 출력하고, STATUS[4:0] 상에 프로세서 활동 코드를 출력하며, 모든$\overline{CAS}$ [7:0] 스트로브를 표명한다. 전송 프로세서(80)는 입력 데이터를 래치하지 않는다.

cs2 상태(388)는 제2 시작 칼럼 상태이다. 외부 메모리 인터페이스(302)는 칼럼 타이밍 선택 CT[1:0] = " 00" 일 때만 파이프라인된 1 사이클/칼럼 판독에 대해 cs1 상태(387)에 후속하는 cs2 상태(388)를 삽입한다. cs2 상태(388)동안, 전송 제어기(80)는 다음 칼럼 액세스의 칼럼 어드레스와 프로세서 활동 코드를 출력한다. 전송 제어기(80)는 상태(388) 동안 입력 데이터를 래치하지 않는다.

cp 상태(389)는 칼럼 타이밍 선택 CT[1] = " 0" 일 때 모든 1 사이클/칼럼 액세스에 대한 주 칼럼 상태이다. 외부 메모리 인터페이스(302)는 요청되는 시작 상태의 완결 후에 각각의 페이지 모드 칼럼 액세스에 대해 상태(389)를 반복한다. cp 상태(389)동안, 전송 제어기(80)는 어드레스 버스 A[31:0] 상에 칼럼 어드레스를 출력하고, STATUS[4:0]에 프로세서 활동 코드를 출력하며, 적절한$\overline{CAS}$ [7:0] 스트로브를 표명한다. 전송 제어기(80)는 유효 데이터를 기입하기 위해 구동하고, 유효 데이터의 판독을 위해 래치한다.

cdr 상태(390)는 칼럼 타이밍 선택 CT[1] = " 0" 일 때 1 사이클/칼럼 액세스에 대한 판독 드레인 상태이다. 외부 메모리 인터페이스(302)는, 페이지 변화 또는 파이프라인 버블(pipeline bubble)로 인해, 또는 인터페이스가 아이들 상태로 되는 것으로 인해 다른 칼럼 액세스가 보류 중이지 않은 경우, 최종 cp 상태(389) 또는 cs1 상태(387) 후에 cdr 상태(390)를 갖는다. cdr 상태(390)동안, 전송 제어기(80)는 $\overline{CAS}$ [7:0] 스트로브를 표명하지 않지만, 선행하는 cp 상태(389) 또는 cs1 상태(387)에 의해 어드레스된 입력 데이터를 래치한다.

외부 메모리 인터페이스(302)는 칼럼 타이밍 선택 CT[1:0] = "00일 때 파이프라인된 1 사이클/칼럼 기입 드레인 사이클에 대한 cdw 상태(391)에 진입한다. 외부 메모리 인터페이스(302)는 새로운 페이지의 시작에서 r1 상태(370)로 복귀하기 직전에 cdw 상태(391)에 진입한다. cdw 상태(391)동안, 전송 제어기(80)는 모든 $\overline{CAS}$ [7:0] 스트로브를 표명하며, 기입된 최종 데이터로 파이프라인된 메모리 어레이를 갱신한다.

ci 상태(392)는 아이들 칼럼 상태이다. 외부 메모리 인터페이스(302)는 임의의 요청된 드레인 상태 후에 더 이상의 칼럼 액세스가 보류 중이지 않을 때 ci 상태(392)에 진입한다. 이는 파이프라인 버블로 인한 것일 수도 있고, 더 이상 사이클이 요청되지 않지만 페이지 변경이 발생하지 않아서 인터페이스가 페이지 모드로 계속 남아 있기 때문일 수도 있다.

제10도에 도시된 상태 천이 인디케이터는 어떤 상태 또는 사건이 다른 상태로의 천이를 일으키는지 결정한다. 어떤 경우, 어느 상태로의 천이가 발생하지 않도록 하기 위해서는 여러 가지 조건이 만족되어야 한다. 이런 상태 천이 조건과 사건의 리스트는 다음과 같다.

CT = "XX" - r3 상태(372) 동안에 래치된대로 표지된 칼럼 타이밍 선택 CT[1:0]에 대해 상태 변화가 발생한다;

r, 판독 - 판독 사이클이 발생한다;

w, 기입 - 기입 사이클이 발생한다;

rt - 판독 전송 사이클이 발생한다;

wt - 기입 전송 사이클이 발생한다;

fault - r3 상태(372) 동안에 로우(low) 샘플링된 $\overline{FAULT}$ 입력은 메모리 액세스 폴트를 표지한다;

retry - r3 상태(372) 동안에 로우 샘플된 $\overline{RETRY}$ 입력은 로우-시간 재시도를 요청한다;

wait - r3 상태(372), r6 상태(377) 또는 c3 상태(386) 동안 로우 샘플된 READY 입력은 현재 상태의 반복을 요청한다;

spin - 전송 제어기(80)의 파이프라인이 로드할 수 있게 하는 추가의 r6 상태(377)를 추가하도록 내부적으로 발생된 요청이다. 외부 메모리 인터페이스(302)는 버스 크기가 이전 페이지 액세스로부터 변경되었을 때 각각의 2 사이클/칼럼 기입동안 1회, 각각의 1 사이클/칼럼 기입동안 2회, 1 사이클/칼럼 액세스 동안 1회 만큼 상태(377)에 진입한다.

new co1 - 현재 메모리 사이클은 동일한 메모리 페이지의 다음 칼럼 액세스가 된다;

drain - 현재 메모리 사이클은 1 사이클/칼럼 판독 사이클이다. 현재 판독 상태에 후속하여 보류 중인 칼럼 액세스는 없다.

idle - 현재 상태 후에 보류 중인 칼럼 액세스는 없다. 모든 드레인 상태는 완결되었다; 및

new page - 다음 메모리 액세스는 페이지 변경과 새로운 로우 액세스(row access)를 요청한다.

전송 제어기(80)는 내부 메모리 인터페이스(301)와 외부 메모리 인터페이스(302) 모두에서 파이프라인을 포함한다. 이 파이프라인은 현재 액세스가 아직 완결되지 않았을 때 전송 제어기(80)가 요청하는 메모리 액세스를 대기할 수 있다. 예를 들어 패킷 요청이 온-칩에서 오프칩 메모리로 데이터를 전송하는 것인 경우, 데스티네이션은 액세스당 2 사이클을 요청할 것이다. 요청되는 데이터가 패킷 전송 FIFO 버퍼(311) 안에 있다고 가정하면, 데이터는 액세스당 1 사이클에서 패킷 전송 FIFO 버퍼(311)로부터 추출될 것이므로, 다른 데스티네이션 사이클은 제1 사이클이 완결되기 전에 파이프라인 내에 배치되어야 한다.

이 파이프라인은 정상적으로는 사용자에게 완전히 투명(transparent)하다. 동작에 대한 효과는 파이프라인이 " 드레인되었을 때" 에만 알 수 있다. 사이클이 외부 메모리 인터페이스(302)의 외부 메모리 파이프라인으로 로드되기 위해서는, 파이프라인에 이미 포함된 임의의 다른 사이클과 동일한 메모리 페이지 내에 위치해 있어야 한다. 일단, 사이클이 파이프라인 내에 배치되면 그 사이클은 제거될 수 없다. 즉, 그 사이클은 반드시 발생한다. 따라서, 새로운 메모리 페이지에 대한 액세스가 요청되면, 파이프라인 내의 사이클은 우선 완결되어야 한다. 전송 제어기(80)가 외부 메모리로의 패킷 전송을 수행 중이고, 프레임 제어기(90)가 메모리 요청을 발행하면, 현재 파이프라인 내에 있는 임의의 칼럼 사이클은, 프레임 제어기 사이클이 더 높은 우선 순위를 갖고 있다하더라도, 프레임 제어기(90)가 요청하는 사이클이 발생하기 전에 완결되어야만 한다. 파이프라인 드레이닝(pipeline draining)은 캐시 및 외부 메모리 액세스, 호스트 액세스, 및 긴급 리프레시 전에, 패킷 전송 중단 동안, 및 " 칼럼 시간" 재시도 후에도 발생할 수 있다. 파이프라인 드레이닝되는 것은 긴급 요청이 보류 중일 때 가장 높은 우선 순위가 주어진다.

외부 메모리 액세스동안, 활성 칼럼 액세스도 발생하지 않는 비활성 사이클(inactive cycle)이 있을 수 있다. 이 사이클들을 전송 제어기(80)의 내부 파이프라인 내에서의 " 버블(bubble)" 로 인한 것이다. 버블은 전송 제어기(80)가 사이클 동안 수행하는 동작이 전혀 없을 때 발생한다. 이는, 크로스바 경쟁의 결과로; 전송 제어기(80)가 기입 사이클 동안 다음 액세스를 수행할 충분한 데이터를 갖고 있지 않기 때문에; 패킷 전송 FIFO 버퍼(311)가 가득 차서 더 이상의 데이터가 판독 사이클에 로드되지 못할 때; 또는 단순히 요청되는 활동이 없기 때문에 발생할 수 있다. 이러한 상황들이 발생하지 않으면, 버블은 전혀 발생하지 않는다.

버블의 일례는 전송 제어기(80)가 내부 소스와 외부 데스티네이션 간에 패킷 전송을 서비스할 때에 발생하는 것이다. 전송 제어기(80)가 한번에 1 바이트씩 소스 데이터를 액세스할 수 있고 한번에 8 바이트씩 데스티네이션 데이터를 기입할 수 있도록 패킷 전송이 설정되는 경우, 전송 제어기(80)는 8번의 내부 사이클마다 1회의 데스티네이션 액세스를 수행할 수 있을 것이다. 이는 전송 제어기가 다음 칼럼 액세스를 수행하기 위해 8 바이트의 충분한 데이터를 갖기 위해 기다리는 동안, 버블을 외부 메모리 데스티네이션 사이클로 삽입하는 결과로 나타난다. 삽입된 버블 사이클의 수는 외부 메모리 사이클의 타이밍에 의존한다.

광범위한 메모리 타입 및 조직과 주변 기기를 지원하기 위해, 멀티프로세서 집적 회로(100)는, 확장, 재시도 및 외부 메모리 사이클의 종결에 대해 다양한 방법을 제공한다. 이런 작용의 각각은 다음 절에서 자세히 설명된다.

일단, 전송 제어기(80)가 파이프라인 내의 보류 중인 모든 칼럼 액세스를 완결하면, 메모리 액세스를 종결시킬 준비를 한다. 그러나 새로운 로우 액세스가 요청될 때까지는 종결은 발생하지 않을 것이다. 외부 메모리 신호는 다음 메모리 액세스까지 ci 상태(392) 내에서 활성으로 남아있을 것이다. 이는 DRAM 장치가 페이지-모드 상태를 유지하도록 만들어 준다. 다음 어드레스가 이전 메모리 액세스와 동일한 메모리 페이지 내에 있고 동일한 방향인 경우, 로우 액세스 사이클은 필요하지 않다. 다음 메모리 액세스가 로우 액세스를 요청한다면, 현재 페이지 모드 액세스는 종결되고 새로운 로우 액세스가 시작될 것이다. 다른 외부 버스 활동이 거의 없다면, 대부분의 메모리 사이클은 새로운 로우 액세스의 발생을 요청하는 트리클 리프레시 사이클의 발생에 의한 그들의 최종 칼럼 액세스의 완결 직후에 곧 종결될 것이다.

멀티프로세서 집적 회로(100)는 메모리 사이클 시간을 확장하는 대기 상태 삽입을 지원한다. 이는 READY 입력을 사용해 수행된다. READY 입력은 CLKOUT의 하강 에지 상의 적절한 시간에 다시 샘플링된다. READY가 하이 상태로 샘플링되는 경우, 사이클은 정상적인 방식으로 계속될 것이다. READY가 로우 상태로 샘플링되는 경우, 현재의 머신 상태가 반복되고 다음의 CLKOUT의 하강 에지에서 READY가 다시 샘플링된다. 메모리 사이클은, 전송 제어기(80)가 READY 하이를 샘플링할 때까지 현재 상태를 반복한다.

전송 제어기(80)는 다수의 대기 상태가 묶고 있는 메모리 액세스를 종결시키는 데에 필요한 시간-아웃 또는 포기 메커니즘을 갖지 않는다. 합리적인 시간 내에 종결될 수 없는 메모리 액세스는 서비스되기를 기다리고 있을 수 있는 높은 우선 순위의 외부 메모리 액세스를 로크-아웃(lock-out)하는 것을 방지하기 위해 폴트되거나 재시도되어야만 한다.

임의의 메모리 사이클 타입에 대해서, 로우 시간동안 대기 상태가 부가될 수 있다. 대기 상태는, 칼럼 액세스당 2 또는 3 사이클 모두에 대해서 칼럼 시간에 삽입될 수 있다. 대기 상태는 $\overline{RAS}$ 의 하강 전후에, 로우 시간 동안 두 위치에서 삽입될 수 있다. 전송 제어기(80)는 먼저 r3 상태(372), 또는 rf3 상태(373) 또는 tr3 상태(374)의 시작에서 READY 신호를 샘플링한다. 이는, 로우 어드레스 및/또는 사이클 타입을 디코딩하고, 어드레스된 장치가 $\overline{RAS}$ 가 하강 전에 부가의 액세스 시간을 필요로 하는지를 결정할 시간을 허용한다. r3 상태(372)는 전송 제어기(80)가 READY 하이를 샘플링할 때까지 반복될 것이다. READY가 로우 샘플링되었을 때 r3 상태(372)가 반복되기는 하지만, $\overline{FAULT}$ 와 $\overline{RETRY}$ 입력은 리샘플링되지 않는다. 액세스는 임의의 대기 상태가 삽입되기 전에 제1 r3 상태(372) 사이클동안 폴트되거나 재시도되어야만 한다. READY 입력은 r6 상태(377) 동안 다시 샘플링된다. 이는 추가적인 $\overline{RAS}$ 액세스 시간을 필요로 할 수도 있는 장치가 그러한 시간을 생성할 수 있게 하는 $\overline{RAS}$ 의 하강 후에 발생한다. READY가 로우 샘플링되면, r6 상태(377)는 READY가 하이 샘플링될 때까지 반복될 것이다.

전송 제어기(80)는, 버스 크기를 변화시키는 2 및 3 사이클 기입과 1 사이클/칼럼 액세스동안 추가의 r6 상태(377)를 자동적으로 삽입한다. 전송 제어기(80)는 각각의 r6 상태(377) 동안에 정규 방식으로 READY를 샘플링한다. 그래서 로우 샘플링된 READY로부터 유발된 임의의 추가적인 r6 상태(377)는 전송 제어기(80)에 의해 이미 삽입된 것들에 부가될 것이다.

칼럼 시간 대기 상태는 칼럼 액세스당 1 사이클에 대해 지원되지 않으며, READY는 이런 사이클에 대한 r6 상태(377)를 넘어서서 샘플링되지 않는다. 칼럼 액세스당 2 및 3 사이클동안, 전송 제어기(80)는 C3 상태(386) 동안 READY 핀을 샘플링한다. 이는 칼럼 어드레스 및 $\overline{CAS}$ 의 하강 후에 발생하며, 이를 요청하는 장치에 대해 확장된 $\overline{CAS}$ 액세스 시간을 허용해 준다. READY가 하이 샘플링되면, C3 상태(386)는 칼럼 액세스를 완결한다. 그렇지 않다면 C3 상태(386)는 반복되고 전송 제어기(80)는 READY가 하이 샘플링될 때까지 각각의 순차적인 CLKOUT 하강 에지에서 READY를 샘플링한다. 멀티프로세서 집적 회로(100)는 두 타입의 재시도 메커니즘 -" 로우 시간" 재시도 그리고 " 칼럼 시간" 재시도- 을 지원한다. 발생하는 재시도 타입은 메모리 액세스의 어떤 시점에서 멀티프로세서 집적 회로(100)의 $\overline{RETRY}$ 입력이 활성 로우로 구동되는지에 의존한다.

로우 시간 재시도는, 외부 논리가 다중 프로세서 집적 회로(100)에게 현재 액세스가 완결되지 않았을 수도 있고 재시도할 필요가 있다는 것을 알려주는 방법을 제공한다. 로우 시간 재시도는, r3 상태(372)의 시작에서 첫번째로 샘플링되었을 때, $\overline{RETRY}$ 입력을 로우 상태로 구동함으로써 발생된다. $\overline{RETRY}$ 가 로우 샘플링되면, 전송 제어기(80)는 r3 상태(372)의 끝에서 현재 액세스를 종결지을 것이다. 더 높은 우선 순위의 요청이 보류 중이지 않다면, 재시도된 액세스는 새로운 r1 상태(370)와 함께 즉시 재시작할 것이다. 이 경우, 재시도된 사이클은 더 높은 우선 순위의 요청이 서비스되어 끝날 때까지 다시 시작되지 않을 것이다.

로우 시간 재시도가 패킷 전송 사이클동안 발생되고, 동일하거나 더 높은 우선 순위의 패킷 전송 요청이 보류 중인 경우, PTMIN에 의해 정의되는 시간 간격이 만료됐다고 가정할 때, 재시도가 발생한 패킷 전송은 중지될 것이다. 전송 제어기(80)는 우선 순위/라운드 로빈 체인 내의 순번에 다시 도달하여, 패킷 전송이 재개될 때까지, 중지된 패킷 전송을 재시도하지 않을 것이다.

칼럼 시간 재시도는 통상적 의미에서의 재시도는 전혀 아니다. 대신에, " 로우 액세스 요청" 으로 생각될 수 있다. 칼럼 재시도가 발생하는 경우, 전송 제어기(80)는 파이프라인 내의 보류 중인 모든 칼럼 액세스를 완결한 후, 로우 액세스로써 다음 액세스를 시작한다. 따라서, 다수의 칼럼 액세스가 재시도가 요청된 후에 발생할 것이다. 이 칼럼 액세스는 로우 액세스가 재시작된 후에는 반복되지 않을 것이다. 칼럼 시간 재시도는, r3 상태(372) 후에 $\overline{RETRY}$ 입력이 로우 샘플링될 때마다 발생한다. 전송 제어기(80)는 각각의 CLKOUT 하강 에지에서 r3 상태(372)에 후속하는 각각의 상태 끝에서 $\overline{RETRY}$ 를 샘플링하며 각각의 샘플링 주기동안 유효한 하이 또는 로우 레벨에 있어야만 한다. 전송 제어기(80)가 $\overline{RETRY}$ 를 로우로 샘플링하는 경우, 현재의 페이지 모드는 현재 전송 제어기(80) 파이프라인 내에 있는 모든 칼럼 액세스가 완결되자마자 종결될 것이다. 일단 재시도가 요청되면, $\overline{RETRY}$ 입력 상의 입력값은 더 이상의 효과가 없다. 그러나, 전송 제어기는 각각의 남아있는 칼럼 액세스 동안 $\overline{RETRY}$ 를 계속 샘플링할 것이며, 따라서 $\overline{RETRY}$ 는 각각의 샘플링 주기동안 유효한 하이 또는 로우 레벨로 유지되어야만 한다. 정규 동작은, 시스템이 $\overline{RETRY}$ 를 로우로 구동하고 현재의 로우 액세스가 끝날 때까지 $\overline{RETRY}$ 를 로우 레벨로 유지하게 한다.

외부 메모리 인터페이스(302) 파이프라인이 로딩되는 방법 때문에, 판독 사이클 동안 $\overline{RETRY}$ 를 r4 상태(375) 또는 r5 상태(376) 내에서 표명하는 것은 아무런 효과가 없다. $\overline{RETRY}$ 입력은 계속 샘플링되어야 하며 유효한 하이 또는 로우 레벨에 있어야 한다. 이런 이유로 인해, 칼럼 시간 재시도를 원한다면, $\overline{RETRY}$ 는 로우 액세스가 끝날 때까지 로우로 표명되어야 한다. 프레임 제어기(90) VRAM 액세스 사이클과 리프레시 사이클은, 본질상, 새로운 로우 액세스가 항상 그들의 단일 칼럼 액세스에 후속하기 때문에, 칼럼 시간 재시도에 의해 영향받지 않는다.

시스템은, 시스템 에러로 인해 메모리 액세스를 완결할 수 없는 경우, 메모리 사이클을 폴트함으로써 멀티프로세서 집적 회로(100)에 통지한다. 이는 메모리 액세스가 재시도되기 전에 마스터 프로세서(60)가 에러를 정정할 수 있게 한다. 메모리 폴트는 로우 시간에서만 발생할 수 있으며 r3 상태(372)의 시작에서 $\overline{FAULT}$ 입력 로우를 구동함으로써 시발된다. 전송 제어기(80)는 메모리 사이클의 임의의 다른 부분동안 $\overline{FAULT}$ 를 샘플링한다. 폴트 메커니즘(faulting mechanism)은 폴트를 유발하는 액세스 타입에 따라 어느 정도 변화한다. 이는 다음에 더 자세히 설명된다. 여러 타입의 사이클 요청에 대한 메모리 폴트는 다음과 같이 지원된다;

프레임 제어기(90) 사이클 - 프레임 제어기(90) 사이클 동안 폴트는 지원되지 않으며 $\overline{FAULT}$ 핀은 무시된다;

리프레시 사이클 - 폴트는 리프레시 사이클 동안 지원되지 않으며 $\overline{FAULT}$ 핀은 무시된다;

디지털 영상/그래픽 프로세서 캐시와 직접 메모리 액세스 요청 - 폴트는 이 사이클 동안 지원된다. 요청 프로세서는 폴트가 제거될 때까지 그 요청이 완결되는 것을 보지 못한다. 다른 디지털 영상/그래픽 프로세서 (71, 72, 73과 74)로부터의 요청은 계속 서비스될 것이다.

마스터 프로세서 명령 또는 데이터 캐시 요청 - 지원되는 폴트는 이 사이클 동안 지원된다. 폴트된 캐시 요청은 즉시 취소되며 마스터 프로세서(60)는 인터럽트된다. 다른 캐시는 서비스되는 보류 중인 요청을 계속 서비스할 수 있을 것이다;

마스터 프로세서(60) 직접 메모리 액세스 요청 - 폴트는 이 사이클 동안 지원된다. 직접 외부 액세스 요청은 즉시 취소되며 마스터 프로세서(60)는 인터럽트 된다; 그리고

패킷 전송 - 폴트는 패킷 전송동안 지원된다. 패킷 전송은 중지되며 그 상태는 요청 중인 프로세서의 파라메타 메모리에 세이브된다. 다른 프로세서로부터의 패킷 전송 요청은 계속 서비스될 수 있다. 전송 제어기(80)의 내부 패킷 전송 상태는 세이브된 파라메타 내에 포함된다. 패킷 전송이 외부 대 외부 메모리 (external to external memory)에서 온 것인 경우, 외부 대 외부 전송 버퍼의 상태도 세이브된다. 버퍼 자신은 변경되지 않는다. 세이브된 패킷 전송 선택 필드 내의 패킷 전송 상태 비트는, 폴트가 소스 전송에서 발생하였는지 또는 데스티네이션 전송에서 발생하였는지를 나타내도록 설정될 것이다.

일단 파리미터가 세이브되면, 전송 제어기(80)는 마스터 프로세서(60)의 FLTSTS 레지스터 내에 적절한 비트를 설정하여, 패킷 전송이 폴트된 프로세서를 표시한다. FLTSTS 레지스터의 엔코딩은 표 13에 나타나 있다. FLTSTS 레지스터 내에서의 프로세서 비트 설정은 또한 마스터 프로세서(60)에 패킷 전송 폴트 인터럽트를 발생시킨다. 마스터 프로세서(60)는 FLTSTS 레지스터를 판독하여, 어느 프로세서가 폴트된 패킷 전송을 요청했는지 찾아낼 수 있다. 일단 프로세서가 식별되기만 하면, 마스터 프로세서(60)는 중지된 패킷 전송 파라메타를 검사하여, 폴트를 발생시키는 메모리 액세스를 결정할 수 있다. 디지털 영상/그래픽 프로세서(71, 72, 73 또는 74)는 폴트가 자신의 패킷 전송 동안에 발생했는지를 알지 못할 것이다. 다만 패킷 전송이 아직 완결되지 않았다는 것을 알뿐이다. 폴트를 정정하거나 요청 중인 디지털 영상/그래픽 프로세서(71, 72, 73 또는 74)가 패킷 전송 요청을 취소하도록 요청하는 것은 마스터 프로세서(60)의 책임이다.

마스터 프로세서(60)가 폴트를 정정할 수 있는 경우, 마스터 프로세서(60)는 FLTSTS 레지스터 내의 비트를 클리어할 수 있으며, 패킷 전송 요청은 자동적으로 다시 제공될 것이다. 일단 폴트된 패킷 전송이 라운드 로빈 우선 순위 스킴 내에서 자신의 순번을 접수하면, 그 내부 상태는 파라메타 메모리 내의 세이브된 파라메타로부터 복원되며, 전송 제어기(80)는 폴트된 액세스에서 패킷 전송을 지속할 것이다.

폴트가 디지털 영상/그래픽 프로세서 요청 캐시 서비스 또는 직접 외부 액세스 요청 동안에 발생하는 경우, 폴트가 발생한 어드레스는 요청 프로세서의 파라메타 메모리의 캐시 폴트 어드레스 위치 내에 세이브된다. 마스터 프로세서(60)의 FLTSTS 레지스터 내의 적절한 비트는, 인터럽트가 마스터 프로세서(60)에게 보내지도록 설정된다. 그 후, 마스터 프로세서(60)는 파라메타 메모리를 검색하여, 폴트된 어드레스를 결정할 수 있다. 마스터 프로세서(60)가 폴트를 교정할 수 있는 경우, 마스터 프로세서는 FLTSTS 레지스터 내의 비트를 클리어할 수 있으며 요청은 다시 스케줄되어야 한다. 폴트가 교정될 수 없고, 마스터 프로세서(60)가, 요청 디지털 영상/그래픽 프로세서가 키시-미스 또는 직접 외부 액세스 요청을 포기하도록 해야할 필요가 있는 경우, 디지탈 영상/그래픽 프로세서에 재설정 요청을 보내야만 한다. 이렇게 하면 작업이 포기될 것이다.

디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)는 캐시 또는 직접 외부 액세스 요청 동안 폴트가 일어났는지는 알지 못할 것이다. 그들은 단지 그 요청들이 아직 완결되지 않았다는 것만을 알뿐이다. 폴트를 정정하거나 요청 디지탈 영상/그래픽 프로세서를 재설정하는 것은 마스터 프로세서(60)의 책임이다.

마스터 프로세서(60)가 요청한 캐시 필 영역 또는 직접 외부 액세스 사이클이 폴트된 경우, 그 요청은 즉시 취소되고 마스터 프로세서(60)는 메모리 폴트 인터럽트를 받는다. 이 메모리 폴트 인터럽트는 폴트가 데이터 캐시 폴트인지 아니면 명령 캐시 폴트인지를 표지해준다. 데이터 캐시 폴트가 발생한 경우, 어드레스는 세이브되고 데이터도 세이브된다.

온칩 어드레스의 임의의 액세스는 $\overline{FAULT}$ 입력에 무관한 폴트를 유발할 수 있다. 이는 비규칙 온칩 액세스가 시도될 때 발생한다. 시도되는 사이클에 대한 정규 폴트 메카니즘이 적용된다. 온칩 폴트는 다음과 같다:

Hex "02000000" 하에서의 임의의 어드레스로/로부터 디지털 영상/그래픽 프로세서 패킷 전송 - 이것은 데이터 메모리 또는 디지탈 영상/그래픽 프로세서 파라메타 메모리가 아님 -;

Hex "02000000" 하에서의 임의의 어드레스로/로부터의 디지탈 영상/그래픽 프로세서 캐시 서비스, 마스터 프로세서(60) 캐시 서비스 또는 직접 외부 액세스 - 이들은 데이터 메모리 또는 디지탈 영상/그래픽 프로세서 파라메타 메모리가 아님 -; 및

Hex "02000000" 하에서의 임의의 어드레스로/로부터의 마스터 프로세서(60) 패킷 전송 - 이것은 데이터 메모리 또는 디지탈 영상/그래픽 프로세서 파라메타 메모리가 아님 -.

$\overline{UTIME}$ 입력은 멀티프로세서 집적 회로(100)가 제공하는 것과는 다른 메모리 타이밍을 발생시킬 필요가 있는 사용자를 지원하기 위해 제공된다.$\overline{UTIME}$ 이 로우 시간(row time)에 로우 샘플링되면, $\overline{RAS}$ 와$\overline{CAS}$ [7:0]의 출력 타이밍은 현재 페이지의 나머지 부분에 대해 변경된다.$\overline{RAS}$ 신호는 칼럼 액세스가 시작되었음을 표시하도록 변경된다. 따라서 사용자 시간에 맞춘$\overline{CAS}$ 신호는$\overline{RAS}$ 하강에 의해 트리거될 수 있다.$\overline{RAS}$ 는 실제 칼럼 액세스가 시작되어 전송 제어기(80)의 파이프라인 내의 임의의 버블이 파악될 때의 머신 상태에 대해서만 활성 로우 상태로 표명될 수 있을 것이다. 칼럼 어드레스 스트로브$\overline{CAS}$ [7:0]은 칼럼 어드레스와 동시에 출력된다. 이는 어느 바이트가 액세스되고 있는지를 표시하는 데에 이들이 이용될 수 있게 하여, 외부$\overline{CAS}$ 발생을 더 쉽게 해준다.$\overline{RAS}$ 의 타이밍이 변경됐기 때문에, 외부 논리는 필요한 경우, 그 자신의$\overline{RAS}$ 타이밍을 발생시켜야만 한다. 외부로 발생되는$\overline{RAS}$ 는 RL의 하강 에지에 의해 트리거된다. 사이클 시작점에서 출력되는 상태 코드는$\overline{TRG}$,$\overline{WE}$ 등의 메모리 타이밍을 발생시키기 위해 필요한 모든 정보를 제공한다는 것을 주의하라. 이 출력의 타이밍은 사용자 시간에 맞춘 액세스 동안 변경되지 않는다.

전송 제어기(80)의 내부 메모리 인터페이스(301)는 크로스바(50)를 통한 온칩 메모리로의 액세스를 제공한다. 64 비트 내부 데이터 버스는 사이클당 0 - 8 바이트를 전송할 수 있다. 외부 메모리 인터페이스(302)는 모든 오프칩 메모리 및 주변 장치로의 액세스를 제공한다. 외부 메모리 인터페이스(302)는 다양한 메모리 및 주변 장치 타입에 인터페이스하는 데에 필요한 사이클 및 제어 신호를 발생시킨다. 64 비트 외부 데이터 버스는 사이클당 0 - 8 바이트를 전송하고 8, 16, 32, 및 64 비트의 데이터 폭을 가지는 영상 시스템 버스에 접속된 장치를 위하여 동적 버스 크기 지원을 제공할 수 있다.

외부 메모리 인터페이스(302)는, 전송 제어기(80)가 외부 장치와 버스를 공유할 수 있게 하는 단순한 핸드세이크 메카니즘(handshake mechanism)을 포함한다. 핸드세이크 메카니즘은 다음과 같은 4 가지의 동기적 신호 - 호스트 요청 입력$\overline{HREQ}$; 호스트 승인 출력$\overline{HACK}$; 및 내부 요청 출력들 REQ[1:0]을 사용한다.

외부 장치가 영상 시스템 버스를 제어하기 원할 때, 호스트 요청 입력$\overline{HREQ}$ 신호는 멀티프로세서 집적 회로(100)에 신호를 보낸다. 외부 장치가 버스를 점유하고자 할 때에 호스트 요청 입력$\overline{HREQ}$ 를 로우 상태로 한다. 이는 전송 제어기(80)가 수신할 수 있는 최우선 순위의 요청이며, 전송 제어기(80)는 가능한 조속히 버스의 구동을 중단시킬 것이다. 현재 동작이 중단되고 전송 제어기(80)의 파이프라인이 비워지면, 전송 제어기의 버스 점유가 종료된다. 외부 장치는 버스를 원하는 한$\overline{HREQ}$ 를 활성 로우 상태로 계속 유지하여야 한다.$\overline{HREQ}$ 가 비활성 하이 상태일 때마다, 전송 제어기(80)는 버스를 점유하고 구동할 것이다.$\overline{HREQ}$ 입력은 멀티프로세서 집적 회로(100)의 내부 클록에 내부적으로 동기화된다.

전송 제어기(80)가 영상 시스템 버스의 제어를 제공할 준비가 되면, 호스트 승인 출력 $\overline{HACK}$ 신호가 공급된다. 전송 제어기(80)는 활성 $\overline{HREQ}$ 에 이어서 $\overline{HACK}$ 를 로우로 구동하여, 전송 제어기가 그 신호를 고 임피던스로 구동하였고 버스를 포기함을 표시한다. $\overline{HACK}$ 가 액티브 로우인 동안, CLKOUT을 제외한 모든 외부 메모리 인터페이스(302) 버스 출력은 고 임피던스 상태가 된다. 내부 요청 출력 REQ[1:0], $\overline{HACK}$ 및 프레임 제어기(90) 출력은 계속하여 구동된다. 그러면 외부 장치는 필요에 따라 영상 시스템 버스를 구동할 수 있다. 전송 제어기는 $\overline{HREQ}$ 가 비활성임이 검출됨에 이어서 $\overline{HACK}$ 를 비동기적으로 비활성 하이로 구동하고, 버스의 구동을 재개한다.

내부 요청 출력 REQ[1:0]은 전송 제어기(80)에 의하여 수신되는 최우선 순위의 내부 요청의 2 비트 엔코딩을 형성한다. 표 10은 요청 코드 및 그와 연관된 사이클들을 보여준다.

[표 10]

REQ[1:0]	연관된 내부 요청
0 0	낮은 순위 패킷 전송
	티클(tickle) 리프레시
	아이들
0 1	높은 우선 순위의 패킷 전송
1 0	명령 캐시 서비스
	데이터 캐시 서비스
	MP 긴급 패킷 전송
1 1	프레임 제어기 액세스
	긴급 리프레시

외부 논리는 내부 요청 출력 REQ[1:0]을 모니터링하여, 버스를 전송 제어기(80)로 넘겨줄 때를 결정할 수 있다. 호스트 요청은 전송 제어기(80)에 의하여 최우선 순위를 부여받기 때문에, 시스템 설계자는 어느 REQ[1:0] 값이 호스트가 버스 점유를 전송 제어기(80)에 넘겨주도록 하는지를 결정함으로써 외부 장치가 동작하는 레벨을 결정할 수 있다.

전송 제어기(80)는, 외부 메모리 시스템에 의하여 요청되는 대로 자동적으로 DRAM 리프레시 사이클을 발생시키기 위하여 프로그래밍 가능한 리프레시 제어기를 포함한다. 이 리프레시를 위한 파라메타들은 캐시, VRAM 및 리프레시 제어기(310)에 저장된다. 캐시 제어 논리는 마스터 프로세서(60) 및 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)에 의하여 요청되는 대로 캐시 필 및 재기 입을 실행하는 데에 필요한 어드레스들을 발생시킨다. 프레임 제어기(90)의 요청은 VRAM 제어 논리에 의하여 처리된다.

전송 제어기(80)는 패킷 전송을 처리하기 위한 2개의 독립된 제어기를 가진다. 소스 레지스터(321) 및 소스 제어기(322)를 포함하는 소스 머신(320)은 소스 메모리로부터 데이터를 불러오는 데 필요한 어드레스들을 발생시킨다. 패킷 전송 요청이 전송 제어기(80)에 제공될 때, 이는 소스 데이터가 액세스될 방법을 특정하는 다수의 파라메타를 포함한다. 이 파라메타들은 소스 레지스터(322)에 로드되고, 소스 제어 논리(323)를 이용하여 소스 어드레스를 발생시킨다. 유사한 세트의 파라메타들이 데스티네이션 레지스터(341) 내에 로드되고, 데스티네이션 제어 논리(342)에 의해 사용되어, 데스티네이션 메모리 영역에 패킷 데이터를 기입하기 위해 필요한 어드레스를 발생시킨다. 데스티네이션 레지스

터(341) 및 데스티네이션 제어 논리(342)의 조합은 데스티네이션 머신(340)을 형성한다. 소스 머신(320) 및 데스티네이션 머신(340)은 각각 온칩 메모리(10, 20) 및 오프칩 메모리 양자에 각각 어드레스할 수 있다.

제11도는 소스 머신(320)의 구조를 도시한다. 소스 머신(320) 및 데스티네이션 머신(340)의 내부 구성은 동일하다. 다만 전송 제어기(80)의 나머지 부분에 대한 몇가지 접속만이 상이하다.

소스 머신(320)은 다음의 레지스터들을 포함한다. CACHEADR 레지스터(401)는 전송의 방향에 따라 내부 또는 외부 어드레스를 포함한다. 이는 캐시 서브-블록의 어드레스와 함께 전송 제어기(80)의 내부 S 포트로부터 로드된다. 온칩 어드레스의 상위 비트들을 발생시키기 위하여 멀티플렉서가 사용될 수 있다.

ABPITCH 레지스터(402)는 다음 라인의 시작 어드레스를 발생시키기 위하여 시작 어드레스에 가산될(또는 감산될) 피치를 포함한다.

GTPIPE 레지스터(403)는 실제로는 세 부분으로 나뉘어진다. 제1 부분은 ASTART를 위하여 배정된 32 비트 어드레스이다. 제2 부분은 BCOUNT를 위하여 배정된 16 비트 라인-카운트이다. 제3 부분은 ACOUNT를 위하여 배정된 16 비트 바이트-카운트이다. 이는 페치된 가이드 테이블(guide table)이 연관된 레지스터 내에 즉시 저장될 수 없을 때의 가이드 테이블 동작동안 임시 레지스터로서 기능한다.

ASTART 레지스터(404)는 현재 라인의 시작 어드레스의 기록을 저장한다. ASTART 레지스터(404)는 또한 가이드된 전송을 실행할 때에 가이드 테이블 어드레스/오프셋 엔트리를 위하여 배가된다.

BSTART 레지스터(405)는 현재 패치(patch)의 시작 어드레스의 기록을 저장한다. BSTART 레지스터(405)는 대신에 오프셋 가이드된 전송(offset guided transfer)을 위한 베이스 어드레스 및 델타 가이드된 전송(delta guided transfer)을 위한 최종 패치 시작 어드레스를 포함한다.

CRNTADR 레지스터(406)는 패킷 서비스의 현재 어드레스를 저장한다. 이는 다음의 액세스가 실행될 어드레스이다. 이 값에 전송되는 바이트의 수가 가산(또는 감산)되어 다음 어드레스를 발생시킨다. 라인 또는 패치가 완성되면, CRNTADR 레지스터(406)가 다음 라인 또는 패치의 시작 어드레스로 대신 로드된다.

BCPITCH 레지스터(407)는 편재 패치의 시작 어드레스에 가산(또는 감산)되어 다음 패치의 시작 어드레스를 발생시키는 피치를 저장한다. 가이드된 전송을 실행할 때에, BCPITCH 레지스터(407)는 가이드 테이블 포인터로 사용된다.

ACOUNT 레지스터(408)는 한 라인 내의 바이트의 수를 저장한다. 이는 상수일 수도 있고, 가변 패치 가이드된 전송을 위한 가이드 테이블 엔트리로부터 로드될 수도 있다.

ACURRENT 레지스터(409)는 라인 내에 남아있는 현재 바이트의 수를 저장한다.

BCOUNT 레지스터(410)는 하나의 패치 내의 라인의 수보다 적은 라인 수를 저장한다. 이는 상수일 수도 있고, 가변 패치 가이드된 전송을 위한 가이드 테이블 엔트리로부터 로드될 수도 있다.

BCURRENT 레지스터(411)는 패치 내에 남아있는 라인의 수보다 적은 현재의 라인 수를 저장한다.

CCURRENT 레지스터(412)는 패킷 내에 남아있는 패치의 수보다 적은 현재의 패치 수를 저장한다. 가이드된 전송을 실행할 때에, 이는 대신하여 가이드 테이블에 남아있는 엔트리의 수를 나타낸다.

CBYTES 레지스터(413)는 직접 메모리 액세스 또는 캐시 서비스에 의하여 전송될 남아있는 현재 바이트 수를 저장한다.

BUFPTR 레지스터(414)는 소스 머신(320)과 데스티네이션 머신(340)에 의하여 공유되는 레지스터이다. BUFPTR 레지스터(414)는 외부 대 외부 패킷 전송 중에 버퍼로 사용되는 파라메타 메모리의 부분을 지정한다. 온칩 어드레스의 상위 비트를 발생시키기 위하여 멀티플렉서가 사용될 수 있다.

FCOUNT 레지스터(431; 제12도에 도시됨)는, 현재 파이프라인 내에 있는 모든 명령이 완료된 후에 패킷 전송 FIFO 버퍼(311) 내에 얼마나 많은 바이트가 남는지를 나타내는 5 비트 값(16 - 0의 값)을 저장한다.

BUFCOUNT 레지스터(441; 제13도에 도시됨)는 외부 대 외부 버퍼에 얼마나 많은 여분의 바이트가 남는지를 나타내는 8 비트 값(128 - 0의 값)을 저장한다.

소스 머신(320)의 핵심은 머신이 전송하고자 하는 바이트의 수를 결정하는 계산 하드웨어와 방법이다. 이는 바이트 수 계산 회로(415)에서 이루어진다. 이는 차원 내에 남아있는 바이트 수, 어드레스 정렬, 버스 사이즈 및 차원이 순방향 또는 역방향으로 어드레스되는지 여부에 의하여 결정된다. 이 계산에서 엔디언은 중요하지 않다. 예를 들어, 차원 내에 남아있는 바이트의 수가 8 이상이라 가정하면, 표 11은 상이한 정렬, 버스 크기 및 어드레싱 방향(순방향 또는 역방향)에 대하여 머신이 전송하는 바이트 수를 나타낸다.

[표 11]

3LSB			64비트 버스		32비트 버스		16비트 버스		8 비트 버스	
어드레스 비트			순방향	역방향	순방향	역방향	순방향	역방향	순방향	역방향
0	0	0	8	1	4	1	2	1	1	1
0	0	1	7	2	3	2	1	2	1	1
0	1	0	6	3	2	3	2	1	1	1
0	1	1	5	4	1	4	1	2	1	1
1	0	0	4	5	4	1	2	1	1	1
1	0	1	3	6	3	2	1	2	1	1
1	1	0	2	7	2	3	2	1	1	1
1	1	1	1	8	1	4	1	2	1	1

버스가 64 비트 폭이고 어드레싱이 순방향인 경우, 상기 값은 3 최하위 어드레스 비트의 2의 보수(complement)를 취하여 연역된다. 제4 어드레스 비트는 캐리아웃(carry out)이다. 차원이 역방향으로 어드레스되는 경우, 바이트 수는 3 최하위 어드레스 비트의 값에 단순히 1을 더하여 계산된다. 버스가 32 비트 폭이고 어드레싱이 순방향인 경우, 이상의 값은 2 최하위 어드레스 비트의 2의 보수를 취하여 연역된다. 캐리 아웃은 제3 비트가 된다. 차원이 역방향으로 어드레스되는 경우, 바이트 수는 2 최하위 어드레스 비트의 값에 단순히 1을 더하여 계산된다. 이러한 경우들에서 통상 계산에 포함되는 다음의 최상위 비트의 효과는 무효화되어야 한다. 버스가 16 비트 폭이고 어드레싱이 순방향이면, 상기 값은 단지 최하위 어드레스 비트의 2의 보수를 취하여 연역된다. 캐리 아웃은 제2 비트가 된다. 차원이 역방향으로 어드레스되는 경우, 바이트의 수는 최하위 비트의 값에 단순히 1을 더하여 계산된다. 이러한 경우들에서, 통상 계산에 포함되는 다음의 두개의 최상위 비트들의 영향이 무효화되어야 한다.

이상에서는 머신이 현재 버스 크기에 정렬하기 위하여 전송하는 바이트 수를 설명하였다. 이 동작 후에 발생하는 일련의 전송은 자동적으로 현재의 버스 사이즈에 의하여 허용되는 최대 수의 바이트가 된다. 버스 사이즈가 변화하면, 이것도 역시 자동적으로 변화한다.

그 다음, 어드레스 레지스터는 전송될 바이트 수만큼 증분(또는 감소)되어, 다음 데이터의 시작을 지정한다. 현재 차원 카운트도 동일한 수만큼 감소되어 남아있는 바이트의 새로운 수를 나타낸다. 이 과정은 머신이 전송하고자 하는 바이트의 수가 차원에 남아있는 바이트의 수보다 커질 때까지 계속된다. 실제로 전송되는 바이트 수는 남아있는 바이트 수보다 낮은 수이어야 하고 그렇지 않을 때 전송이 종료된다. 이 조건은 남아있는 바이트의 수를 전송하고자 하는 바이트의 수로부터 감산함으로써 검출되며, 이 감산에서 자리내림(borrow)이 발생하지 않으면 차원의 최종 전송에 도달한 것이고, 바이트 카운트는 전송할 바이트 수를 위하여 사용된다. 자리내림 조건에 기초하여 이들 두 값 중에 선택하기 위하여 멀티플렉서가 사용된다.

차원 내에서 전송이 최후의 것이고 패킷 요청이 실행 중인 경우, 정상적인 제1 차원 계산 대신에 다음 차원 파라메타들이 계산된다. BCURRENT 레지스터(411) 내의 값이 0 인지의 여부에 따라, BCURRENT 레지스터(411)가 감소되거나 CCURRENT 레지스터(412)가 감소된다. 두 경우에서, 낮은 차원의 XCURRENT 계수가 그 XCOUNT 레지스터로부터 로드되고 다음의 제1 차원이 시작된다.

다음 차원의 어드레스 역시, BCURRENT 레지스터(411)의 값이 0 인지의 여부에 따라, ABPITCH 레지스터(402)에 저장된 값을 ASTART 레지스터(404)에 또는 BCPITCH 레지스터(407)에 저장된 값을 BSTART 레지스터(405)에 가산(또는 감산)하여 계산된다. 계산된 값은 CRNTADR 레지스터(406) 및 ASTART 레지스터(404)에 기입되고, BCURRENT 레지스터(411) 내의 값이 0인 경우, BSTART 레지스터(405)에도 기입된다.

제1 차원의 최종 전송이 실행되고 BCURRENT 레지스터(411) 및 CCURRENT 레지스터(412)의 값이 둘 다 0일 때, 소스 머신(320)은 패킷 서비스를 종료한 것이 된다. 소스 머신(320) 및 데스티네이션 머신(340) 모두가 이 상태에 도달했을 때 모든 패킷 서비스가 완료된다. 패킷 전송 옵션 필드의 소스 및 데스티네이션 갱신 모드에 따라, 추가의 차원 계산이 실행될 수 있고 CRNTADR 레지스터(406) 내에 저장된 데이터가 원래의 패킷 요청 위치에 세이브된다. 이는 전송이 중단된 지점에서 이후에 패킷 전송을 재개하는 것을 가능하게 한다.

이러한 어드레스 계산은 두개의 32 비트 가산기(420, 423)에서 이루어진다. 멀티플렉서(421, 422)는 가산기(420)로의 입력을 선택한다. 유사한 방법으로, 멀티플렉서(424, 425)는 가산기(423)로의 입력을 선택한다. 가산기(420)는 CACHEADR 레지스터(401), ABPITCH 레지스터(402), ASTART 레지스터(404), BSTART 레지스터(405), CRNTADR 레지스터(406) 및 BCPITCH 레지스터(407)를 서비스한다. 가산기(423)는 ACURRENT 레지스터(409), BCURRENT 레지스터(411), CCURRENT 레지스터(412) 및 CBYTES 레지스터(413)를 서비스한다.

이상의 설명은 주로 패킷 전송에 관한 것이나, 캐시 서비스 및 버퍼 조작에도 유사한 절차가 응용될 수 있음에 유의한다. 캐시 서비스의 경우에, 어드레스가 항상 정렬된 경계에서 시작하므로 바이트 카운터가 없고, 어드레스가 캐시 서브 블록의 끝에 이르렀을 때 서비스가 종료된다. 이는 소스 머신(320)이 디지탈 영상/그래픽 프로세서의 명령 캐시, 또는 마스터 프로세서(60)의 데이터 또는 명령 캐시를 서비스하는지의 여부에 따라 달라진다.

이상의 설명은 계산된 수의 바이트 전송이 항상 수행될 수 있음을 전제로 한다. 그러나, 그러한 수의 바이트를 전송할 머신의 패킷 전송 FIFO 버퍼(311)가 너무 채워지거나 비게 될 수 있기 때문에, 이것은 옳지 않다. 따라서, 이 경우에는 파이프라인에 동작을 로드하기 이전에 추가의 계산이 요구된다.

소스 머신(320)이 패킷 전송 FIFO 버퍼(311)의 빈 공간보다 많은 바이트를 전송하고자 할 때에는 대기하여야 한다. 데스티네이션 머신(340)이 패킷 전송 FIFO 버퍼(311)의 빈 공간보다 많은 데이터를 전송하고자 할 때에는, 대기하여야 한다. 소스 머신(320)이 데스티네이션 머신(340)의 판독하기에 충분한 바이트를 패킷 전송 FIFO 버퍼(311)에 로드하려고 할 때, 양 전송은 계속될 수 있다.

이러한 계산은 제12도에 도시된 논리에 의하여 실행된다. FCOUNT 레지스터(431)는, 이미 파이프라인 내에 있는 동작이 일단 완료된 후에 패킷 전송 FIFO 버퍼(311) 내에 있게 될 데이터의 바이트 수를 기록한다. 가산기(434)는 소스 머신(320)이 패킷 전송 FIFO 버퍼(311)로 전송하고자 하는 바이트 수에 이 값을 가산한다. 그 합이 16을 초과하면, 소스 머신(320)은 동작을 계속할 수 없고 데스티네이션 머신(340)이 어떤 데이터를 비울 때까지 대기하여야 한다. 가산기(434)는 소스머신(320)에 공급되어 이를 정지시키는 16보다 큰 출력을 발생시킨다. 현재의 값이거나 또는 가산기(343)의 16보다 큰 출력에 의하여 선택된 합산의 결과인, 패킷 전송 FIFO 버퍼(311) 내의 새로운 바이트 수는 멀티플렉서(435)를 경유하여 감산기(436)로 전달된다. 감산기(436)는 패킷 전송 FIFO 버퍼(311) 내의 새로운 바이트 수를 데스티네이션 머신(340)이 전송하고자 하는 바이트의 수로부터 감산한다. 감산기(436)가 자리내림을 발생시키지 않으면, 데스티네이션 머신(340)은 계속 동작할 수 있다. 감산기(436)가 자리내림을 발생시키면, 그 신호가 데스티네이션 머신(340)을 정지시킨다. 자리내림 신호는 멀티플렉서(437)가 차분값 또는 감산 이전의 값을 선택하여 FCOUNT 레지스터(431)에 세이브하도록 제어한다.

캐시 서비스를 위하여 캐시 버퍼(312)를 사용할 경우에도 유사한 계산이 요구된다. 멀티플렉서(433)로의 모드 신호는 FCOUNT 레지스터(431)가 계산의 소스의 역할을 할지 또는 캐시 버퍼(312) 내에 저장된 바이트 수를 저장하는 유사한 CHCOUNT 레지스터(432)의 역할을 할지를 선택한다.

외부 대 외부 전송을 실행할 때에, 전송 프로세서(80)는 요청 프로세서의 파라메타 메모리의 일부를 버퍼로서 사용한다. 이 외부 대 외부 버퍼의 상태 및 그 내용은 관리될 필요가 있다. 버퍼 포인터 BUFPTR 레지스터(414)는 소스 머신(240) 및 데스티네이션 머신(340)에 의하여 공유되는데, 이는 이 버퍼에 동시에 하나의 머신만이 액세스할 수 있기 때문이다. BUFPTR 레지스터(414)는 소스 머신(320) 및 데스티네이션 머신(340)의 정상적 논리를 사용하여 버퍼로/로부터 전송되는 바이트 수만큼 증가된다. BUFPTR 레지스터(414)는 단지 8 비트의 길이만 가지면 된다. 어드레스의 최상위 비트는 각각의 프로세서에 대하여 고정되며, 요청 프로세서에 기초한 불변(constant) 멀티플렉서로부터 공급된다. 전송되는 바이트의 수는 정상 정렬/스페이스-레프트 논리(normal alignment/space-left logic)를 사용하여 계산된다. 외부 대 외부 버퍼를 비우는 소스 머신(320)은, 그 바이트의 남은 값을 저장하기 위하여 BUFCOUNT 레지스터(441)를 사용한다. 외부 대 외부 버퍼를 채우는 데스티네이션 머신(340)은, 그 바이트의 남은 값을 저장하기 위하여 FCOUNT 레지스터(431)를 사용한다.

제13도는 버퍼-카운트 계산법을 도시한다. BUFCOUNT 레지스터(441)는 외부 대 외부 버퍼 내의 여분의 바이트 수 또는 전송의 방향에 따라 비워질 잔류 데이터 바이트의 수의 8 비트 계수이다. 소스 머신(320)은 외부 대 외부 버퍼를 채울 때 패킷 전송 FIFO 버퍼(311)에 로드하는 바이트 수만큼 BUFCOUNT 레지스터(441)를 감소시킨다. BUFCOUNT 레지스터(441)가 8 미만의 값에 달하면, 소스 머신(320)이 정지하고, 데스티네이션 머신(340)은 패킷 전송 FIFO 버퍼(311)가 외부 대 외부 버퍼로 비워질 때까지 계속 동작한다. 따라서 소스 머신(320)은 외부 대 외부 버퍼에 맞는 데이터보다 많은 데이터를 불러내올 수 없다. 외부 대 외부 버퍼는 최소한 120 바이트로 채워질 것이다.

외부 대 외부 버퍼를 채우는 동작과 비우는 동작 사이에서, 128로부터 BUFCOUNT 레지스터(441)를 감산하여, 외부 대 외부 버퍼 내에 남아있는 데이터 바이트의 수가 얻어진다. 외부 대 외부 버퍼를 비울 때 소스 머신(320)은 BUFCOUNT 레지스터(441)를 0에 달할 때까지 감소시킨다. 데스티네이션 머신(340)은 계속하여 패킷 전송 FIFO 버퍼(311)를 비우고, 다 비워지면 방향이 다시 역전된다. BUFCOUNT 레지스터(441)는 외부 대 외부 버퍼를 채우는 동작이

개시되기 전에 128로 재설정된다. 멀티플렉서(443, 443) 및 전(full) 가산기(444)는 이 필요한 계산을 수행한다.

패킷 전송이 중단되었을 때, BUFCOUNT 레지스터(441) 및 BUFPTR 레지스터(414)에는 다른 패킷 파라메타가 저장된다. 외부 대 외부 버퍼를 비우거나 채우는 현재의 방향을 나타내는 하나의 비트도 세이브되어 복원시에 동작이 정화하게 재개될 수 있도록 한다.

제14a 및 14b도는 패킷 전송 FIFO 버퍼(311)와 리틀 엔디언 및 빅 엔디언 각각을 위한 포인터 FDPTR 및 FSPTR의 관계를 도시한다. 패킷 요청이 소스 및 데스티네이션의 형태가 거의 임의적이며 독립적인 것을 허용하기 때문에, 데스티네이션에 정렬시키는 것이 항상 가능하지는 않다. 제8도에서, 소스 머신(320)은 매 사이클당 0 내지 8 바이트를 저장할 수 있고 데스티네이션 머신(340)은 매 사이클당 0 내지 8 바이트를 페치할 수 있으며, 소스 및 데스티네이션의 정렬은 서로에 대하여 계속적으로 변화한다. 따라서, 패킷 전송 FIFO 버퍼(311)는 소스로부터 0 - 8 바이트로 로드됨과 동시에 데스티네이션으로 0 - 8 바이트로 비워질 수 있는 16 바이트 폭의 FIFO이다. 소스 멀티플렉서 및 정렬 논리(330)는 소스로부터 적절한 바이트를 추출하고, 이들을 패킷 전송 FIFO 버퍼(311) 내에서 연속적인 FIFO 데이터를 유지하는 위치에 저장시킨다. 데스티네이션 멀티플렉서 및 정렬 논리(350)는 패킷 전송 FIFO 버퍼(311)로부터 "가장 오래된" 바이트를 추출하고, 이들을 데스티네이션에 기입하기 전에 8 바이트 워드 내의 정확한 위치에 정렬시킨다. 소스 멀티플렉서 및 정렬 논리(330)와 데스티네이션 멀티플렉서 및 정렬 논리(350)는, 각각 자신의 전송을 가능한 한 신속하게 64 비트 더블 워드 경계로 정렬시키고, 가능한 한 오랫동안 정렬 상태로 남아있을 것이다. 따라서, 소스 멀티플렉서 및 정렬 논리(330)와 데스티네이션 멀티플렉서 및 정렬 논리(350)는 변동하는 정렬을 지원한다.

패킷 전송 FIFO 버퍼(311)는 16 바이트를 순환형 버퍼로 취급한다. 데이터의 개시 위치는 임의의 바이트 위치일 수 있다. 데스티네이션 멀티플렉서 및 정렬 논리(350)가 데이터를 추출하는 위치는, 4 비트 FDPTR 레지스터에 의하여 표시된다. FDPTR 레지스터는, 데이터가 추출되는 사이클과 동일한 사이클 내에서 변형포인터를 데이터의 새로운 시작 위치로 이동시킨다. 데스티네이션 멀티플렉서 및 정렬 논리(350)는 추출된 바이트를 취하여 이들을 데스티네이션 더블 워드 내의 정확한 위치로 정렬시킨다. 패킷 전송 FIFO 버퍼(311)로부터의 추출 및 정렬은, 데스티네이션 멀티플렉서 및 정렬 논리(350) 내의 8개의 병렬 16 비트 배럴 시프터로 구성되는 16 위치 바이트 회전기를 사용하여 하나의 단계에서 적절히 이루어진다. 그리고 나서 바이트를 교체하여 다른 데이터 회전이 실행되도록, 건드-데이터(gunned-data) 조작이 실행될 수 있다. 또한, 데스티네이션 멀티플렉서 및 정렬 논리(350)는 적절한 바이트만이 데스티네이션에 기입되도록, 유사한 방식으로 바이트 스트로브를 조작할 수 있다.

소스 멀티플렉서 및 정렬 논리(330)로부터의 패킷 전송 FIFO 버퍼(311)를 로드시키는 것은 거의 동일한 작업이다. 소스 멀티플렉서 및 정렬 논리(330)는 그 데이터를 4 비트 FSPTR 레지스터에 의하여 표시된 첫번째 빈 위치에 기입하고, 새로운 첫번째 빈 위치로 FSPTR 레지스터를 갱신시킨다. 패킷 전송 FIFO 버퍼(311)가 다 찼을 때, 일단 데스티네이션 멀티플렉서 및 정렬 논리(350)가 어떤 데이터를 추출하였으면 FSPTR 레지스터는 실제로 비게되는 첫번째 위치를 지적한다.

데스티네이션 머신(340)은 클록 사이클의 슬레이브 위상 상에서 자신의 데이터를 추출한다. 소스 머신(320)은 자신의 데이터를 후속하는 클록 사이클의 마스터 위상에 로드한다.

빅 엔디언 동작 모드와 리틀 엔디언 동작 모드를 지원하기 위하여 데이터 회전이 양방향으로 이루어질 필요가 있다. 데이터의 연속성이 항상 정확하도록 하기 위하여, 패킷 전송 FIFO 버퍼(311)의 외부로부터 보았을 때, 데이터의 회전 방향은 리틀 엔디언의 경우 시계 방향이고 빅 엔디언의 경우 시계 반대 방향이다. 따라서, 소스 멀티플렉서 및 정렬 논리(330), 데스티네이션 멀티플렉서 및 정렬 논리(350), 포인터들 FSPTR 및 FDPTR 및 패킷 전송 FIFO 버퍼(311)의

레지스터는 두 가지 엔디언에 대처할 수 있도록 설계되어야 한다.

제12도의 FCOUNT 레지스터(431)는 패킷 전송 FIFO 버퍼(311) 내에 얼마나 많은 바이트들이 보유되어 있는지를 나타내는 표시를 저장한다. FCOUNT 레지스터(431)는 16보다 큰 값을 담을 수 없다. 이 레지스터는 파이프라인을 로드하기 전에 소스 머신(320) 및 데스티네이션 머신(340)에 의하여 조작되어, FCOUNT 레지스터(431)에 저장된 값이 현재 파이프라인 내의 모든 명령이 완료된 후에 패킷 전송 FIFO 버퍼(311) 내에 얼마나 많은 바이트가 존재하게 될지를 나타내게 한다. 이는 패킷 전송 FIFO 버퍼(311) 내에 장애를 야기할 동작이 파이프라인 내로 로드될 수 없도록 한다. 그리하여 데스티네이션 머신(340)이 항상 충분한 바이트를 가지게 되고, 소스 머신(320)이 항상 충분한 공간을 가지게 된다. 이 결과 FDPTR 또는 FSPTR이 상호 오버 테이킹할 위험이 없다. 새로운 패킷 요청을 시작하면서 FDPTR과 FSPTR 양자가 동일한 장소를 가리키도록 이들을 0으로 재설정하고, FCOUNT 레지스터(431)를 0으로 재설정한다. 이는 패킷 전송 FIFO 버퍼(311)의 정확한 초기화를 보장한다.

패킷 전송이 중단되었을 때, 패킷 전송 FIFO 버퍼(311) 레지스터 자체뿐만 아니라 FDPTR, FSPTR 및 FCOUNT 레지스터(341)가 세이브된다. 이는 패킷 전송 FIFO 버퍼(311)의 전체 상태를 유지하여, 패킷 전송의 재개 시에 복구될 수 있도록 한다.

필-위드-밸류(fill-with-value) 패킷 전송은 패킷 전송 FIFO 버퍼(311)의 최하위 더블 워드로 충전 값을 로드하고, FCOUNT 레지스터(431)를 16으로 설정하고, FDPTR을 엔디언에 대한 정확한 데이터 시작 지점으로 설정한다, 이는 리틀 엔디언에 대해서는 바이트 0이며 빅 엔디언에 대해서는 바이트 7이다. FCOUNT 레지스터(431)의 값은 필-위드-밸류 패킷 전송 도중에 변경되지 않는다. 데스티네이션 머신(340)은 항상 충만한 패킷 전송 FIFO 버퍼(311)를 인지하고 채워진 값들을 페치하고 정렬시킨다. 소스 머신(320)도 항상 충만한 패킷 전송 FIFO 버퍼(311)를 인지하고 필요에 따라 스톨(stall)시킨다. 필-위드-밸류 패킷 전송을 중단하거나 복구할 때에 패킷 전송 FIFO 버퍼(311)의 상태가 단순히 세이브되고 임의의 다른 중단 형태에 대하여 복구된다. 따라서 값으로 채워진 패킷 전송을 중단하기 위하여 다른 특별한 고려가 필요하지 않다.

소스 머신(320)은 두개의 파이프라인을 포함하고 다른 하나의 파이프라인을 데스티네이션 머신(340)과 공유한다. 파이프라인은 내부 메모리 인터페이스(301) 및 외부 메모리 인터페이스(302)와 인터페이스하여, 온칩 대 온칩, 오프칩 대 온칩, 온칩 대 오프칩, 및 오프칩 대 오프칩의 4 방향의 데이터 전송을 허용한다. 소스 머신(320)의 어드레스 파이프라인(427)과 바이트 파이프라인(428)은 데스티네이션 머신의 파이프라인들로부터 분리되어 있으나, 공통의 파이프라인 어드밴스를 제공하기 위하여 인터로크(interlock)되어 있다. 이는 한 단계에 머무르면서 내부 메모리 인터페이스(00301) 및 외부 메모리 인터페이스(302) 상에 독립적으로 작용을 가하는 것을 허용한다. 하나의 인터페이스 내의 불연속성은 또한 다른 인터페이스를 스톨시킨다. 어드레스 파이프라인(427)은 소스 머신(320)에 의하여 액세스 될 어드레스들을 저장한다. 바이트 파이프라인(428)은 바이트 스트로브와 정렬 정보를 저장한다. 이 정보는 어느 바이트가 적절한 것인가에 관한 정보 및 이들을 패킷 전송 FIFO 버퍼(311) 또는 캐시 버퍼(312)에 로드시키고 제거시키는 방법을 나타낸다. 전송 제어기(80) 레지스터 전송에서, 바이트 파이프라인(428)은 레지스터 번호를 저장한다. 서비스된 정보는 또한 동작이 완료되는 시기를 나타내기 위하여 바이트 파이프라인을 통과한다.

소스 머신(320) 및 데스티네이션 머신(340)은 요청 파이프라인(429)을 공유한다. 요청 파이프라인(429)은, 예를 들어 낮은 우선 순위의 패킷 전송, 직접 외부 액세스 등의 실행되는 동작의 유형과 같은 상태 정보 및 요청 프로세서의 표시를 저장한다. 요청 파이프라인(429)은 또한 언제 상태적 이벤트가 발생한 것인가를 나타내는 병렬 제어 정보를 저장한다.

내부 파이프라인 내에는 하나 또는 그 이상의 액세스가 스톨될 수 있도록 때때로 버블(bubble)이 있을 수 있다. 전송 제어기(80)는 이러한 버블들을 가이드된 전송 중에 가이드 테이블 값을 페칭하기 위하여 사용한다. 이는 전송 제어기(80)의 전송 대역폭을 더욱 효율적으로 사용할 수 있도록 한다. 데스티네이션 레지스터가 가용 상태에 있지 않으면 이 불러내온 가이드 테이블의 값은 GTPIPE 레지스터(403)의 연관된 섹션에 저장된다.

전송 제어기(80)는 바이트 정렬의 형태만으로 패킷 전송 동안 데이터 정렬을 제공할 수 있다. 소스 머신(320)과 데스티네이션 머신(340)은 서로 독립적이기 때문에, 매 사이클마다 0 바이트와 8 바이트 사이에서 각각 페치하거나 저장할 수 있다. 이는 서로에 대한 소스와 데스티네이션 어드레스 정렬이 일정하게 변할 수 있음을 뜻한다. 유동성 있는 정렬을 지원하기 위하여, 전송 제어기(80)는 패킷 전송 선입선출 FIFO 버퍼(311), 소스 정렬 논리(332) 및 데스티네이션 정렬 논리(352)들을 갖는다. 패킷 전송 FIFO 버퍼(311)는, 소스로부터 0 - 8 바이트가 로드되는 동시에 데스티네이션으로 0 - 8 바이트가 출력될 수 있는 16-바이트 FIFO이다. 소스 멀티플렉서(331)와 소스 정렬 논리(332)는 소스로부터 적절한 바이트를 추출하고 그들을 이전의 소스 바이트에 인접하게 패킷 전송 FIFO 버퍼(311)에 저장한다. 한편 데스티네이션 정렬 논리(352)와 데스티네이션 멀티플렉서(351)는 패킷 전송 FIFO 버퍼(311)로부터 " 가장 오래된" 바이트를 추출하고 그들을 현재 어드레스된 데스티네이션 8-바이트 더블 워드 내의 정확한 위치로 정렬시킨다. 패킷 전송 정렬 및 FIFO 동작은 자동적이며 실시예를 숙고한 프로그래머는 이를 알기 쉬울 것이다.

캐시 버퍼(312)는 동작면에서 패킷 전송 FIFO 버퍼(311)와 유사한 8-바이트 버퍼이고 캐시 및 직접 외부 액세스 동작동안 사용된다. 캐시로의/부터의 전송은 항상 8-바이트 폭이다. 캐시 버퍼(312)는 외부 메모리 버스가 64 바이트 폭보다 작은 경우에 데이터를 정렬하는 것을 돕는다. 패킷 전송 FIFO 버퍼(311)와는 독립적으로 캐시 버퍼(312)를 제공하는 것은 현재 패킷 전송 FIFO 버퍼(311) 내에 있는 패킷 데이터를 먼저 비우지 않고서도, 높은 우선 순위의 캐시와 직접 외부 액세스 요청이 패킷 전송 중에 제공될 수 있게 한다.

전송 제어기(80)는 온칩 메모리로 맵핑된 4개의 사용자 액세스 가능한 레지스터를 갖는다. 이들은 로드 및 저장 명령을 통하여 마스터 프로세서(60)에 의하여 액세스 가능하다. 이 레지스터들은 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)로 액세스할 수 없다. 표 12는 이러한 레지스터들의 목록이다.

[표 12]

어드레스	TC 레지스터
0x01820000	REFCNTL
0x01820004	PTMIN
0x01820008	PTMAX
0x0182000C	FLTSTS

REFCNTL 레지스터는 시스템 DRAM 리프레시 사이클을 제어하는데 사용되는 2개의 16 비트 값을 포함한다. 16 비트 REFRATE 필드(비트 15 - 0)는 DRAM 리프레시 사이클 요청이 발생될 간격을 결정한다. REFRATE의 값은 각 리프레시 요청 사이에서 발생하는 멀티프로세서 집적 회로(100)의 클럭 사이클 수를 나타낸다. REEFRATE의 값이 32(Hex " 0020")보다 작은 경우는 DRAM 리프레시가 디스에이블된다. REFRATE 필드는 재설정시 32(Hex " 0020")로 설정된다. DRAM 리프레시동안, 16 비트 의사 어드레스(pseudo-address)가 리프레시 뱅크 디코딩이나 $\overline{RAS}$ 리프레시에서만 사용되기 위한 외부 어드레스 버스 상에 출력된다. 16 비트 RPARLD 필드(비트 31 - 16)는 리프레시 사이클 동안 출력될 최대값을 포함한다. 리프레시 어드레스 카운터는 현재의 리프레시 어드레스의 트랙을 유지하는 데에 이용된다. 리프레시 사이클이 발생될 때마다, 카운터는 감소된다. 카운터가 0에 도달하면, RPARLD에 값이 다시 로드된다. RPARLD 필드는 재설정시 Hex " OFFFF" 로 설정된다.

패킷 전송 최소 레지스터 PTMIN(511)과 패킷 전송 최대 레지스터 PTMAX(512)는 패킷 전송의 동작 시간을 제어한다. 이러한 레지스터들은 제31도에 도시되어 있으며, 아래에서 설명될 것이다. 패킷 전송 최소 레지스터 RTMIN(511)은, 전송 제어기(80)가 더 높은 우선 순위의 패킷 전송에 의하여 인터럽트되기 전에 전송 제어기에 의하여 패킷 전송이 반드시 서비스되어야만 하는 클럭 사이클의 최소수를 저장한다. 패킷 전송 최소 레지스터 PTMIN(511)의 24개의 최하위 비트들만이 양호한 실시예에서 구현되었다. 상위 비트들을 판독하는 것은 언제나 " 0" 을 복귀시키고 이러한 비트들을 기입하는 것은 아무런 영향도 끼치지 않는다. 패킷 전송 최소 레지스터 PTMIN(511)은 재설정시 Hex " 10000" (64K 사이클)으로 로드된다. 패킷 전송 최대 레지스터 PTMAX(512)는, 타임-아웃 전에 패킷 전송 최소 레지스터 PTMIN(511)에 의하여 규정된 시간에 이어서 패킷 전송이 계속되는 최대 시간 양을 결정한다. 본 실시예에서, 패킷 전송 최대 레지스터 PTMAX(512)는 24개의 최하위 비트들을 구현한다. 상위 비트에 대한 판독은 " 0" 을 복귀시키고 이러한 비트들을 기입하는 것은 아무런 영향도 미치지 않는다. 따라서 패킷 전송은 PTMIN 사이클에 후속하는 더 높은 우선 순위의 패킷 전송에 의하여 인터럽트될 수 있고 PTMIN + PTMAX 사이클에 후속하는 동일한 우선 순위의 또 다른 패킷 전송을 수행하기 위하여 지연될 수도 있다. 패킷 전송 최대 레지스터 PTMAX(512)에 저장된 값은 더 높은 우선 순위의 패킷 전송을 위하여 패킷 전송이 지연된 때에는 아무런 영향도 끼치지 않는다. 재설정시, Hex " 10000" (64K 사이클)의 값이 PTMAX로 로드된다.

FLTSTS 레지스터는 패킷 전송이나 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 명령 캐시 필(fill) 또는 직접 외부 액세스 사이클 동안 폴트가 발생했음을 나타내는 상태 비트들을 포함한다. 표 13은 FLTSTS 레지스터 상의 특정 비트의 의미를 보여준다.

[표 13]

FLTSTS 비트	명칭	기 능
31	DIGPC7	보류
30	DIGPC6	보류
29	DIGPC5	보류
28	DIGPC4	보류
27	DIGPC3	DIGP3 캐시/DEA 폴트
26	DIGPC2	DIGP2 캐시/DEA 폴트
25	DIGPC1	DIGP1 캐시/DEA 폴트
24	DIGPC0	DIGP0 캐시/DEA 폴트
23	DIGPC7	보류
22	DIGPC6	보류
21	DIGPC5	보류
20	DIGPC4	보류
19	DIGPC3	DIGP3 패킷 전송 폴트
18	DIGPC2	DIGP2 패킷 전송 폴트
17	DIGPC1	DIGP1 패킷 전송 폴트
16	DIGPC0	DIGP0 패킷 전송 폴트
0	M	MP 패킷 전송 폴트

전송 제어기(80)는 소정 메모리 폴트의 검출에 근거하여 FLTSTS 레지스터의 개개의 비트들을 설정한다. M 비트(비트 0)는 마스터 프로세서(60) 요청 패킷 전송 동안 폴트가 발생된 때 1로 설정된다. 비트(19 - 16)들은 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)들에 의하여 요청된 패킷 전송 폴트를 나타내기 위하여 사용된다. DIGP3(비트 19)은 디지탈 영상/그래픽 프로세서(74) 패킷 전송 폴트를 나타낸다. DIGP2(비트 18)는 디지탈 영상/그래픽 프로세서(73) 패킷 전송 폴트를 나타낸다. DIGP1(비트 17)은 디지탈 영상/그래픽 프로세서(72) 패킷 전송 폴트를 나타낸다. DIGP0(비트 16)은 디지탈 영상/그래픽 프로세서(71) 패킷 전송 폴트를 나타낸다. 비트(23 - 20)들은 8개의 디지탈 영상/그래픽 프로세서를 포함하는 실시예에서 패킷 전송 폴트를 보고하기 위하여 보류된다. 이러한 비트들의 각각은 상응하는 디지탈 영상/그래픽 프로세서에 의하여 요청된 패킷 전송 동안 폴트가 발생한 때에 1로 설정된다.

DIGPC3 비트(비트 27)는 디지탈 영상/그래픽 프로세서(74)에 의하여 요청되는 직접 외부 액세스 동작이나 캐시 필 동안 폴트가 발생된 때 1로 설정된다. 유사하게, DIGPC2 비트(비트 26)에 있는 " 1" 은 디지탈 영상/그래픽 프로세서(73)에 의하여 요청되는 직접 외부 액세스 동작이나 캐시 필 동안의 폴트를 나타낸다. DIGPC1 비트(비트 25)에 있는 " 1" 은 디지탈 영상/그래픽 프로세서(72)에 의하여 요청되는 직접 외부 액세스 동작이나 캐시 필 동안의 폴트를 나타낸다. DIGPC0 비트(비트 24)에 있는 " 1" 은 디지탈 영상/그래픽 프로세서(71)에 의하여 요청되는 직접 외부 액세스 동작이나 캐시 필 동안의 폴트를 나타낸다. 비트(31 - 28)들은 8 개의 디지탈 영상/그래픽 프로세서를 사용하는 실시예에서 추가의 디지탈 영상/그래픽 프로세서에 의하여 요청되는 직접 외부 액세스 동작이나 캐시 필 동안의 폴트를 나타내기 위하여 보류된다.

M, DIGP(3 - 0), 또는 DIGOC(3 - 0) 비트들 중의 어느 하나에 대한 설정은 마스터 프로세서(60)를 인터럽트한다. 설정되었던 FLTSTS 비트에 대한 클리어는 연관된 패킷 전송, 캐시 필 또는 직접 외부 액세스를 리스케쥴링(rescheduling)되게 만든다. FLTSTS 비트들은 적절한 비트에 " 1" 을 기입함으로써 클리어된다. 비트에 " 0" 을 기입하는 것은 아무런 효과도 갖지 않는다.

마스터 프로세서(60) 레지스터는 스코어 보드되기(score boarded) 때문에, 폴트된 직접 외부 액세스 요청은 이러한 프로세서의 명령 파이프라인을 스톨시킬 필요가 없다. 마스터 프로세서(60)는 데이터를 액세스하는 다른 시도들이 이루질 때에만 직접 외부 액세스 저장 폴트에 의해 스톨될 것이다. 폴트된 직접 외부 액세스 로드들은, 동일한 조건 하에서 또는 직접 외부 액세스에 의하여 로드될 레지스터를 사용하려는 시도가 이루어지는 경우에서 마스터 프로세서(60)가 스톨되도록 만든다. 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74)에 의하여 요청된 폴트 직접 외부 액세스는, 요청 프로세서를 그들에 의하여 요청된 직접 외부 액세스가 완료될 때까지 요청하는 항상 스톨시킬 것이다.

패킷 전송은 메모리의 두 영역 사이에서 데이터 블럭들을 전송하는 것이다. 데이터는 전송 제어기(80)에 의하여 소스(Src) 메모리 영역으로부터 데스티네이션(Dst) 메모리 영역으로 전송된다. 소스 및 데스티네이션 영역은 온칩 메모리 영역일 수도 있고 오프칩 메모리 영역일 수도 있다. 패킷 전송은 마스터 프로세서(60) 가나 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74) 중의 어느 하나에 의해 전송 제어기(80)에 대한 요청으로서 시작된다. 전송 제어기(80)는 요청 큐잉 및 우선 순위화 논리(request queing and prioritization logic, 303)를 통하여 제6도 및 제7도에 도시된 바와 같은 고정 라운드 로빈 우선 순위화(fixed and round robin prioritization)를 사용하여 요청을 서비스한다. 일단 프로세서가 요청을 제공하면, 프로그램 실행이 계속된다. 패킷 전송은 추가의 프로세서 사이클에 대한 요청 없이 전송 제어기(80)에 의하여 완료될 것이다. 패킷 전송은 다른 우선 순위의 레벨로 제공될 수 있기 때문에, 높은 우선 순위의 전송이 낮은 우선 순위를 인터럽트할 수 있다. 이러한 상황이 발생된 때에는, 낮은 우선 순위의 전송이 전송 제어기(80)에 의하여 중단되고 전송의 현 위치가 세이브된다. 더 높은 우선 순위의 전송이 완료된 때에는, 중단된 전송이 인터럽트되었던 지점에서 자동적으로 재개한다.

패킷 전송에 대한 이해를 돕기 위하여, 패킷 전송을 검토하는데 사용된 용어에 대한 간단한 정의가 요청된다. 라인은 메모리의 연속된 바이트의 수이다. 패치는 시작 어드레스가 동일한 거리만큼 떨어진 라인들의 그룹이다. 패킷은 패치들의 집합이다. 피치는 2개의 라인의 시작 사이 또는 2개의 패치의 시작 사이의 어드레스 상의 차이이다. 파라메타 테이블은 데이터의 패킷을 기술하며, 이들이 소스로부터 데스티네이션으로 어떻게 이동될 것인지를 설명하는 파라메타의 8 더블-워드 길이 집합(8 double-word long collection)이다. 링크 리스트(linked-list)는 각각이 리스트 내의 다음의 테이블을 지시하는 파라메타 테이블의 집합이다. 가이드 테이블은 패킷 전송 내의 개개의 패치들을 기술하는 파라메타의 테이블이다. 소스 전송은 소스 메모리 위치로부터의 데이터 전송이다. 데스티네이션 전송은 데스티네이션 메모리 위치로의 데이터 전송이다.

각 프로세서는 연관된 파라메타 메모리를 갖는다. 연관 파라메타 메모리 내의 다수의 위치는, 프로세서로부터 서빙 패킷 전송이 요청된 때, 전송 제어기(80)에 의한 사용을 위하여 따로 설정된다. 이러한 영역들이 제5도에 도시되어 있다. 전송 제어기(80)에 의하여 사용된 파라메타 메모리들은 전송 제어기(80) 사용에만 제한되는 것이 아니다. 그러나, 프로그래머는 이러한 위치에 자리한 데이터가 패킷 전송 동작동안 전송 제어기(80)에 의하여 재기입(overwritten)될 수 있다는 점을 알아야만 한다. 이러한 위치들은 상기 재기입이 패킷 전송이나 패킷 전송의 데이터를 변경시킬 수 있기 때문에 패킷 전송 요청이 활성 상태인 때에는 절대 기입되어서는 안된다.

패킷 전송을 초기화시키는 프로세서에 대한 프로세스, 방법 또는 시퀀스의 예는 다음과 같다. 해당 파라메타 메모리 내에 패킷 전송 파라메타 테이블을 생성한다. 필요하다면, 온칩 메모리에 패킷 전송의 가이드 테이블을 생성한다. 가이드 테이블의 본질이 아래에 더 설명될 것이다. 제1 파라메타 테이블의 시작을 나타내기 위하여 파라메타 메모리 내에 링크 리스트 시작 어드레스를 설정한다. 제 15도는 이러한 링크 리스트 시작 어드레스가 저장되어야만 하는 해당 파라메타 메모리 내의 위치를 나타내고 있음에 유의한다. 마스터 프로세서(60)에 대하여, 이러한 링크 리스트 시작 어드레스는 Hex "010100FC" 내지 Hex "010100FF" 인 어드레스에 저장되어야만 한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)에 대하여, 이러한 링크 리스트 시작 어드레스는 #이 디지탈 영상/그래픽 프로세서의 번호일 때 Hex "0100#0FC" 내지 Hex "0100#0FF" 인 어드레스에 저장되어야만 한다. 다음에는, 전송 제어기(80)에 요청을 제공하기 위하여 적절한 패킷 전송 우선 순위 비트와 P비트를 설정한다. 이러한 비트들은 마스터 프로세서(60)를 위한 PKTREQ 제어 레지스터와 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 COMM 레지스터 안에 위치된다. 전송 제어기(80)는 이러한 레지스터의 상태를 검출한 후, 제6도와 제7도에 도시된 우선 순위에 따라 패킷 전송 요청을 제공한다.

패킷 전송 요청은 링크 리스트 구조로서 제공된다. 링크 리스트들은, 각각의 패킷 전송이 리스트 상의 다음 엔트리의 엔트리 포인트에 대한 포인터를 포함하는 경우의 패킷 전송 테이블의 단순한 집합이다. 비록 패킷 전송이 온칩 또는 오프칩 메모리(off-chip) 상에서 동작할 수 있지만, 파라메타 테이블들 자체의 링크 리스트들은 온칩 메모리에 저장되어야만 한다. 각각의 프로세서는 메모리에 저장된 다수의 링크 리스트를 가질 수 있지만 그들 중의 하나만이 한번에 활성 상태일 수 있다. 활성화된 링크 리스트의 시작은 요청 프로세서의 파라메타 메모리에 있는 전용 링크 리스트 시작 어드레스에 저장된다. 링크 리스트의 각각의 엔트리 파라메타 테이블들은 리스트 상의 다음 엔트리의 위치에 대한 포인터를 갖는다. 리스트-앤드(list-end)가 패킷 전송 테이블의 패킷 전송 옵션 필드의 스톱 비트(stop bit)에 의하여 표시되기 때문에, 리스트의 최종 엔트리는 어느 곳이라도 지시할 수 있다.

제16도는 간단한 링크 리스트 구조를 도시한다. 이러한 리스트는 2개의 패킷 전송 파라메타 테이블을 갖는다. 리스트 시작 포인터는 제1 패킷 전송 파라메타 테이블의 시작 어드레스를 저장한다. 제1 패킷 전송 파라메타 테이블에 있는 제1 데이터는 본 경우는 마지막인 제2 패킷 전송 파라메타 테이블의 어드레스를 지시한다. 비록 본 실시예는 단지 2개의 패킷 전송 파라메타 테이블을 포함하지만, 링크 리스트는 온칩 메모리 내에서 적절한 한도에서 가능하다.

일단 프로세서가 자신의 P비트를 설정하여 패킷 전송을 제공하면, 전송 제어기(80)는 해당 프로세서의 Q비트를 설정하여 응답한다. 이는 상기 프로세서의 링크 리스트가 전송 제어기(80) 내에서 큐잉되었음을 나타낸다. 패킷 전송 우선순위 레벨에 적절한 라운드 로빈 토큰이 요청 프로세서에 도달한 때, 전송 제어기(80)는 상기 요청을 능동적으로 서비스한다. 전송 제어기(80)는 요청 프로세서의 파라메타 메모리의 링크 리스트 시작 어드레스 위치로부터 링크 리스트의 시작 위치를 판독한다. 전송 제어기(80)는 자신의 소스 머신(320)과 데스티네이션 머신(340) 내의 레지스터로 제1 패킷 전송 파라메타 테이블을 판독한다. 그 다음으로, 전송 제어기(80)는 소스 머신(320)과 데스티네이션 머신(340)을 이용하여, 파라메타 테이블 내에 표시된 대로 데이터를 전송한다. 일단 패킷 전송이 완료되면, 전송 제어기(80)는 파라메타 메모리의 링크 리스트 시작 어드레스를 링크 리스트 내의 다음 어드레스로 갱신한다. 다음의 어드레스는 막 완료된 패킷 전송 파라메타 테이블의 제1 엔트리로부터 판독된다. 다음으로, 전송 제어기(80)는 링크 리스트 상의 최종 엔트리에 도달할 때까지 프로시져를 반복한다.

본 명세서에서 설명된 패킷 전송 장치, 프로시져 및 방법들은, 데이터 이동에 있어서 최대한의 융통성을 허여하기 위한 다양한 포맷과 옵션들을 제공한다. 이러한 포맷들을 고려할 때에는, 패킷 전송의 소스 전송과 데스티네이션 전송들이 서로 독립적이라는 것을 기억하는 것이 중요하다. 이는 패킷 데이터가 자신의 판독된 포맷과는 완전히 다른 포맷을 사용하여 기록될 수 있게 하여, 어떤 수의 스프레딩 또는 머징(merging) 기능이 자동적으로 달성될 수 있게 한다. 2개의 기본 패킷 전송 포맷들이 " 차원화(dimensioning)" 되고 " 가이드(guiding)" 되었다. 이러한 포맷들은 그들이 소스 전송 또는 데스티네이션 전송을 뜻하는 지에 따라서 데이터를 판독하는 방법과 기입하는 방법을 결정할 수 있다. 소스 및 데스티네이션 전송에 대하여 다른 포맷을 규정할 수 있음을 주목해야 한다.

차원화된 전송들은 가장 간단한 형태이나 역시 가장 고정된 형태이다. 차원화된 전송은, 데이터 바이트들의 간단한 연속 선형 시퀀스일 수 있고, 또한 다수의 그와 같은 영역들로 구성될 수 있는 소스나 데스티네이션을 설명한다. 어드레싱 메카니즘은 3차원에 이르는 어레이가 규정될 수 있게 한다. 이는 다수의 2-차원 패치들이 한번의 패킷 전송에 의하여 전송될 수 있게 한다. 제1 또는 A 차원을 따르는 데이터는 언제나 1 바이트 떨어져 있다. 제2 또는 B 차원 및 제3 또는 C 차원을 따르는 간격은 임의적이나 전 패킷에 대하여는 고정된다, 데스티네이션 차원들이 완료된 때에 전송이 완료된다.

제17도는 차원화된 전송이 소스나 데스티네이션 메모리를 액세스하는 방법의 예를 설명한다. 제17도는, 각각이 512개의 인접 8 비트 픽셀들로 구성된, 3 라인의 2개의 패치로 구성된 패킷을 도시한다. 이는, 예를 들어, 2 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)가 각각 라인의 패치 하나 상에서 작용하는 3×3 컨벌루션(3 by 3 convolution)을 수행하려는 경우에 필요할 수 있다. 제1 패치(" PQR")는 디지탈 영상/그래픽 프로세서(71)와 연관된 데이터 메모리(22)로 전송될 데이터를 나타낼 수 있으며, 제2 패치(" STU")는 디지탈 영상/그래픽 프로세서(72)와 관련된 데이터 메모리(27)로 전송될 데이터를 나타낼 수 있다. 이러한 예는 전송 제어기(80)가 소스 메모리 영역으로부터 데이터를 판독하는 방법을 규정하기 때문에 소스 전송을 나타낸다. 데이터 패킷은 다음의 파라메타에 따른 용어에 의하여 규정된다: A 카운트; B 카운트; C 카운트; 시작 어드레스; B 피치; C 피치. A 카운트는 제1 차원의 연속된 데이터 바이트들의 수이다. 제17도의 예에서는 A 카운트는 512이다. B 카운트는 패치를 형성하거나 라인의 수를 하나보다 작게 만들기 위한 단계의 수이다. 제17도는 2인 B 카운트를 도시한다. C 카운트는 패킷을 하나 작게 만들기 위한 패치-단계의 수이다. 제17도는 1인 C 카운트를 도시한다. 시작 어드레스는 패치의 시작의 선형 어드레스이다. 시작 어드레스는 " P" 로 나타내진 바이트의 어드레스이다. B 피치는 제2 차원의 리니어 피치, 즉 바이트 " P" 및 " Q" 또는 " Q" 및 " R" 의 어드레스의 차이이다. C 피치는 제3 차원의 리니어 피치, 바이트 " P" 와 " S" 의 어드레스의 차이이다. 소스 및 데스티네이션 전송 둘다 이러한 방법으로 정의되나, 파라메타들은 소스 및 데스티네이션의 형상이 아주 다를 수 있도록 독립적이다.

차원화된 전송의 모든 차원들이 활성 상태일 필요는 없다. B 카운트 및/또는 C 카운트를 0으로 설정함으로써, 전송이 개개의 바이트, 픽셀 멀티플 바이트, 라인, 또는 피치들에 제한될 수 있다. 0 값의 A 카운트는 전송될 데이터가 없다는 결과가 되며 이는 에러를 발생시킬 수 있다.

가이드 전송은, 차원 어드레스의 시퀀스가 패킷 전송 파라메타 내의 값들로부터 계산되지 않고 온칩 메모리의 테이블로부터 가이드되는 전송이다. 그들의 동작이 차원화된 전송보다는 좀더 복잡한 반면에, 그들은 훨씬 융통성이 있다. 가이드 전송에는 고정 패치 및 가변 패치의 두가지 클래스가 있다. 고정 패치 가이드된 전송은 차원화된 전송에서처럼 패킷 전송 파라메타 내에 기술되는 제1 및 제2 차원을 가지나, 제3 차원은 온칩 가이드 테이블의 엔트리로부터 가이드된다. 가변 패치 가이드된 전송에서는, 가이드 테이블은 역시 각 패치에 대한 A 및 B 차원의 크기를 결정한다. 어떠한 형태에서도, 제1의 2 차원들은 차원화된 전송과 유사한 방법으로 액티브상태가 된다. 따라서, 각각의 가이드된 테이블 엔트리들은, 제1의 2차원의 크기에 따라 개개의 바이트, 멀티플 바이트의 개개의 픽셀, 라인 또는 2-차원 패치를 이동시킬 수 있다. 이는 룩-업(look-up) 테이블을 통한 라인 그리기나 데이터를 처리하는 경우 발생할 수 있는 것과 같은 다수의 불규칙 동작(irregular)이 수행될 수 있게 한다.

다른 언급이 없다면, 가이드된 전송에 대한 다음 설명에 있는 사항은 소스 및 데스티네이션 전송 둘 모두에 똑같이 적용된다.

가이드 테이블은 전송 형태에 따라 32 비트 또는 64 비트일 수 있는 엔트리들의 블럭이다. 가이드 테이블은 온칩 메모리 내에 위치하여야만 한다. 고정된 패치 전송인 경우에는, 가이드 테이블이 32 비트 워드 어드레스에 정렬되어야 한다. 가변 패치 전송의 경우는, 가이드 테이블은 64 비트 더블 워드 어드레스에 정렬되어야 한다. 가이드 테이블의 시작 어드레스와 가이드 테이블이 갖는 엔트리의 수들은 해당 패킷 전송 파라메타 테이블 내에 표시된다. 각각의 가이드 테이블 엔트리들은 패킷 전송 내의 하나의 2-차원 패치에 대응한다. 전송 제어기(80)가 패킷 전송을 서비스하므로, 제어기는 패킷 전송 중에 다음의 패치를 프로세스할 필요가 있을 때에, 가이드 테이블 엔트리들을 차례로 하나씩 페치한다. 가이드 테이블 엔트리들에 관한 추가의 상세한 사항들이 아래의 상세한 가이드된 전송 설명에 설명되었다.

고정 패치 가이드된 전송은 32 비트 엔트리들을 갖는 온칩 가이드 테이블을 사용한다. 테이블은 정렬된 워드이어야 하며, 따라서 테이블의 엔트리 어드레스의 최하위 2 비트는 "00" 이어야 한다. 각각의 엔트리들은 전송의 제3 차원을 위한 어드레스를 계산하기 위하여 사용되는 정보들을 갖는다. 고정 패치 전송은 고정 패치 델타 가이드, 고정 패치 오프셋 가이드, 및 고정 패치 오프셋 가이드된 룩-업 테이블의 3가지 형태로 나타난다. 이러한 형태들이 아래에 자세히 설명될 것이다.

고정 패치 델타 가이드된 전송인 경우, 가이드 테이블은 현재 패치의 시작 어드레스를 만들기 위하여 이전의 2-차원 패치의 시작 어드레스에 가산될 32 비트 "델타" 값을 갖는다. 패치 크기는 A 카운트 및 B 카운트 패킷 전송 파라메타에 의하여 고정되고 정의된다.

고정 패치 델타 가이드된 패킷 전송 프로세스와 방법의 예가 제18도에 도시되었다. 이때 델타 A의 값은 패킷 전송 파라메타에 주어진 시작 어드레스에 가산되어, 제1 패치, 즉 패치 A의 시작 어드레스를 형성한다. 이러한 시작 어드레스는 최종 패치 시작 레지스터(504)에 저장된다. 그 다음으로, 델타 B가 최종 패치 시작 레지스터(504)에 저장된 패치 A의 시작 어드레스에 가산되어, 패치 B의 시작 어드레스를 형성하며, 계속 이와 같이 이루어진다. 그 합은 다음 패치 시작 어드레스의 근거를 만들기 위하여 항상 최종 패치 시작 레지스터(504)에 저장된다는 것을 주목해야 한다. 도 18에 도시된 바와 같이, 테이블 포인터(501)는 각각의 패치 후에 4바이트씩 증분되어, 가이드 테이블(502) 내의 다음 엔트리를 지시한다. 엔트리 수 카운터(503)가 각 패치 후에 1씩 감소된다. 패킷 전송은 엔트리 수 카운터(503)가 0에 도달한 때에 종료된다.

고정 패치 오프셋 가이드된, 패킷 전송은, 패킷 전송 파라메타에 주어진 베이스 어드레스에 가산되어 각 패치의 시작 어드레스를 형성하는 32 비트 값을 갖는 가이드 테이블(502)을 사용한다. 패치 크기는 고정되며, A 카운트 및 C 카운트 패킷 전송 파라메타에 의하여 정의된다.

제19도는 고정 패치 오프셋 가이드된 패킷 전송을 위한 어드레싱 메카니즘 프로세스와 방법을 도시한다. 가이드 테이블(502)의 제1 엔트리로부터의 값 델타 A는 패킷 전송 파라메타 테이블애 규정된 베이스 어드레스에 가산되고 베이스 어드레스 레지스터(505)에 저장되어 패치 A의 시작 어드레스를 형성한다. 그 다음으로, 델타 B가 베이스 어드레스 레지스터(505)에 저장된 어드레스에 가산되어 패치 B의 시작 어드레스를 형성하며, 이것은 가이드 테이블(502)의 최종 엔트리까지 계속된다. 패킷 전송 파라메타 테이블에 규정된 베이스 어드레스가 0인 경우, 가이드 테이블(502)은 절대 어드레스(absolute address)를 규정할 것이다. 델타 가이드된 전송에서와 같이, 테이블 포인터(501)는 현재의 가이드 테이블 엔트리를 지시하며, 엔트리 수 카운터(503)는 수행된 패치의 수를 추적한다.

고정 패치 오프셋 가이드된 룩업 테이블 패킷 전송인 경우, 가이드 테이블(502)은, 오프셋 레지스터(506) 내에서 0, 1, 2 또는 3 비트에 의하여 제로 필(zero fill)과 함께 좌향-시프트된 후, 베이스 어드레스 레지스터(505)에 저장된 어드레스에 가산될 32 비트 오프셋 값을 갖는다. 베이스 어드레스 레지스터(505)에 저장된 베이스 어드레스는 패킷 전송 파라메타에 주어진다. 이는 전송이 룩업 테이블의 데이터 크기에 무관하게 룩업 테이블 동작에 사용될 수 있게 한다. 고정 패치 오프셋 가이드된 룩업 테이블 포맷은 소스 전송에만 사용될 수 있다. 시프트 양은 패킷 전송 파라메타의 A 카운트 필드의 비트 1 내지 3의 최좌측 비트의 위치에 의하여 표시된다. 비트 3의 " 1" 은 3 자리, 비트 2는 2 자리, 비트 1은 1 자리의 좌향-시프트를 나타낸다. 비트 1 내지 3의 " 1" 은 0 시프트를 나타내지 않는다. 이러한 좌향 시프트는 8, 16, 32 및 64 비트 데이터 크기의 룩업 테이블에 대한 지원을 허용한다. 룩업 테이블 전송에 대한 패치 크기는 1차원으로 고정되며 1, 2, 4 또는 8 바이트로 설정된다.

제20도는 고정 패치 오프셋 가이드된 룩업 테이블 패킷 전송에 대한 어드레스 계산의 예를 도시한다. 이때, 오프셋 A는, A 카운트 필드의 값에 의하여 표시된 바와 같은 룩업 테이블 데이터 크기에 따라, 0, 1, 2 또는 3 비트씩 좌향으로 시프트된다. 오프셋 레지스터(504)의 이러한 시프트된 어드레스는 베이스 어드레스 레지스터(405)에 저장된 베이스 어드레스에 가산되어, 패치 A의 시작 어드레스를 형성한다. 다음으로, 오프셋 B 값이 시프트되고 베이스 어드레스 레지스터(405)의 베이스 어드레스에 가산되어, 패치 B 시작 어드레스를 형성하며, 계속 이와 같이 이루어진다. 시프트는 오프셋 값이 가이드 테이블(503)로부터 로드될 때 발생한다. 앞의 예에서 설명된 바와 같이, 테이블 포인터(501)와 엔트리 수 레지스터(502)는 가이드 테이블(503)의 현재 위치와 패치들의 수를 계속 추적한다.

가변 패치 가이드된 전송은 패킷 전송 파라메타보다는 가이드 테이블 내의 모든 패치 크기 정보를 규정한다. 이는 패킷 전송 내의 각각의 패치들이 다른 차원을 가질 수 있게 한다. 전송은 델타 또는 오프셋 가이드될 수 있다. 가변 패치 가이드된 전송의 경우, 가이드 테이블은 64 비트의 더블-워드 엔트리들로 구성된다. 가이드 테이블을 위한 리틀 엔디언 포맷이 제21도에 도시되었다. 더블 워드의 하반부는 제1의 2차원에 대한 A 및 B카운트 값을 갖는다. 상위 32 비트는 제3차원의 어드레스를 계산하는데 사용되는 값을 갖는다. 제22도는, 상위 32 비트가 A 및 B 카운트 값을 포함하고 하위 32 비트가 오프셋 또는 델타를 포함하는 빅 엔디언 가이드 테이블 포맷을 도시한다.

32 비트 기입을 사용하여 엔디언 형식에 무관하게 가이드 테이블을 생성하도록 소프트웨어가 적절히 제공된다. 워드 1이 워드 0보다 4 바이트 큰 어드레스에 있는 경우, A 및 B 카운트는 워드 1 어드레스에 나타나고, 오프셋/델타 어드레

스는 워드 0 어드레스에 나타난다. 전송 제어기(80)는 64 비트 액세스 동안 한번에 32 비트 워드 둘다를 액세스할 수 있고, 현재 선택된 엔디언 모드에 따라 정확한 내부 동작에 필요한대로 워드의 순서를 조절할 수 있다. 가변 패치 전송을 위한 가이드 테이블 엔트리는 정렬된 더블 워드여야 하는데, 즉 바이트 어드레스의 3개의 최하위 비트들이 "000" 이어야 한다.

제23도는 가변 패치 델타 가이드된 패킷 전송의 예를 도시한다. 가변 패치 델타 가이드된 패킷 전송의 경우, 가이드 테이블 엔트리의 제1 워드는 최종 패치 시작 레지스터(504)에 저장된 이전의 패치의 시작 어드레스에 가산될 32 비트 델타 양을 갖는다. 제1 패치를 위한 시작 어드레스는 패킷 전송 파라메타에 주어진다. 패치 크기는 가변적이고, 각각의 가이드 테이블 엔트리의 제2 워드 내에 규정된다. 이때 델타 W가 패킷 전송 파라메타에 규정된 시작 어드레스에 가산되어, 패치 W의 시작 어드레스를 형성한다. A 카운트는 제1 차원의 바이트 수의 길이를 결정하며, B 카운트는 라인의 수보다 1만큼 적은 제2 차원의 크기를 결정한다. 다음으로, 델타 X가 최종 패치 시작 레지스터(504)에 저장된 패치 W 시작 어드레스에 가산되어 패치 X 시작 어드레스를 발생시키며, 이와 같이 계속된다. 고정 패치 전송의 경우에서 이미 설명된 바와 같이, 테이블 포인터(501)와 엔트리 수 카운터(503)는 가이드 테이블(502) 내에서의 위치를 계속 추적한다. 각각의 가이드 테이블 엔트리가 64 비트 또는 8 바이트이기 때문에, 테이블 포인터(501)가 8 바이트 높은 어드레스를 지시하도록 증분된다는 점을 주목해야 한다.

가변 패치 오프셋 가이드된 패킷 전송의 예가 제24도에 도시되었다. 가변 패치 오프셋 가이드된 패킷 전송에 대한 각 가이드 테이블 엔트리의 한 워드는 32 비트 오프셋 값을 갖는다. 이러한 32 비트 오프셋 값은 각각의 패치의 시작 어드레스를 계산하기 위하여 베이스 어드레스 레지스터(505)에 저장된 베이스 어드레스에 가산된다. 베이스 어드레스는 패킷 전송 파라메타에 규정된다. 패치의 크기는 가변적이고, 각 가이드 테이블 엔트리의 나머지 다른 반(half)에 규정된다. 오프셋 W 값은 베이스 어드레스 레지스터(505)에 저장된 베이스 어드레스에 가산되어, 패치 W의 시작 어드레스를 발생시킨다. A 카운트 및 B 카운트 값들이 패치의 크기를 결정한다. 다음으로, 오프셋 X가 베이스 어드레스 레지스터(505)에 저장된 최초의 베이스 어드레스에 가산되어 패치 X에 대한 시작 어드레스를 얻고, 이와 같이 계속된다. 테이블 포인터(501)와 엔트리 수 카운터(503)는 테이블 포인터(501)가 8씩 증가되는 상태로 가이드 테이블(502)의 위치를 계속 추적한다.

룩업 테이블 패킷 전송과 같은 필-위드-밸류 패킷 전송은 소스 전송만을 위하여 규정될 수 있다. 필-위드-밸류 패킷 전송은 실질적으로 소스 메모리로부터 데이터를 전송하지 않는 대신에, 패킷 전송 파라메타 내에 소스값을 규정한다. 최상위 필 값 워드 및 최하위 필 값 워드인 2개의 32 비트 필드들은, 데스티네이션 메모리를 채우기 위하여 사용될 64 비트 값을 규정한다. 필 패턴(fill pattern)이 63 비트보다 작은 경우, 패턴은 최하위 및 최상위 필 값 워드 전부에 대해 복사되어야만 한다. 필 값에 대해서는 어떠한 정렬 동작도 수행되지 않으며, 데스티네이션 더블-워드에 기입된 바이트들은 필 값 더블-워드로부터의 대응하는 바이트들이다. 필-위드-밸류 패킷 전송을 위하여는 어떠한 소스 시작 어드레스나 소스 차원 카운트도 규정되지 않는다. 패킷 전송의 크기가 데스티네이션 전송 파라메타에 의하여 결정된다.

다양한 전송 포맷은 다양한 소스와 데스티네이션 전송 조합을 가능하게 한다. 이러한 조합들을 규정하기 위하여, 패킷 전송 파라메타 테이블의 파라메타 포맷은 요청된 소스 및 데스티네이션 전송의 형태에 따라 변경될 것이다. 파라메타 테이블 내용에 대한 검토를 돕기 위하여, 제25도 내지 제29도가 가능 파라메타의 범위의 예로서 도시되었다.

패킷 전송 파라메타 테이블들은, 마스터 프로세서(60)의 파라메타 메모리(15); 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 파라메타 메모리(25, 30, 35 또는 40); 또는 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74)의 데이터 메모리(22, 23, 24, 27, 28, 29, 32, 33, 34, 37, 38 또는 39)와 같은 온칩 메모리에 위치하는 것이 바람직하다. 패킷 전송 파라메타 테이블은 정렬된 64 바이트, 즉 6개의 최하위 어드레스 비트들이 "00000" 이어야 한다. 어떠한 메

모리가 테이블을 갖느냐에 대해서는 제한이 없다. 따라서, 마스터 프로세서(60)가 디지탈 영상/그래픽 프로세서(71)에 대응하는 파라메타 메모리(25)에 위치된 파라메타 테이블을 사용할 수 있으며, 이하 마찬가지이다. 요청 프로세서는, 패킷 전송 요청을 제공하기 전에, 프로세서 자신의 파라메타 메모리 내의 링크 리스트 시작 어드레스 위치 내에서 적절한 시작 어드레스를 위치시킨다.

제25도 내지 제28도와 관련하여, 다음의 부분은 패킷 전송 파라메타의 다양한 필드들을 설명한다. " PT" 로 표시된 패킷 전송 파라메타의 시작 어드레스에 대한 각 필드의 어드레스뿐만 아니라, 그 필드가 유효한 전송 형태도 주어진다. 많은 경우, 2개의 동일한 필드들이 있게 되는데, 하나는 소스 전송을 위한 것이며 하나는 데스티네이션 전송을 위한 것이다. 이러한 경우, 필드 설명이 개개의 경우에 따라 주어질 것이다. 전송 형태에 따라서, 특정의 필드는 사용되지 않는다. 이러한 경우에는, 상기 필드는 프로그래밍되지 않고 남겨진다. 표 14는 모든 허용되는 옵션들의 목록이다. 이러한 점들이 아래에 상세히 설명될 것이다.

[표 14]

동 작	소스	데스티네이션
차원화된 경우	예	예
고정 패치 델타 가이드된 경우	예	예
고정 패치 오프셋 가이드된 경우	예	예
고정 패치 오프셋 가이드된 LUT인 경우	예	아니오
가변 패치 델타 가이드된 경우	예	예
가변 패치 오프셋 가이드된 경우	예	예
필-워드-밸류의 경우	예	아니오
투과 차원화된 경우	예	아니오
투과 고정 패치 델타 가이드된 경우	예	아니오
투과 고정 패치 오프셋 가이드된 경우	예	아니오
투과 가변 패치 델타 가이드된 경우	예	아니오
투과 가변 패치 오프셋 가이드된 경우	예	아니오
블럭 기입 경우	아니오	예

패킷 전송 파라메타의 제1 데이터 워드는 다음 파라메타 테이블의 다음 패킷 전송 시작 어드레스이다. 이러한 워드는 모든 패킷 전송 파라메타에 존재한다. 이것은 패킷 전송 링크 리스트 상의 다음 엔트리의 시작에 대한 32 비트 포인터이다. 이것은 64 바이트 정렬 온칩 어드레스를 지시하며, 따라서 이러한 어드레스의 6개 최하위 비트들은 " 000000" 이어야 한다. 패킷 전송 옵션 필드의 스톱 비트가 링크 리스트를 종료시키기 위하여 사용되기 때문에, 링크 리스트의 최종 엔트리는 특별한 다음 어드레스 엔트리를 필요로 하지 않는다. 패킷 전송이 성공적으로 종료할 때마다, 다음 패킷 전송 엔트리 어드레스 워드의 값이, 요청 프로세서의 파라메타 메모리의 링크 리스트 시작 어드레스 위치에 기입된다. 이는 자동적으로 포인터를 다음 패킷 전송으로 전진시킨다. 포인터는 링크 리스트 상의 최종 패킷이 완료된 후에 갱신되어, 스톱 비트가 링크 리스트 실행을 " 일시 정지(pause)" 시키는데 이용되는 경우, 링크 리스트가 다시 인에이블될 때

포인터가 링크 리스트 상의 다음 패킷 전송을 지시하게 한다.

제2 워드는 패킷 전송 옵션 필드이다. 이는 PT + 4에 위치된다. 패킷 전송 옵션 필드는, 데이터가 전송되는 방식에서 다양한 옵션들을 규정하기 위하여 모든 패킷 전송 형태에서 사용된다. 이러한 패킷 전송 옵션 필드가 아래에 상세히 검토될 것이다.

차원화된 델타 가이드 패킷 전송에 사용된 제3 및 제4 데이터 워드들은 소스 및 데스티네이션 시작 어드레스들이다. 소스 시작 어드레스는 PT + 8에 위치되고 데스티네이션 어드레스는 PT + 12에 위치된다. 이러한 필드들은 차원화된 델타 가이드 패킷 전송에 사용된다. 차원화된 패킷 전송에 대하여, 각각의 필드들은 소스나 데스티네이션에 대한 시작 바이트 어드레스를 나타내는 32 비트 워드들이다. 델타 가이드 전송에 대하여, 각각의 필드들은 제1 델타 오프셋이 소스나 데스티네이션 전송을 위하여 가산되는 시작 어드레스를 나타낸다.

오프셋 가이드된 패킷 전송에 사용된 제3 및 제4 데이터 워드들은 소스 및 데스티네이션 오프셋들이다. 소스 시작 어드레스는 PT + 8에 위치되고 데스티네이션 어드레스는 PT + 12에 위치된다. 이러한 필드들은, 오프셋 가이드된 전송 모드들이 사용된 때에, 소스 및 데스티네이션 시작 어드레스를 대체하는 32 비트 오프셋 값들이다.

제5 및 제6 데이터 워드들은 소스 및 데스티네이션 A 및 B 카운트들이다. PT + 16에 있는 제5 데이터 워드는 비트 15 - 0에 있는 소스를 위한 A 카운트와 비트 31 - 16에 있는 소스를 위한 B 카운트를 갖는다. 유사하게, PT + 20에 있는 제 6 데이터 워드는 비트 15 - 0에 있는 데스티네이션을 위한 A 카운트와 비트 31 - 16에 있는 데스티네이션을 위한 B 카운트를 갖는다. A 카운트 16 비트 필드는 차원화된 또는 고정 패치 전송을 위한 소스 또는 데스티네이션의 제1 차원에 해당하는 곳으로 전송될 바이트의 수를 규정한다. A 카운트 필드는 가변 패치 가이드된 전송에는 사용되지 않는다. 또한, 소스 A 카운트는 필-위드-밸류 패킷 전송에 대해서도 사용되지 않는다. B 카운트 16 비트 필드는 차원화된 또는 고정 패치 전송을 위한 소스 또는 데스티네이션의 제2 차원에 해당하는 곳에서 발생되어야 할 " 단계" 의 수를 규정한다. 이러한 값은 라인 - 1과 같은 값이다. 따라서 0인 값은 제 2 차원을 디스에이블하며, 패치당 1 라인만 전송하게 한다. B 카운트 필드는 가변 패치 전송동안에는 사용되지 않는다. 소스 B 카운트는 필-위드-밸류 패킷 전송 동안 사용되지 않는다.

차원화된 패킷 전송에서 PT +24 및 PT +28에 위치된 제7 및 제8 데이터 워드들은 각각 소스 및 데스티네이션 C 카운트들이다. 이러한 32 비트 필드들은 소스 또는 데스티네이션의 제3 차원에서의 " 패치-단계" 의 수를 규정한다. 이러한 값들은 패치의 수 - 1과 같은 값들이다. 따라서, 0인 값은 제3 차원을 디스에이블시키며, 단 하나의 패치만이 전송되게 한다. 소스 C 카운트 필드는 필 전송을 위해서는 사용되지 않는다.

델타 또는 오프셋 가이드된 패킷 전송에서의 제7 및 제8 데이터 워드들은 엔트리 수이다. 이러한 32 비트 필드의 값들은 각각 소스 및 데스티네이션을 위한 가이드 테이블의 엔트리 수를 규정한다. 따라서 이러한 필드들은 전송될 정보의 패치 수를 나타낸다. 0의 값은 데이터 전송을 발생치 않는다.

제9 및 제10 데이터 워드들은 각각 PT + 32 및 PT + 36에 위치된다. 차원화 된 고정 패치 패킷 전송에서는, 이러한 32 비트 필드들이 소스 및 데스티네이션의 제2 차원의 피치를 규정한다. 이러한 피치 값은 소스 또는 데스티네이션 라인의 시작 어드레스에 가산되어, 다음 라인의 시작 어드레스를 얻는다. 대응하는 B 카운트 필드에 있는 값이 0인 경우, 이러한 필드는 프로그래밍되지 않고 남겨질 수 있다.

제11 및 제12 데이터 워드들은 PT + 40 및 PT + 44에 각각 위치된다. 차원화된 패킷 전송에서는, 이러한 32 비트 필드들이 소스 또는 데스티네이션의 제3 차원의 피치를 규정한다. 이러한 피치 값이 소스 또는 데스티네이션 패치의 시작 어드레스에 가산되어 다음 패치의 시작 어드레스를 얻는다. 해당 C 카운트 필드의 값이 0인 경우, 이러한 필드는 프로그래밍되지 않고 남겨질 수 있다.

델타 또는 오프셋 가이드된 패킷 전송에서는, 제11 및 제12 데이터 워드들은 가이드 테이블 어드레스들이다. 이러한 필드들은 각 가이드 테이블의 제1 엔트리를 지시하는 정렬된 어드레스로 로드된다. 이러한 워드들의 값은, 엔트리가 가이드 테이블로부터 얻어질 때마다, 고정 패치 전송인 경우는 4씩, 가변 패치 전송인 경우는 8씩 자동적으로 증가된다. 이러한 가이드 테이블 시작 어드레스들은 고정 패치 패킷 전송인 경우는 32 비트 워드 어드레스로, 가변 패치 패킷 전송인 경우에는 64 비트 더블-워드 어드레스로 정렬되어야 한다. 가이드 테이블은 온칩 메모리 내에 있어야만 한다. 불법 어드레스 에러 인터럽트는, 상태 조건 중 어느 하나가 참이 아닌 경우 발생할 것이다.

제9 및 제11 데이터 워드는 필-위드-밸류 패킷 전송에서는 특별하게 사용된다. 이러한 2개의 32 비트 필드들은 함께 64 비트 필 패턴을 정의한다. 데스티네이션 더블-워드에 기입되는 바이트들은 필 값 더블-워드 내의 해당 바이트들로부터 취해진다. 따라서 필 패턴이 64 비트보다 작은 경우, 데이터 값들은 이러한 2 데이터 워드들 내에 반복되어야 한다. 필-위드-밸류 패킷 전송인 경우, 데스티네이션이 차원화, 고정 또는 가변 패치, 델타 또는 오프셋 가이드될 수 있는 반면에, 필 값은 단지 소스에만 허용된다.

패킷 전송이 소스 투과(source transparency)를 사용하는 경우, 제13 및 제 14 데이터 워드들은 투과 값을 유지한다. 이러한 64 비트 더블 워드는 PT + 48에서 시작한다. 이러한 64 비트 필드는 투과가 선택된 때 비교될 값을 나타내기 위하여 사용된다. 패킷 전송 옵션 필드의 패킷 액세스 모드 필드는, 투과 크기나 이러한 필드 내에 포함된 값들의 수를 나타낸다. 데스티네이션 더블-워드에 기입되려고 하는 바이트들은 투과 값 내의 해당 바이트들에 대해서 비교되고, 매칭이 이루어진 경우에는 바이트 스트로브들이 비활성 상태로 구동된다. 픽셀 크기가 64 비트보다 작은 경우, 데이터 값들이 복사되어야만 한다. 이러한 필드는 블럭 기입을 제외한 모든 비-투과(non-transparency) 전송에 대해서는 사용되지 않는다. 소스 투과 값은 엔디언 모드에 상관없이 메모리에 기입된 대로 전송 제어기(80)에 의하여 사용되고, 따라서 워드-교체는 발생하지 않는다. 이는 제25도에 도시되었다. 패킷 전송 파라메타 테이블 시작 어드레스에 후속하는 48 바이트를 어드레스하기 위하여 상기 값을 더블-워드 기입으로서 항상 기입하여, 혼동을 피할 수 있다.

패킷 전송이 블럭 기입 모드를 사용하는 경우, 제13 및 제14 데이터 워드들은 컬러 레지스터 데이터를 유지한다. 64 비트 컬러 레지스터 필드는 블럭 기입 사이클을 준비함에 있어서 비디오 랜덤 액세스 메모리(video random access memory)의 컬러 레지스터를 로드하기 위하여 사용되는 값을 포함한다. 이는 모의 블럭 기입(simulated block write)을 준비할 때도 역시 사용된다. 전송 제어기(80)는 투과 값에 대하여 위에 설명된 바와 같은 방법으로 엔디언에 상관없이 기입된 것과 완전히 동일한 컬러 레지스터 값을 사용한다. 이는 제29도에 도시되었다. 이러한 필드를 사용하는 것은 아래에 상세히 설명된다.

패킷 전송 파라메타의 최종 더블-워드를 구성하는, PT + 56에서 시작하는 제 15 및 제16 데이터 워드들은 현재 모든 전송 모드에 대해서 사용되지 않으며, 프로그래밍되지 않고 남겨진다. 이러한 데이터 워드들은 멀티프로세서 집적 회로(100)의 장래의 버전(version)에서 사용될 수 있다.

제25도 내지 제29도는 패킷 전송 파라메타 테이블에 대한 약간의 가능 포맷을 도시한다. 제25도는 소스 투과를 갖는 소스 및 데스티네이션의 차원화된 패킷 전송을 위한 파라메타 테이블의 예를 도시한다. 투과 워드 0이 PT + 48에 저장되고 투과 워드 1이 PT + 52에 저장된다는 것을 주목해야 한다. 제26도는 필-위드-밸류 모드의 데스티네이션 차원화된 패킷 전송을 위한 파라메타 테이블의 예를 도시한다. 필 값의 최상위 비트들은 PT + 40에 저장된다. 필 값의 최하위 비트들은 PT + 32에 저장된다. 제27도는 차원화된 소스 및 고정 패치 가이드된 데스티네이션 패킷 전송을 위한 파라메타 테이블의 예를 도시한다. PT + 12에 있는 데이터 워드는 베이스 어드레스를 저장하며, PT + 28에 있는 데이터 워드는 가이드 테이블의 엔트리의 수를 저장하고, PT + 44에 있는 데이터 워드는 가이드 테이블의 시작 어드

레스를 저장한다는 것을 주목해야 한다. 제28도는 차원화된 소스 및 가변 패치 가이드된 데스티네이션 패킷을 위한 파라메타 테이블의 예를 도시한다. 보통 PT + 20에 저장된 데스티네이션 A 및 B 카운트들은 데스티네이션 가이드 테이블의 일부분이기 때문에, 프로그래밍되지 않는다. 제29도는 차원화된 블럭 기입 패킷 전송을 위한 파라메타 테이블의 예를 도시한다. PT + 48에서 시작하는 더블 데이터 워드가 컬러 레지스터 워드 0 및 1을 포함한다는 것을 주목해야 한다.

패킷 전송 옵션 필드는 소스 밑 데스티네이션 전송에 어떠한 전송 형식이 사용될 것인지를 선택하며, 현재의 패킷이 링크 리스트를 종료시키는지를 결정한다. 그것은 또한 패킷 전송 종료시 추가의 어드레스 계산을 수행하며, 차원 어드레스 계산의 방향을 변경시키고, 또는 소스 및 데스티네이션 전송들을 반대로 만드는 등의 특별한 전송 모드들과 같은 다수의 추가적인 특징들을 선택할 수 있게 만든다. 패킷 전송 옵션 필드의 포맷이 제30도에 도시되었다. 옵션 필드를 모두 0으로 로드하는 것은 특별한 어드레싱 모드를 갖지 않는 소스 및 데스티네이션 상의 차원화된 전송들을 사용하는 디폴트(default) 패킷 전송을 제공함에 유의한다.

패킷 전송 옵션 필드의 비트 1 - 0들은 데스티네이션 갱신 모드 필드를 구성한다. 이러한 2개의 비트들은, 일단 패킷 전송이 종료된 후, 원래의 패킷 전송 파라메타 테이블 내의 데스티네이션 시작 어드레스들이 어떻게 갱신되는가를 나타낸다. 이러한 비트들이 0이 아닌 경우, 패킷 전송이 종료된 후에 별도의 어드레스 계산이 수행된다. 그 다음으로, 이러한 값은 패킷 전송 파라메타 테이블에 규정된 원래의 데스티네이션 시작 어드레스 위에 기입된다. 이는 가능하면 패킷 전송이 멈춘 곳으로부터 계속되어 다시 제공될 수 있게 한다. 이는 2개의 메모리 영역 사이에서 핑-퐁(ping-pong)을 하는 경우 특히 유용하다.

데스티네이션 갱신 모드들이 표 15에 열거되어 있다. 아래에 설명된 반전(reverse) 데스티네이션 C 또는 반전 데스티네이션 B와 같은 반전 어드레싱 비트들의 데스티네이션 중 하나가 설정되는 경우, 가산보다는 감산이 수행된다.

[표 15]

비트 1 0	데스티네이션 갱신 동작
0 0	갱신하지 않음
0 1	데스티네이션 전송의 최종 라인의 시작 어드레스에(로부터) B 피치를 가산(감산)하고 그 결과를 데스티네이션에 기입함
1 0	데스티네이션 전송의 최종 패치의 시작 어드레스에(로부터) C 피치를 가산(감산)하고 그 결과를 데스티네이션에 기입함
1 1	데스티네이션 전송의 최종 패치의 시작 어드레스에(로부터) C 피치를 가산(감산)하고, 그 결과를 데스티네이션 및 PT 파라메타 테이블의 시작 어드레스에 기입함, 그 다음에 PT 옵션 필드의 "반전 데스티네이션 C 어드레싱" 비트를 토글링(toggling)함

데스티네이션 갱신 모드는 사용된 제일 큰 차원에 대하여 하나의 별개의 단계를 수행하는데 사용될 수 있으나, 다른 유용한 동작들도 역시 수행될 수 있다. 모드 "10"은, 전송이 단지 2-차원인 경우에도 C 피치를 가산하는데 사용될 수 있다. 이는 다음의 2-차원 패킷이 최종의 것에 비교하여 위치될 수 있게 한다. 모드 "11"은 온칩 데이터 메모리와 같은 2개의 1-차원 또는 2-차원 패치들 사이에서 핑-퐁하는 패킷 전송을 다시 제공하는 데 특히 유용하다. 제3 차원을 어드레싱하는 방향은 패킷 전송 파라메타들이 종료시 갱신될 때마다 반전되기 때문에, 그것은 메모리 영역에서 교대로 발생할 수 있다.

이러한 기능은 주로 차원화된 전송을 사용하려고 의도되었다. 비록 그것은 가이드된 전송으로 규정될 수 있지만, 주의가 요구된다. 가이드된 전송에 있어서, 데스티네이션 C 피치는 데스티네이션 가이드 테이블 포인터로 대체된다.

패킷 전송 옵션 필드의 비트 6 - 4들은 데스티네이션 전송 모드 필드를 형성한다. 이러한 3개의 비트들은 데스티네이션 어드레싱에 어떠한 전송 형식이 사용될 것인지를 나타낸다. 코딩들이 표 16에 나타나 있다. 소스 특유 모드 룩업 테이블과 필-위드-밸류들이 데스티네이션 어드레싱을 위하여는 정의(보류)되지 않는다는 점을 주목해야 한다.

[표 16]

비트 6 5 4	데스티네이션 전송 모드
0 0 0	차원화된 경우
0 0 1	보류된 경우
0 1 0	보류된 경우
0 1 1	보류된 경우
1 0 0	가변 패치 델타 가이드된 경우
1 0 1	가변 패치 오프셋 가이드된 경우
1 1 0	고정 패치 델타 가이드된 경우
1 1 1	고정 패치 오프셋 가이드된 경우

패킷 전송 옵션 필드의 비트 9 - 8들은 소스 갱신 모드 필드를 형성한다. 소스 갱신 모드들이 표 17에 나타나 있다. 적절한 소스 반전 어드레싱 비트 반전 소스 C 또는 반전 소스 B가 설정되는 경우, 가산보다는 감산이 수행된다.

[표 17]

비트 9 8	소스 갱신 동작
0 0	갱신하지 않음
0 1	소스 전송의 최종 라인의 시작 어드레스에(로부터) B 피치를 가산(감산)하고 그 결과를 소스에 기입함
1 0	소스 전송의 최종 패치의 시작 어드레스에(로부터) C 피치를 가산(감산)하고 그 결과를 소스에 기입함
1 1	소스 전송의 최종 패치의 시작 어드레스에(로부터) C 피치를 가산(감산)하여 그 결과를 소스와 PT 파라메타 테이블의 시작 어드레스에 기입하고; PT 옵션 필드의 "반전 소스 C 어드레싱" 비트를 토글시킴

이러한 2개의 비트들은 원래의 패킷 전송 파라메타의 소스 시작 어드레스가 일단 패킷 전송이 종료된 후 어떤 값으로 갱신되어야만 하는 지를 나타낸다. 이러한 비트들이 0이 아닌 경우, 별도의 소스 어드레스 계산이 패킷 전송 종료 후에 수행된다. 다음으로, 이 값이 패킷 전송 파라메타에 규정된 원래의 소스 시작 어드레스 위에 기입된다. 이는 패킷 전송이 가능한 한 이전 시간에 중단된 곳으로부터 연속하여 다시 시작될 수 있게 한다. 이는 2개의 메모리 영역 사이의 핑-퐁에 특히 유용하다. 이러한 모드들은 데스티네이션 어드레스 발생에서 이미 설명된 것과 동일하게 소스 어드레스 발생에서 사용된다.

패킷 전송 옵션 필드의 비트 14 - 12들은 소스 전송 모드 필드를 형성한다. 이러한 3개의 비트들은 소스 어드레싱에 어떠한 전송 형식이 사용될 것인지를 나타낸다. 코딩들이 표 18에 나타나 있다.

[표 18]

비트 14 13 12	소스 전송 모드
0 0 0	차원화된 경우
0 0 1	필-위드-밸류인 경우
0 1 0	보류된 경우
0 1 1	고정 패치 오프셋 가이드된 LUT의 경우
1 0 0	가변 패치 델타 가이드된 경우
1 0 1	가변 패치 오프셋 가이드된 경우
1 1 0	고정 패치 델타 가이드된 경우
1 1 1	고정 패치 오프셋 가이드된 경우

패킷 전송 옵션 필드의 비트 18 - 16들은 패킷 전송 액세스 모드 필드를 형성한다. 이러한 3개의 비트들은 표 19에 나타난 것과 같은 특별한 액세스 모드들을 엔코딩하기 위하여 사용된다. 이러한 모드들은 소스 데이터가 데스티네이션에 기입되는 방법을 변형시킨다. 온칩 메모리 데스티네이션에 대하여는, 모드 " 000" 의 정상적인 전송만이 허용된다.

[표 19]

비트 18 17 16	패킷 전송 액세스 모드
0 0 0	통상의 페이지-모드
0 0 1	주변 장치 전송
0 1 0	블럭-기입
0 1 1	직렬-레지스터 전송
1 0 0	8 비트 소스 투과
1 0 1	16 비트 소스 투과
1 1 0	32 비트 소스 투과
1 1 1	64 비트 소스 투과

"000" 인 코딩은 외부 메모리를 위한 정상 액세스 모드 또는 페이지 - 모드 액세스를 가능케 한다. 소스 또는 데스티네이션에는 특별한 어드레싱 모드들이 사용되지 않는다. 데이터는 변경없이 소스로부터 데스티네이션으로 전송된다.

"001" 인 코딩은 주변 장치 전송 모드를 인에이블한다. 주변 장치 모드는 전송 제어기(80)를 메모리 제어기로 사용하여 멀티프로세서 집적 회로(100)의 외부에 있는 다른 장치들이 메모리를 판독하거나 메모리에 기입하게 한다. 메모리의 장치 판독은 소스를 프로그래밍하여 이루어진다. 메모리에 대한 장치 기입은 데스티네이션을 프로그래밍하여 이루어진다. 둘 중의 어느 한 경우, 전송 제어기(80)는 메모리 어드레스를 구동하며 전송 파라메타에 따라 발생된 어드레스로 라인을 정상적으로 제어하나 데이터 버스를 고임피던스(high impedance) 모드로 두어, 주변 장치가 데이터를 판독 또는 구동시킬 수 있도록 한다. 주변 장치 모드는 필 - 위드 - 밸류를 제외한 소스 또는 데스티네이션 전송의 어떠한 형태로도 사용될 수 있다.

"010" 인 코딩은 패킷 전송이 VRAM 블럭 기입을 사용할 수 있도록 허용한다. 이는 전송 제어기(80)가 VRAM 컬리 레지스터를 로드시킨 후, VRAM 블럭 기입 모드를 사용하여 외부 메모리로 데스티네이션 기입을 수행하게 한다. 이러한 모드에서는, 소스 데이터가 VRAM의 어느 위치가 VRAM 컬러 레지스터 데이터로 기입될 것인지를 규정하는 블럭 기입 어드레스 마스크 비트들을 나타낸다. 이러한 비트들은 통상의 어드레싱을 사용하여 소스로부터 페칭되고 블럭 기입 모드를 사용하여 데스티네이션 VRAM에 순차적으로 기입된다. VRAM 컬러 레지스터에 로드된 값은 패킷 전송 파라메타의 컬러 레지스터 값으로 규정된다.

블럭 - 기입 동작은 오프칩 데스티네이션에만 지원된다. 온칩 데스티네이션 어드레스에 블럭 - 기입하려는 시도는 패킷 전송이 에러 상태로 지연되도록 만든다.

"011" 인 코딩은 직렬 - 레지스터 전송 모드를 인에이블한다. 직렬 - 레지스터 전송 모드는 전형적으로 VRAM의 벌크(bulk) 초기화에 사용된다. 소스 어드레스는 VRAM 로우를 VRAM 직렬 시프트 레지스터 안으로 복사하는데 사용된다. 한편 데스티네이션 어드레스들은 시프트 레지스터를 다수의 VRAM 메모리 로우 안으로 복사하는데 사용된다. 정상적인 동작은 1 바이트의 값으로 소스 A 카운트를 로드시키며, n이 기입될 로우의 수인 경우, 데스티네이션을 1인 A 카운트나 n - 1인 B 카운트로 로드시키는 것이다. 이러한 모드로는 데이터 버스나 크로스바(50)에 대하여 어떠한 데이터 전송도 발생할 수 없고 모든 데이터 전송은 VRAM과 함께 발생한다는 점을 주목해야 한다. 모든 소스 및 데스티네이션 액세스들은 비 - 페이지(non - page)모드일 것이다. 직렬 - 레지스터 전송 동작은 오프칩 소스 및 데스티네이션에 대해서만 지원된다.

"1XX" 형태의 코딩들은 투과를 인에이블한다. 소스 및 데스티네이션 차원화 또는 가이드된 전송들은 정규적으로 수행된다. 그러나 데스티네이션 데이터가 기입되기 전에, 데이터는 패킷 전송 파라메타에 주어진 투과 값과 비교된다. 이러한 투과 비교는 투과 및 바이트 기입 회로(353)에서 발생한다. 패킷 액세스 모드의 2개의 최하위 비트들은 투과 데이터의 크기를 나타낸다. 따라서 하나의 64 비트, 2개의 32 비트, 4개의 16 비트 또는 8개의 8 비트 비교가 행해진다. 어떠한 비교라도 참이라면, 다음으로 투과 및 바이트 기입 회로(353)가 해당 바이트 - 스트로브를 디스에이블시켜, 데스티네이션 바이트들이 기입되지 않도록 만든다. 온칩 소스 또는 데스티네이션 어드레스로 투과를 시도하는 것은 패킷 전송이 오류 상태로 지연되도록 만든다.

패킷 전송 옵션 필드의 비트 19는 소스 및 데스티네이션 파라메타 비트의 교환을 뜻한다. 비트 19를 설정하는 것은 소스 및 데스티네이션 파라메타를 수동적으로 교환하지 않고 패킷 전송의 방향이 반전되게 한다. 이는 데이터가 자신의 원래의 위치로 복귀할 때 유용하다. 교환 비트가 설정된 때에는, 전송 제어기(80)가 패킷 전송 파라메타를 로드할 때의

모든 소스 및 데스티네이션 값들, 시작 어드레스, 피치, 카운트, 가이드 테이블 포인터 및 필 값을 교환시킨다. 표 20은 이러한 비트가 설정되는 경우 발생할 32 비트 교환을 도시한다. " PT" 가 패킷 전송 파라메타 테이블의 다음 엔트리 어드레스를 나타낸다는 점을 주목해야 한다.

[표 20]

바이트 어드레스		바이트 어드레스
Src 시작 어드레스 PT + 08	<= =>	PT + 12 Dst 시작 어드레스
Src A/B 카운트 PT + 16	<= =>	PT + 20 Dst A/B 카운트
Src C 카운트 PT + 24	<= =>	PT + 28 Dst C 카운트
Src B 피치 PT + 32	<= =>	PT + 36 Dst B 피치
Src C 피치 PT + 40	<= =>	PT + 44 Dst C 피치

PT에 있는 다음 엔트리 어드레스 데이터와 PT + 04에 있는 패킷 전송 옵션 필드는 관련된 소스 또는 데스티네이션이 아니기 때문에, 그들의 이전 위치에 남아 있는다는 점을 주목해야 한다. 또한, 투과 데이터 또는 컬러 레지스터 데이터를 저장하는 PT + 48 및 PT + 52들은 교환되지 않는다. 이는 64 비트 투과 또는 컬러 레지스터 파라메타가 자신의 값을 유지할 수 있도록 허용한다.

소스 및 데스티네이션 관련 파라메타 워드를 교환하는 것에 덧붙여서, 전송 제어기(80)는 역시 패킷 전송 옵션 필드 안에 있는 소스 및 데스티네이션 관련 비트들을 교환한다. 이는 표 21에 나타나 있다.

[표 21]

비트 번호		비트 번호
	0 <= =>	8
Dst 갱신 모드	1 <= =>	9 Src 갱신 모드
	4 <= =>	12
Dst 전송 모드	5 <= =>	13 Src 전송 모드
	6 <= =>	14
Dst 반전 B 어드레싱	24 <= =>	21 Src 반전 B 어드레싱
Dst 반전 C 어드레싱	25 <= =>	22 Src 반전 C 어드레싱

교체 중 어느 것이 지원되지 않는 기능이라면, 패킷 전송은 오류 상태로 지연될 것이다. 소스 및 데스티네이션 파라메타의 교환이 패킷 전송 파라메타들이 로드될 때마다 수행된다. 패킷 전송이 지연된다면, 현재의 파라메타들은 요청 프로세서의 파라메타 메모리로 세이브되기 전에 그들의 원래의 위치로 다시 교환될 것이다. 지연된 패킷 전송이 복구된다면, 파라메타들이 전송 제어기(80)에 의하여 그들이 로드될 때 다시 교환될 것이다.

만일 갱신 모드 중의 하나가 패킷 전송 옵션 필드에 규정된다면, 상기 동작이 패킷 전송이 종료된 때 정상적으로 발생할 것이다. 예를 들어 만일 소스 갱신 동작이 선택된다면, 원래의 패킷 전송 파라메타의 소스 시작 어드레스는 패킷 전송 동안 데스티네이션 시작 어드레스로 실제 사용되는 경우에도 갱신될 것이다. 마찬가지로, 갱신 모드로서 " 토글 반전 소스 C 어드레싱" 을 규정하는 것은 원래의 패킷 전송 옵션 필드의 비트 22를 토글되도록 만드는데, 이는 패킷 전송이 다시 제공되는 경우 데스티네이션 C 어드레싱이 실제적으로 반전되도록 만든다.

위의 예가 나타낸 바와 같이 X 비트를 사용할 때에는 주의하여야 한다. 차원화된 데스티네이션 패킷 전송에 필-위드-밸류를 위하여 규정된다면, 예를 들어 필-위드-밸류가 데스티네이션 동작으로 규정될 수 없기 때문에, 패킷 전송은 에러로 지연된다. 에러가 생기지 않은 경우에도, 데스티네이션 시작 어드레스 및 카운트가 전형적으로 프로그래밍되지 않는 소스 값으로 로드될 것이다. 유사하게 필값 워드는 데스티네이션 피치로 로드될 것이다. 이는 매우 무의미한 패킷 전송을 유발할 수 있다.

패킷 진송 옵션 필드의 비트 21은 반전 소스 B 어드레싱 비트이다. 이러한 비트를 " 1" 로 설정시키는 것은 소스의 제2 차원을 후방으로 어드레싱시키게 된다. B 피치는 이전 라인 시작 어드레스에 가산되기보다는 이전 라인 시작 어드레스로부터 감산된다. 이러한 비트가 " 0" 이라면, 패킷 전송 소스 어드레싱이 정상적으로 발생한다.

패킷 전송 옵션 필드의 비트 22가 반전 소스 C 어드레싱 비트이다. 이러한 비트를 " 1" 로 설정시키는 것은 소스의 제3 차원을 후방으로 어드레싱시키게 된다. C 피치는 이전 패치 시작 어드레스에 가산되는 대신에 이전 패치 시작 어드레스로부터 감산된다. 이러한 비트가 " 0" 이라면, 패킷 전송 소스 어드레싱이 정상적으로 발생한다. 이러한 비트가 가이드 테이블을 통하여 C 어드레싱을 규정하는 가이드된 전송에 대해서는 아무 의미도 갖지 않는다는 것을 주목해야 한다.

패킷 전송 옵션 필드의 비트 23은 반전 A 어드레싱 비트이다. 이러한 비트를 " 1" 로 설정시키는 것은 소스 및 데스티네이션의 제1 차원을 후방으로 어드레싱시키게 된다. 더블-워드 어드레스는 증분되는 대신에 감소된다. 이는 B 피치가 더해질(감산될) 값이 제1 차원의 최상위 어드레스임을 의미한다. 더블-워드 내의 바이트 어드레싱은 반전되지 않고, 더블-워드 어드레싱만이 반전된다는 점을 주목해야 한다.

패킷 전송 옵션 필드의 비트 24는 반전 데스티네이션 B 어드레싱 비트이다. 이러한 비트를 " 1" 로 설정시키는 것은 데스티네이션의 제2 차원을 후방으로 어드레싱시키게 된다. B 피치는 이전 라인 시작 어드레스에 가산되는 대신에, 이전 라인 시작 어드레스로부터 감산된다. 이러한 비트가 " 0" 이라면, 패킷 전송 소스 어드레싱이 정상적으로 발생한다.

비트 25는 반전 데스티네이션 C 어드레싱 비트이다. 이러한 비트를 " 1" 로 설정시키는 것은 데스티네이션의 제3 차원을 후방으로 어드레싱시키게 된다. C 피치는 이전 패치 시작 어드레스에 가산되는 대신에, 이전 패치 시작 어드레스로부터 감산된다. 이러한 비트가 " 0" 이라면, 패킷 전송 소스 어드레싱이 정상적으로 발생한다. 이러한 비트가 가이드 테이블을 통하여 C 어드레싱을 규정하는 가이드된 전송에 대해서는 아무 의미도 갖지 않는다는 것을 주목해야 한다.

패킷 전송 옵션 필드의 비트 28은 비트를 종료했을 때의 인터럽트이다. 이러한 비트를 " 1" 로 설정시키는 것은, 링크 리스트 상의 엔트리가 종료되자마자 패킷 전송을 초기화시키는 프로세서로 " 정상" 인터럽트가 보내지게 만들 것이다. 링크 리스트는 엔트리를 더 포함할 수 있다. 이는 링크 리스트의 특정 포인트에 도달한 때에 요청 프로세서를 플래그(flag)되게 한다. 이러한 비트가 " 0" 이라면, 링크 리스트의 해당 패킷 엔트리가 종료된 때 어떠한 인터럽트도 프로세서로 보내지지 않는다. 따라서 이러한 비트 세트를 만나거나 종료될 때까지, 또는 링크 리스트의 끝에서의 패킷 전송이 종

료될 때까지는 어떠한 인터럽트도 발행되지 않는다. 그러나 언제라도 오류가 생긴다면, 전송 제어기(80)는 즉시 요청 프로세서에 오류-인터럽트를 보낼 것이다.

패킷 전송 옵션 필드의 비트 30 - 29들은 패킷 전송 상태 필드를 형성한다. 패킷 전송 상태 필드는 패킷 전송 요청의 상태를 반영한다. 이러한 비트들은, 프로세서가 요청을 제공할 때에는 언제나 " 00" 으로 설정되어야 한다. 링크 리스트 안에서의 어떠한 패킷 전송이라도 지연된다면, 전송 제어기(80)는 지연된 패킷 전송 영역으로 세이브하는 패킷 전송 옵션 필드의적절한 패킷 전송 상태 비트를 설정시킬 것이다. 이는 지연된 패킷 전송이 프레쉬(fresh) 패킷 전송보다 많은 파라메타를 포함하고 따라서 다르게 세이브 및 복구되기 때문에 필요하다. 이러한 비트중 하나가 전송 제어기(80)가 패킷 전송 파라메타를 로드할 때 전송 제어기(80)에 의하여 " 1" 로 된다면, 패킷 전송은 지연되며, 따라서 모든 별도의 파라메타들이 복구된다는 것을 알아야 한다. 이러한 필드는 표 22에 나타난 바와 같이 엔코딩된다.

[표 22]

비트	패킷 전송 상태
30 29	
0 0	지연되지 않는다
0 1	지연되나 폴트되지 않는다
1 0	지연되고 소스 상에서 폴트된다
1 1	지연되고 데스티네이션 상에서 폴트된다

전송 제어기(80)는, 전송 제어기(80)에 의하여 높은 우선 순위의 패킷 전송 요청이 수신된 때, 패킷 전송이 크고 타임-아웃된 때, 패킷 전송을 요청한 프로세서가 패킷 전송을 지연시킬 것을 요청한 때, 또는 오류 상태가 생긴 때 중 어느 하나가 발생하여 패킷 전송이 지연된 때, 이러한 비트들을 " 01" 로 기입한다. 비트 30은 실질적으로 오류 발생으로 인하여 패킷 전송이 지연되었음을 나타낸다. 한편 비트 29는 오류가 소스 또는 데스티네이션 중 어디에 있는가를 나타낸다. 이러한 정보는 오류 상태를 해결하기 위하여 마스터 프로세서(60)에 의하여 필요된다. 그것은 지연 및 복구 프로세스가 모든 지연된 패킷 전송 형태에 대하여 동일하기 때문에 전송 제어기(80)가 오류 패킷 전송의 파라메타들을 다시 로드할 때 전송 제어기(80)와는 무관하다.

소스 및 데스티네이션 파라메타 교환 비트(비트 19)가 지연 패킷 전송 옵션 필드에서 " 1" 이라면 오류 패킷 전송에 대한 비트 29의 의미가 반전된다. 따라서 " 10" 은 데스티네이션 상의 오류를, " 11" 은 소스 상의 오류를 나타낸다. 따라서 마스터 프로세서(60) 소프트웨어는 오류 어드레스 위치를 결정할 때 비트 29 및 19 둘다를 검사해야 한다.

전송 제어기(80)는 패킷 전송동안 오류가 발생하는 경우 여러 동작을 취해야 한다. 전송 제어기(80)는 비트 29 및 30을 적절히 설정한다. 전송 제어기(80)는 FLTSTS 레지스터에 적절한 비트를 설정한다. 전송 제어기(80)는 마스터 프로세서(60)로 오류 인터럽트를 발행한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74) 중의 하나가 패킷 전송을 발행할 때, 이러한 프로세서는 오류가 발생했다는 사실을 모를 것이다. 마스터 프로세서(60)는 오류 상태를 클리어하며 FLTSTS 레지스터의 오류 플래그를 클리어한다. FLTSTS 레지스터의 해당 오류 플래그를 클리어한때, 전송 제어기(80)는 자동적으로 패킷 전송을 다시 발행한다.

패킷 전송 옵션 필드의 비트 31은 스톱 비트이다. 이러한 비트는 링크 리스트의 " 끝" 을 표시하기 의하여 사용된다. 패킷 전송이 이러한 비트가 " 1" 인 곳을 만나면, 패킷 전송은 완료되고 링크 리스트는 종료될 것이다. 종료 전, 다음 엔트리 어드레스 필드가 요청 프로세서의 파라메타 메모리 안의 링크 리스트 시작 어드레스 위치 안으로 복사된다. 따라서, 링크 리스트가 다시 사용 가능해지면, 링크 리스트의 다음 엔트리에서 실행이 시작될 것이다. 이는 핑-퐁과 같은 반복된 동작에 특히 유용한 순환하는 링크 리스트를 발생시키는 것을 허용한다. 스톱 비트는 역시 원하는 위치에서 링크 리스트를 멈추게 하기 위하여 사용될 수 있다.

일단 전송 제어기(80)가 파라메타 테이블로부터 패킷 전송 파라메타들을 판독하면, 데이터를 전송하는 것을 언제라도 시작할 수 있게 된다. 이러한 것을 하기 위하여, 전송 제어기(80)는 크로스바 또는 외부 메모리 액세스 중의 어느 하나 또는 둘 다를 발생하여야만 한다. 데스티네이션 데이터 흐름 가능성(flow possibility)에 대한 4개의 소스 - 온칩 대 온칩, 온칩 대 오프칩, 오프칩 대 온칩, 오프칩 대 오프칩 - 가 있게 된다. 제일 마지막의 것은 특별한 경우인데 반하여 앞의 3개들은 " 정상적으로" 다루어진다.

정상 패킷 전송 데이터 흐름동안, 소스 머신(320)은 전송 파라메타에 근거된 소스 어드레스를 발생시키며 이러한 어드레스를 온칩 또는 오프칩 메모리로부터 데이터를 페치하는데 사용한다. 일단 데이터가 크로스바나 외부 메모리 버스로부터 수신되었다면, 요청된 바이트들이 추출되며 소스 정렬 논리(332)에 의하여 정렬되고, 그 다음으로 패킷 전송 FIFO 버퍼(311)에 놓여진다. 이와 동시에, 데스티네이션 머신(340)이 데스티네이션 메모리를 위한 어드레스를 역시 발생시킨다. 일단 패킷 전송 FIFO 버퍼(311)가 다음 데스티네이션 메모리 액세스를 위해 요청된 바이트의 수를 가지면, 데스티네이션 머신(340)은 요청된 크로스바나 외부 메모리 사이클을 발생시킨다.

패킷 전송 FIFO 버퍼(311)는 데이터 흐름을 제어하고 소스 머신(320)을 데스티네이션 머신(340)과 동기 상태로 유지시키는 역할을 맡는다. 어떤 때에라도 패킷 전송 FIFO 버퍼(311)가 다음 데스티네이션 액세스를 위한 충분한 소스 바이트를 포함하지 않는다면, 데스티네이션 머신(340)은 데이터가 사용 가능할 때까지 멈춰진다. 같은 방법으로, 패킷 전송 FIFO 버퍼(311)가 가득 차면, 소스 머신(320)은 데스티네이션 머신(340)이 다음 소스 액세스를 완료하기에 충분한 바이트들을 다 써버릴 때까지 소스 머신(320)은 멈춘다. 이는 소스 전송이 데스티네이션 전송을 과도하게 구동시키는 것을 막는다. 특별한 패킷 전송 액세스 모드를 위하여, 이러한 데이터 흐름이 약간 수정될 수 있다.

크로스바 및 외부 메모리 인터페이스들이 독립적이기 때문에, 소스 전송은 외부 버스 상에서 생길 수 있으며 데스티네이션 전송은 크로스바 또는 그 역으로 병렬적으로 생길 수 있다. 온칩 대 온칩 전송에 대하여, 소스 및 데스티네이션이 필요에 따라 크로스바 인터페이스와 인터리브(interleave) 사이클을 공유할 수 있다.

캐시 서비스 요청, 직접 외부 액세스 요청, 프레임 제어기(90) 요청, 긴급 리프레시, 및 호스트 요청들은 패킷 전송 지연을 야기시키지 않는다. 이러한 요청들은 소스 머신(320) 및 데스티네이션 머신(340) 중 둘다 또는 어느 하나를 중단시킬 수 있으나, 이는 요청이 패킷 전송보다 높은 우선 순위를 갖는 경우일 때뿐이다. 프레임 제어기(90) 및 긴급 리프레시 사이클은 단지 외부 메모리 인터페이스만을 사용한다. 따라서 패킷 전송 크로스바 액세스들은 발생을 계속할 수 있다. 소스 머신(320) 또는 데스티네이션 머신(340)중 어느 하나가 외부 메모리 인터페이스를 사용하고 있다면, 패킷 전송 FIFO 버퍼(311)는 결국에는 꽉 차거나 텅비거나 둘 중 어느 한 상태가 될 것이다. 따라서 크로스바를 사용하는 제어기는 외부 메모리 인터페이스가 다시 사용 가능해질 때까지 멈추어야만 한다. 전송이 온칩 대 온칩이면 소스 머신(320) 및 데스티네이션 머신(340) 모두 방해되지 않는 상태가 계속된다. 캐시 및 직접 외부 액세스 서빙은 크로스바 및 외부 인터페이스 모두를 사용하며, 이에 따라 그들은 전형적으로 그들이 요청하는 사이클 수 동안 어떠한 패킷 전송을 멈추게 만들 것이다.

오프칩 소스 대 오프칩 데스티네이션 패킷 전송들은 나머지 다른 세가지 경우의 패킷 전송과는 다르게 처리된다. DRAM 및 VRAM 상의 페이지 모드의 장점을 취하기 위하여, 전송 제어기(80)는 오프칩 소스로부터 온칩으로 컬럼 액세스의 페이지 모드 버스트를 수행한 후, 온칩으로부터 오프칩 데스티네이션으로의 또 다른 페이지 모드 버스트를 수행한다. 이러한 것은 온칩 버퍼를 사용하는 것을 요청한다. 각각의 프로세서들은 제15도에 도시된 바와 같이 이러한 목적을 위하여 보류된 자신의 해당 파라메타 메모리에 있는 128-바이트 영역을 갖는다. 제15도에 도시된 바와 같이, 마스터 프로세서(60)에 의하여 요청된 오프칩 대 오프칩 패킷 전송들은 어드레스가 Hex "01010100" 에서 Hex "0101017F" 까지를 사용한다. 디지탈 영상/그래픽 프로세서(71, 72, 73 또는 74) 중 어느 하나에 의하여 요청된 오프칩 대 오프칩 패킷 전송들은 #이 표 3에 나열된 바와 같이 디지탈 영상/그래픽 프로세서 번호에 해당하는 경우 어드레스 Hex "0100#100" 내지 Hex "0100#17F" 를 사용한다. 이러한 파라메타 메모리 버퍼로/로부터의 데이터 전송은 전송 제어기(80)에 의하여 처리되며 사용자가 알 수 있게 된다.

패킷 전송 파라메타들의 조합은 완료되기 위하여 매우 긴 시간이 걸리는 매우 긴 전송을 규정하는 것을 허용할 수 있다. 또한 높은 패킷 전송 요청이 자신의 파라메타를 훨씬 초과한 로드를 갖는 것을 금지시키는 패킷 전송을 인터럽트하는 것을 계속한다. 이러한 상황중의 하나가 발생하는 것을 막는 것을 돕기 위하여, 전송 제어기(80)는 제31도에 도시된 패킷 전송 최소 레지스터 PTMIN(511) 및 패킷 전송 최대 레지스터 PTMAX(512)인 2개의 24 비트 레지스터를 포함하는 타이머를 포함한다. 이러한 레지스터들은 패킷 전송의 최소 및 최대 길이를 규정한다. 패킷 전송 카운터 PTCOUNT(513)은 패킷 전송이 실행된 클럭 사이클 수를 나타낸다.

패킷 전송 최소 레지스터 PTMIN(511)은 패킷 전송이 높은 우선 순위 패킷 전송 요청에 의하여 인터럽트될 수 있기 전에 패킷 전송이 실행되어야 할 최소 클럭 사이클 수를 나타내기 위하여 사용된다. 파라메타가 로드된 후에 패킷 전송이 시작할 때, 타이머 시퀀서(515)는 패킷 전송 최소 레지스터 PTMIN(511)에 저장된 값을 패킷 전송 카운터 PTCOUNT(513)으로 로드한다. 패킷 전송 카운터 PTCOUNT(513)은 전송 제어기(80)에 의하여 패킷 전송이 능동적으로 제공되고 있는 각각의 클럭 사이클 동안 1만큼씩 감소된다. 패킷 전송 카운터 PTCOUNT(513)은 패킷 전송 과정 동안에 발생할 지도 모르는 캐시-제공, 프레임 제어기(90), 호스트 또는 리프레시 사이클 동안은 감소되지 않으나, 능동 패킷 전송 서비스 동안 일어나는 재시도 또는 대기 상태 동안은 감소된다. 그러나, 그것은 크로스바 패킷 전송 액세스들이 여전히 발생한다면, 외부 버스 상의 프레임 제어기(90), 호스트, 또는 리프레시 액티브 동안 감소될 것이다. 패킷 전송은 제로 감지기가 PTCOUNT가 0에 도달했음을 타이머 시퀀서(515)에 신호할 때까지 높은 우선 순위 패킷 전송 요청에 의하여 인터럽트될 수 있다. 그러나, 그것은 요청 프로세서에 의하여 지연될 수 있다. 또한 그것은 오류 또는 폴트 상태에 의하여 역시 지연될 수 있다. 패킷 전송 최소 레지스터 PTMIN(511)은 재설정시 Hex "10000" (65,536 사이클)으로 로드된다.

패킷 전송 최소 레지스터 PTMIN(511)의 한 중요한 쓰임은 이미 지연된 패킷 전송의 파라메타들을 다른 메모리 영역으로 전송하기 위한 "중지시킬 수 없는" 패킷 전송을 만드는 것이다. 다른 지연이 데이터가 그로부터 전송되고 있는 파라메타 메모리 영역을 재기입할 수 있기 때문에, 패킷 전송 최소 레지스터 PTMIN(511)은 패킷 전송이 완료될 수 있음을 확인하기 위하여 사용된다. 이러한 것이 파라메타 메모리 영역이 재기입되도록 야기시킬 수도 있기 때문에 그와 같은 전송이 잘못된 것이 아니라는 사실이 중요하다. 이러한 특색은 높은 우선 순위 메모리 액세스가 최소의 시간 기간 전에 패킷 전송을 멈추는 것을 방지하며, 결과적으로 최소의 데이터 전송이 발생한다.

패킷 전송 최대 레지스터 PTMAX(512)는 한 패킷 전송이 데이터의 전송을 독점하는 것을 방지하기 위하여 사용된다. 일단 PTMIN 시간이 경과되고 패킷 전송 카운터 PTCOUNT(513)이 제로 감지기(514)에 의하여 감지된 때 0으로 감소되면, 타이머 시퀀서(515)는 패킷 전송 최대 레지스터 PTMAX(512)에 저장된 값을 패킷 전송 카운터 PTCOUNT

(513)으로 로드한다. 이는 전송이 타임 아웃되기 전에 진행될 수 있는 시간의 남은 기간을 정의한다. 따라서 패킷 전송을 인터럽트하기 위한 최대 지속 기간은 PTMIN + PTMAX 클럭 사이클이 된다. 패킷 전송 카운터 PTCOUNT(513)은 패킷 전송이 리프레시와 같은 비-패킷 전송에 대한 능동적 배제를 계속하는 각각의 사이클 동안 감소된다. 제로 감지기(514)가 패킷 전송 카운터 PTCOUNT(513)의 값이 패킷 전송 완료 전에 0에 도달한 것을 감지한다면, 패킷 전송은 타임 아웃된 것으로 간주된다. 타이머 시퀀서(515)는 패킷 전송을 지연시키며, 전송 제어기(80)는 라운드 로빈에서의 같은 우선 순위의 다음 요청 상으로 이동한다. 같은 우선 순위의 다른 요청들이 보류 중이지 않다면, 패킷 전송은 같은 우선 순위 요청 또는 높은 우선 순위 요청이 발생할 때까지, 또는 전송이 완료될 때까지 0인 PTCOUNT를 계속 유지시킬 수 있다. 높은 우선 순위 패킷 전송 요청이 패킷 전송 최소 레지스터 PTMIN(511)에 규정된 수의 사이클이 경과된 후 수신된다면, 능동 패킷 전송은 PTMAX가 도달되었는지에 상관없이 지연될 것이다. 패킷 전송 최대 레지스터 PTMAX(512)는 재설정시 Hex " 1000" (65,536 사이클)으로 로드된다.

PTMIN 및 PTMAX 값들이 패킷 전송이 로드된 때마다 인가된다. 따라서 그들은 링크 리스트 안에 있는 각각의 패킷 전송으로 인가된다. 패킷 전송이 지연된다면, 전체의 링크 리스트가 지연된다. 라운드 로빈 토큰은 링크 리스트 안의 다음 패킷 전송으로 진행하는 것이 아니라 긴급한 요청을 갖는 다음의 프로세서로 진행한다. 패킷 전송이 타임 아웃된 때, 지연된 전송의 상태는 요청 프로세서의 파라메타 메모리로 세이브된다. 전송 제어기(80)는 자동적으로 라운드 로빈 우선 순위가 상기 프로세서로 복귀된 때 계속을 위한 이러한 요청을 다시 내놓는다. 지연된 패킷 전송이 다시 시작된 때에는, 꽉찬 PTMIN 및 PTMAX 값들이 효과를 나타낼 것이다.

양호한 실시예에서는 패킷 전송 최소 레지스터 PTMIN(511)과 최대 레지스터 PTMAX(512)들은 단지 24 비트들을 포함한다. 따라서 패킷 전송 서비스가 시간 아웃 없이 계속될 수 있는 최대 시간은 50 MHz의 타깃 동작 주파수(target operation frequency)인 경우 거의 0.67 초가 된다.

패킷 전송 링크 리스트들은 전송 제어기(80)에 의하여 다음과 같이 운영된다. 리프레시, 프레임 제어기(90), 호스트 인터페이스 또는 캐시 서비스 요청이 긴급 우선 순위와는 다른 패킷 전 동안 수신된다면, 패킷 전송 파라메타들의 상태가 전송 제어기(80)의 내부 레지스터들 내에 있게 되고, 필요하다면 소스 및/또는 데스티네이션 전송이 멈춰질 것이다. 높은 우선 순위 요청들이 제공된다면, 패킷 전송은 다시 시작될 것이다.

패킷 전송이 높은 우선 순위 패킷 전송 요청, 타임 아웃, 요청 프로세서로부터의 지연 요청, 폴트 또는 오류에 의하여 인터럽트된다면, 패킷 전송은 지연된다. 요청 프로세서의 파라메타 메모리 내의 링크 리스트 시작 어드레스가 세이브된 패킷 전송 파라메타들을 지시하기 위하여 변경된다. 이러한 인터럽션이 높은 우선 순위 패킷 전송에 의한 것이라면, 라운드 로빈 토큰은 인터럽트된 패킷 전송으로 멈추게 되며, 이에 따라 낮은 우선 순위 레벨이 다시 시작된 때에 자신의 서비스가 다시 시작되게 된다. 높은 우선 순위 요청이 패킷 전송 파라메타들이 로드되는 동안 생긴다면, 로드는 중단될 것이다. 어떠한 지연도 생기지 않는다. 패킷 전송 파라메타들이 높은 우선 순위 패킷 전송이 완료된 때에 원래의 파라메타 리스트로부터 로드될 것이다. 패킷 전송이 타임 아웃, 폴트, 오류 또는 지연 요청 등 때문에 지연된다면, 라운드 로빈 토큰이 인터럽트된 패킷 전송을 우선 순위화 체인(prioritization chain)의 끝에 위치시키는 단계로 진행한다.

패킷 전송이 완료되고 패킷 전송 옵션 필드의 인터럽트 비트가 " 1" 인 때에는, 전송 제어기(80)는 요청 프로세서에게 " 패킷 종료" 인터럽트를 내놓는다. 링크 리스트 안에서 " 1" 인 패킷 전송 옵션 필드의 스톱 비트를 갖는 최종 패킷 전송이 완료된 때에는, 전송 제어기(80)는 요청 프로세서에게 " 패킷 종료" 인터럽트를 내놓는다. 패킷 전송이 완료된 때, 패킷 전송 파라메타로부터 다음-어드레스 필드가 패킷 전송의 패킷 전송 옵션 필드 스톱 비트가 설정된 경우에도 요청 프로세서의 파라메타 메모리 안의 링크 리스트 시작 어드레스 위치로 기입된다.

패킷 전송 파라메타들이 로드된 때, 전송 제어기(80)는 패킷 전송 옵션 필드의 패킷 전송 상태 비트들을 체크한다. 이러한 비트들이 패킷 전송이 지연되었음을 나타낸다면, 지연된 전송에 대한 추가 상태 정보들이 로드된다. 이러한 것들이 아래에 더 설명된 것이다.

패킷 전송이 오류 상태에 빠진다면, 전송 제어기(80)는 즉시 데이터를 전송시키는 것을 중단하며 지연을 수행한다. 전송 제어기(80)는 요청 프로세서의 PTERROR 플래그를 설정한다. 또한 전송 제어기(80)는 링크 리스트가 종료되었음을 나타내기 위하여 요청 프로세서의 COMM 레지스터(120)의 Q 비트를 설정한다. 요청 프로세서는 세이브된 데이터들로부터 오류의 원인을 판정하도록 프로그래밍될 수 있다.

어떤 수의 상태들도 패킷 전송 동안 오류의 원인일 수 있다. 블럭 기입 데스티네이션 전송 또는 온칩 메모리 내에 소스 또는 데스티네이션을 갖는 직렬 레지스터 전송과 같은 시도된 VRAM 액세스 모드는 오류를 야기시킬 수 있다. 투과를 사용하는 시도된 온칩 데스티네이션 액세스가 오류의 원인일 수 있다. 오류는 패킷 전송 요청의 데스티네이션 전송의 총 바이트의 길이가 필-위드-밸류, 시프트 레지스터 전송 또는 주변 장치 전송의 경우를 제외하고 자신의 소스 전송의 길이보다 길 때에 생길 것이다. 오류는 룩업 테이블 데스티네이션 동작이 시도된 때 발생할 것이다. 이는 " 소스 및 데스티네이션 파라메타 교환" 동작으로부터 생기는 결과일 수 있다. 오류는 링크 리스트 시작 어드레스 또는 링크 리스트의 다음 엔트리가 오프칩 어드레스를 지시할 때에 생길 것이다. 오류는 패킷 전송 파라메타 테이블 및 링크 리스트 시작 어드레스가 64-바이트 경계로 정렬되지 않은 때에 생길 것이다. 오류는 소스 또는 데스티네이션 가이드 테이블 포인터가 오프칩 메모리를 지시하거나 적절히 정렬되지 않은 때에 시도된 패킷 요청 상에서 발생한다. 실제의 오류 상태를 나타내기 위하여는 어떠한 상태 비트들도 세이브되지 않으며, 이는 지연된 파라메타의 상태로부터 추론되어야 한다.

링크 리스트 시작 어드레스 또는 링크 리스트 어드레스 오프칩의 다음 엔트리, 잘못 정렬된 패킷 전송 파라메타 테이블 또는 오프칩 메모리에 대한 또는 잘못 정렬된 것에 대한 가이드 테이블 포인터들로 인해 야기된 오류는 요청 프로세서의 파라메타 메모리의 지연 영역으로 패킷 전송 파라메타를 세이브하지 못하게 할 것이다. 이는 패킷 전송이 어떠한 데이터를 전송시키기 전에 포기하거나 결코 실제적으로 시작하지 않기 때문이다. 따라서, 오류의 원인을 진단하려고 시도할 때에는, 링크 리스트의 유효성이 지연된 파라메타들이 유효한 것들인지를 확인하기 위하여 반드시 체크되어야 한다.

전송 제어기(80)는 다수의 상태에서 패킷 전송을 지연시킬 것이다. 높은 우선 순위 패킷 전송 요청을 접수한 때에, 전송 제어기(80)는 현재의 패킷 전송을 위해 PTMIN에 규정된 수의 사이클이 끝났다면 패킷 전송을 지연시킬 것이다. 전송 제어기(80)는, 사이클의 수가 PTMIN + PTMAX와 타임 아웃을 초과하며 다른 프로세서로부터 같은 우선 순위의 패킷 전송 요청이 처리되어야 한다면, 패킷 전송을 지연시킬 것이다. 전송 제어기(80)는 요청 프로세서가 자신의 COMM 레지스터(120)의 S 비트를 통하여 패킷 전송을 지연시킬 것을 요청한다면 패킷 전송을 지연시킬 것이다. 전송 제어기(80)는 소스 어드레싱이나 데스티네이션 어드레싱동안에 메모리 폴트가 발생한다면 패킷 전송을 지연시킬 것이다. 전송 제어기(80)는 오류 상태가 감지된다면 패킷 전송을 지연시킬 것이다. 전송 제어기(80)는 외부 메모리 액세스 동안 재시도가 생기고 다른 프로세서로부터의 같은 우선 순위의 패킷 전송 요청이 대기 중이며 현재의 패킷 전송을 위해 PTMIN 내에 규정된 수의 사이클이 끝났다면 패킷 전송을 지연시킬 것이다.

이러한 경우의 각각에서의 지연 메카니즘은 동일하다. 전송 제어기(80)는 현재의 패킷 전송 파라메타들을 세이브하며 전송 제어기(80)의 내부 상태를 세이브한다. 이러한 정보는 장래에 패킷 전송이 계속될 수 있도록 허용한다. 파라메타들은 요청 프로세서의 파라메타 메모리의 지연 영역에 저장된다. 제15도에서 도시된 바와 같이, 마스터 프로세서(60)가 패킷 전송을 요청했다면 이러한 영역은 Hex " 01010000" 내지 Hex " 0101007F" 까지이다. 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 중의 어느 하나에 의하여 요청된 패킷 전송을 위해 지연된 패킷 파라메타들은 #이 표 3에 나열된 바와 같이 디지탈 영상/그래픽 프로세서 번호에 해당하는 경우 어드레스 Hex " 0100#000" 내지 " 0100#07F" 까지를 사용한다. 이러한 파라메타들을 위한 포맷이 제32도 및 제33도에 도시되었다. 제32도에서 " %" 로 표시

된 더블-워드들이 현재의 엔디언에 맞추어진 것이 아니라는 점을 주목하여야 한다.

지연은 지연 조건이 발생한 후 즉시 시작되며 긴급한 외부 메모리 컬럼 액세스들도 완료된다. 요청 프로세서의 파라메타 메모리 상의 외부-대-외부 버퍼는 비워지지 않으나 패킷 전송 FIFO 버퍼(311)의 현재 상태는 세이브된다. 이러한 구조는 원인에 상관없이 일정한 방법으로 패킷 전송이 지연되도록 허용한다. 또한 패킷 전송 FIFO 버퍼(311)가 텅빈 상태가 발생하지 않기 때문에 역시 확실히 신속하게 이루어질 것이다. 따라서 긴급 우선 순위 패킷 전송 요청은 패킷 전송 FIFO 버퍼(311)를 비우기 위하여 내부적으로 느린 패킷 전송을 대기하지 않고 신속하게 제공될 수 있다.

파라메타 메모리 안으로의 지연은, 지연의 원인이 높은 우선 순위 대기 요청인 경우, 전송 제어기(80) 크로스바 우선 순위 레벨에서 수행된다. 따라서 디지탈 영상/그래픽 프로세서(71, 72, 73 및 74) 또는 마스터 제어기(60)는 지연이 완료될 때까지 일시적인 내용들을 경험한다. 지연이 타임 아웃, 폴트, 오류 또는 요청 프로세서로부터의 지연 요청에 기인된 때에는, 패킷 전송의 원래 전송 제어기(80) 크로스바 우선 순위로 수행될 것이다.

지연된 패킷 전송 파라메타들은 다른 곳에서 복사될 수 있으며 링크 리스트 시작 어드레스 포인터가 새로운 어드레스를 지시하기 위하여 변형되었다면 그들의 새로운 위치로부터 다시 제출될 수 있다. 패킷 전송 옵션 필드의 패킷 전송 상태 비트들이 패킷 전송이 지연되었음을 나타내기 때문에, 지연 파라메타들의 전체 집합이 그들이 위치한 위치에 상관없이 다시 제출된 즉시 로드될 것이다. 지연된 파라메타의 새로운 시작 어드레스는 128 바이트 경계인 경우에도 정렬되어야 하며 따라서 7개의 최하위 어드레스 비트들은 지연된 패킷 전송이 다시 제출되기 위하여서는 "0000000" 이여야 한다.

패킷 전송 지연 영역으로 세이브된 처음의 8개의 64 비트 더블-워드들은 "정상" 패킷 전송 파라메타들을 나타낸다. 이러한 것들은 약간을 제외하고는 사용자에 의하여 프로그래밍된 것과 같은 파라메타들이다. 처음의 32 비트 워드는 링크 리스트 상의 다음 엔트리 어드레스라기보다는 지연된 패킷 전송의 원래의 엔트리 어드레스를 포함한다. C 카운트 필드들은 차원화된 전송을 위한 현재의 C 카운트들을 포함한다. 가이드된 전송에 대해서는, C 카운트 필드들이 가이드 카운트들과 가이드 테이블 포인터의 현재의 값을 갖는 가이드 테이블 포인터를 가질 것이다. 또한, 패킷 전송 옵션 필드는 파라메타들이 지연된 패킷 전송을 나타낸다는 사실을 반영하도록 변형된 자신의 패킷 전송 상태 필드를 갖는다.

"정상"의 경우에 추가하여, 사용자 프로그래밍된 패킷 전송 파라메타의 경우는 지연 영역이 역시 8개의 반전된 "내부 상태" 더블 워드들을 갖는다. 이러한 필드들은 지연된 패킷 전송이 다시 시작될 때에 자동적으로 로드되는 필드들이다. 그들은 지연된 때의 패킷 전송의 차원 상호간에 대한 정보를 갖는다. 이러한 값들은 지연된 패킷 전송이 다시 제출된 때에 중단된 곳에서 정확히 시작할 수 있도록 세이브된다. Hex "040"의 어드레스 끝의 데이터 워드는 현재의 소스 A 및 B 카운트를 저장한다. 이러한 것들은 소스 머신(320)의 ACURRENT 및 BCURRENT로부터의 16 비트 크기의 양이다. B 카운트는 비트 31 - 16에 저장되며 A 카운트는 비트 15 - 0에 저장된다. Hex "044"의 어드레스 끝의 데이터 워드는 현재 데스티네이션 A 및 B 카운트들을 저장한다. 이러한 것들은 데스티네이션 머신(340)의 ACURRENT 및 BCURRENT 레지스터로부터의 16 비트 크기의 양들이다. B 카운트는 비트 31 - 16에 저장되며 A 카운트는 비트 15 - 0에 저장된다. Hex "048"의 어드레스 끝의 데이터 워드는 현재 소스 A 시작 어드레스 또는 현재 가이드 테이블 어드레스이다. 지연된 패킷 서비스 파라메타의 해당 P 비트는 이곳에 어떤 데이터가 저장될 것인지를 나타낸다. 가이드된 테이블 페치는 원래의 B 카운트가 0에 도달할 때까지 일어나지 않으며, 따라서 가이드된 패킷 전송조차도 소스 A 시작 어드레스를 이곳에 저장시킬 수 있다는 점을 주목해야 한다. Hex "04C"의 어드레스 끝의 데이터 워드는 현재의 데스티네이션 A 시작 어드레스 또는 현재의 가이드 테이블 패칭된, 지연된 패킷 서비스 파라메타들의 해당 P 비트에 의하여 결정된 동일성을 갖는 값이다. Hex "050" 및 "056"의 어드레스들의 끝의 데이터 워드들은 각각 현재의 데스티네이션 B 시작 어드레스 또는 해당 가이드 테이블 페치된 값들이다.

Hex "058" 의 어드레싱 끝에서 시작하는 64 비트 더블 워드는 지연된 패킷 서비스 파라메타들을 저장한다. 지연된 패킷 서비스 파라메타들의 포맷이 제32도 및 제33도에 도시되었다. 상위 데이터 워드의 비트 31 - 30들은 P 비트들이다. 비트 31의 " 1" 은 지연된 패킷 전송 파라메타들이 가이드 테이블로부터 페치된 소스 머신(320)의 ASTSRT 레지스터 및 BSTSRT 레지스터 값을 저장한다는 것을 나타낸다. 그렇지 않다면, 이러한 레지스터들은 차원화된 전송으로부터 값들을 저장한다. 마찬가지로, 비트 30의 " 1" 은 지연된 패킷 전송 파라메타들이 가이드 테이블로부터 페치된 소스 머신(320)의 ASTSRT 레지스터 및 BSTART 레지스터 값들을 저장한다는 것을 나타내며, " 0" 은 이러한 값들이 차원화된 전송으로 페치되었음을 나타낸다. 지연된 패킷 전송이 가이드된 소스 또는 데스티네이션 어드레싱을 포함한다는 것을 패킷 전송 옵션 필드가 나타낸다면, 이러한 비트들만이 의미를 갖는다는 것을 명심하여야 한다. 그렇지 않다면, 데이터가 차원화된 전송 형태일 것으로 가정된다.

E 비트(상위 데이터 워드의 비트 29)는 지연된 패킷 전송이 오프칩 대 오프칩 패킷 전송인지를 나타낸다. 그와 같은 오프칩 대 오프칩 패킷 전송은 요청 프로세서의 파라메타 메모리 안의 버퍼를 사용한다. 이러한 E 비트가 " 1" 이라면, 지연된 패킷 전송은 오프칩 대 오프칩 패킷 전송이다. 이러한 E 비트가 " 0" 이라면, 지연된 패킷 전송은 3개의 다른 형태 중의 어느 하나이며 요청 프로세서의 파라메타 메모리는 어떠한 외부-대-외부 버퍼된 데이터도 저장하지 않는다.

S 비트(상단 데이터 워드의 비트 28)는 파라메타 메모리 외부-대-외부 버퍼가 소스 머신(320) 또는 데스티네이션 머신(340)에 의하여 액세스 되고 있는가를 나타낸다. 이러한 S 비트는 단지 E 비트가 요청 프로세서의 파라메타 메모리의 외부-대-외부 버퍼된 데이터의 저장을 나타내는 " 1" 인 경우에만 의미가 있게 된다. S 비트가 " 1" 이면, 패킷 전송이 지연된 때 소스 머신(320)이 외부-대-외부 버퍼를 액세스하고 있는 것이다. S 비트가 " 0" 이면, 데스티네이션 제어기가 외부-대-외부 버퍼를 액세스하고 있게 된다.

상위 데이터 워드의 비트 20 - 16들은 BUFPTR 레지스터(414)의 값을 저장한다. BUFPTR 레지스터(414)는 파라메타 메모리 외부-대-외부 버퍼 내로 지시하는 포인터이다. 이러한 포인터를 저장하는 것은 파라메타 메모리 외부-대-외부 버퍼의 상태를 복구함으로써 외부-대-외부 패킷 전송을 다시 시작하게 한다. 물론, 지연된 패킷 전송이 외부-대-외부 패킷 전송이었다면 이러한 필드만이 의미를 갖게된다.

상위 데이터 워드의 비트 14 - 8들은 BUFCOUNT 레지스터(441)를 저장한다. BUFCOUNT 레지스터(441) 값은 S 비트의 상태에 근거하여 해석된다. S 비트가 파라메타 메모리 외부-대-외부 버퍼에 대한 소스 액세스를 지연하는 것을 나타내는 " 1" 이라면, BUFCOUNT 레지스터(441)는 버퍼에 남아 있는 바이트의 수를 나타낸다. S 비트가 파라메타 메모리 외부-대-외부 버퍼에 대한 지연된 데스티네이션 액세스를 나타내는 " 0" 이라면, BUFCOUNT 레지스터(441)는 버퍼에 저장된 바이트의 수를 나타낸다. 이러한 필드는 지연된 외부-대-외부 패킷 전송에 대해서만 의미를 갖는다.

상위 데이터 워드의 비트 3 - 0들은 FCOUNT들이다. 이것은 패킷 전송 FIFO 버퍼(311)에 현재 유지된 데이터의 바이트 수이다.

지연된 패킷 서비스 파라메타의 하위 데이터 워드는 2개의 FIFO 포인터들을 갖는다. 하위 데이터 워드의 비트 11 - 8은 FDPTR 데스티네이션 포인터를 저장한다. 하위 데이터 워드의 비트 3 - 0들은 FSPTR 소스 포인터를 저장한다. 이러한 포인터들은 지연된 장소에서 지연된 패킷 전송의 시작을 허용한다.

Hex "060" 으로 끝나는 어드레스에서 시작하는 4개의 데이터 워드들은 패킷 전송 FIFO 버퍼(311)의 내용을 저장한다. 패킷 전송 FIFO 버퍼(311)를 비우는 것을 요청하기보다는 이러한 데이터를 저장하는 것에 의하여, 높은 우선 순위 패킷 전송이 먼저 시작할 수 있다. Hex "070" 으로 끝나는 어드레스에서 시작하는 4개의 데이터 워드들이 장래에 사용되어야만 하는 더 큰 버퍼와 같은 큰 패킷 전송 FIFO 버퍼의 내용을 저장하는 것을 지원하기 위하여 보류된다.

전송 제어기(80)는 다수의 비-표준 패킷 전송 액세스 모드를 지원한다. 이러한 것들은 블럭-기입, 직렬 레지스터 전송, 투과 및 주변 장치 전송들을 포함한다. 이러한 비-표준 패킷 전송 액세스 모드들은 패킷 전송 파라메타들의 패킷 전송 옵션 필드를 통하여 가능해 진다.

패킷 전송 파라메타의 패킷 액세스 모드가 "010" 인 때에는, 블럭-기입 패킷 액세스 모드가 가능하다. 전송 제어기(80)는 3개의 다른 블럭-기입 메카니즘을 지원한다: "8x", "4x" 및 "시뮬레이티드(simulated)". 시스템 하드웨어는 블럭-기입이 시작하는 시간에 블럭-기입 메카니즘을 결정한다. 이는 어떠한 형태의 블럭-기입을 시스템이 사용하는지, 또는 어드레싱된 메모리가 그것을 지원하는지 또는 지원하지 않는지에 상관없이 소프트웨어가 블럭-기입을 사용할 수 있게 한다. 그러나, 각각의 블럭-기입 모드들은 단지 64 비트 버스 및 8 비트 값(1 내지 8 확장)에 대해서만 지원된다. 블럭-기입을 위한 데스티네이션 시작 어드레스는 오프칩 상태여야 하며 정렬된 64 비트여야 하는데, 이는 즉 6개의 최하위 어드레스 비트들이 "000000" 이어야 한다는 말이다.

전송 제어기(80)에 의하여 사용된 블럭-기입 모드는 외부 메모리에 의하여 멀티프로세서 집적 회로(100)의 BS[1:0] 상에 입력된 값에 의하여 선택된다. 양호한 실시예에서는 블럭-기입이 64 비트 데이터 버스만을 위하여 지원되며, 따라서 이러한 버스 크기 입력들이 블럭-기입 및 컬러 레지스터 로드 사이클 동안 블럭-기입 선택들로서 사용된다. 표 23은 BS[1:0]에 의하여 선택된 블럭-기입 모드들을 도시한다.

[표 23]

BS[1:0]	블럭-기입 모드
0 0	시뮬레이트된 경우
0 1	보류된 경우
1 0	4 x
1 1	8 x

다양한 VRAM 크기 및 구조 때문에, 다수의 다른 블럭-기입 방법들이 존재한다. C x L x R 의 표기법이 다양한 블럭-기입 방법들을 표현하기 위하여 사용된다. 여기에서, C는 컬러 레지스터당 컬럼 위치들의 수, L은 컬러 레지스터의 비트로의 길이 및 R이 컬러 레지스터의 수이다. 예를 들어, 대부분의 1 M비트 VRAM들은 256 K비트 x 4 비트 데이터 어레이를 가지며 4 x 4 x 1 블럭-기입을 지원한다. 그들은 하나의 4 비트 컬러 레지스터를 가지며 각각의 블럭-기입 사이클은 각각의 4 메모리 어레이 평면 상의 4 인접 컬럼 위치로 컬러 레지스터를 기입하는 것을 제어한다.

8x 블럭-기입 모드는 액세스 마다 8개의 컬럼 위치(즉, 위치당 8개 비트들로 8x8x1 또는 8x8x2)를 기입할 수 있는 VRAM에 사용되도록 의도되었다. 소스 데이터의 각각의 비트들은 데이터 버스 D[63:0] 상의 한 비트로의 출력이며, 사이클로 액세스된 64 컬럼들 중 어느 하나에 8 비트 VRAM 컬러 레지스터의 내용을 기록하는 것을 가능하게 하거나 가능하지 못하게 할 수 있다. 각각의 컬럼들은 각각의 VRAM 안의 8개의 메모리 어레이 평면 상의 각각에 있는 컬럼 위치들을 나타낸다. 따라서 컬러 레지스터 데이터의 64 바이트까지 단 한번의 액세스 내에 기입될 수 있다.

8x 블럭-기입을 지원하는 비디오 RAM들은 블럭-기입 사이클 동안 3개의 최하위 컬럼 어드레스들을 무시한다. 따라서, 데이터 버스가 64 비트(8 바이트) 크기이기 때문에 블럭 기입 사이클들은 언제나 64 바이트 경계들에서 시작된다.

정렬된 64-바이트가 아닌 데스티네이션 어드레스인 경우, 전송 제어기(80)는 소스 비트들을 정렬시키며 기입되고 있지 않은 64 바이트 액세스 내의 위치에 대한 " 잃어버린 0들" 을 제공한다. 다음으로 전송 제어기(80)는 데이터 버스 상의 각각의 VRAM들 안에 적절한 컬럼 위치들을 어드레싱 할 수 있도록 비트들을 다시 맵핑한다.

제34도는 리틀 엔디언 모드에서의 8x 블럭-기입을 위한 데이터 재맵핑 프로세스이다. 처음의 8개의 소스 비트들은 데스티네이션의 처음 8개 바이트들을 제어한다. 이러한 바이트들은 액세스되고 있는 8-평면 VRAM 어레이들의 각각에 있는 실제적으로 최하위 컬럼 위치들이다. 최하위 바이트는 블럭-기입 동안 VRAM의 D0 입력들(16 비트 VRAM들을 위한 D0 입력들 및 D8 입력들)에 의하여 제어되기 때문에, 소스 비트 0 - 7들은 각각 데이터 버스의 0, 8, 16, 24, 32, 40, 48 및 56 비트들로 맵핑되며, 이는 액세스된 VRAM들의 DO 입력(16 비트 장치인 경우는 D0 및 D8 입력들)을 나타낸다. 소스 비트들의 나머지들은 유사한 방법으로 맵핑된다. 전송 제어기(80)는 빅 엔디언 8x 블럭-기입을 위하여 유사한 맵핑을 한다. 빅 엔디언 모드에서는, 소스 영상의 최하위 비트가 최좌측의 비트이며 최상위 비트는 최우측의 비트라는 것을 주목하여야 한다. 맵핑 메카니즘이 리틀 엔디언의 경우 사용된 것과 같기 때문에, 소스 비트 0 - 7들이 각각 비트 63, 55, 47, 39, 31, 23, 15 및 7로 맵핑된다. 이러한 비트들은 VRAM에서 액세스되고 있는 최하위 컬럼 위치를 나타낸다. 이러한 비트들이 VRAM의 D0 입력(16 비트 장치인 경우는 D0 및 D8 입력들)에 의하여 제어되기 때문에, 데이터 버스는 블럭-기입을 올바르게 동작하기 위하여서는 반전된 순서로 VRAM들에 접속되어야 한다.

반전된 순서로 데이터 버스를 접속하는 것은 데이터가 반전된 순서에 따라 기입 및 판독 모두가 되어지기 때문에 정상 판독 및 기입에는 영향이 없다. 사용자는 VRAM의 직렬 포트에서부터 또는 직렬 포트로 시프트된 비트들 역시 반전된 순서에 따른다는 것을 인식해야 한다. 이러한 경우 적절한 동작을 위하여는 출력 또는 입력 장치에 접속된 때 직렬 데이터 버스 순서가 반전될 필요가 있다.

4x 블럭-기입 모드는 액세스마다 4개의 컬럼 위치, 즉, 4×4×1, 4×4×4, 4×8×1, 4×8×2를 위치마다 4 또는 8 비트 중의 하나로 기입 할 수 있는 VRAM들에 사용하기 위하여 설계된다. 4×4 블럭-기입의 경우, 소스 데이터의 각각의 비트들은 데이터 버스 D[63:0] 상의 2개의 비트로 출력되며 사이클에 의하여 액세스된 64개의 컬럼 중 2개로 4 비트 VRAM 컬러 레지스터의 2개를 기입하는 것을 가능하게 또는 가능하지 않게 만든다. 각각의 컬럼들은 어레이의 4개 평면 상의 컬럼 위치를 나타내다. 각각의 컬러 레지스터들이 단지 4 비트 크기이기 때문에, 8 비트 픽셀을 나타내기 위하여는 2개의 레지스터들이 필요하다. 따라서 각각의 소스 비트들은 8 비트 기입이 생길 수 있도록 인접한 니블(nibble)에 기입하는 2개의 컬러 레지스터들을 제어하여야 한다. 이는 단 한번의 액세스로 컬러 레지스터 데이터의 32 바이트까지 기입될 수 있게 한다.

4x 블럭-기입을 지원하는 비디오 RAM들은 블럭-기입 사이클 동안 2개의 최하위 컬럼 어드레스들을 무시한다. 따라서, 블럭-기입들은 언제나 32-바이트 경계상에서 시작된다. 8x 블럭-기입에서와 같이, 전송 제어기(80)는 32-바이트 블럭안의 데스티네이션 시작 어드레스에 의하여 규정된 더블-워드에 소스 데이터를 정렬시키며 기입되어 있지 않은 더블-워드에 대한 " 잃어버린 0들" 을 채운다.

제35도는 리틀 엔디언 모드에서의 4× 블럭-기입을 위한 데이터 재맵핑 프로세스를 도시한다. 처음의 8개의 소스 비트들은 데스티네이션의 처음 8개의 바이트들을 제어한다. 그러나, 4×4 블럭-기입에서는 각각의 데이터 입력들이 단지 4개의 비트들이 기입되도록 하기 때문에, 각각의 소스 비트들은 전체 바이트가 기입되도록 하기 위하여 2개의 데이터 버스 핀들에 재맵핑되어야만 한다. 따라서 소스 비트 0 - 7들은 각각 비트 0 및 4, 8 및 12, 16 및 20, 24 및 28, 32 및 36, 40 및 44, 48 및 52, 및 56 및 60들로 맵핑된다.

4x8x1 또는 4x8x2 블럭-기입의 경우에는, VRAM 컬러 레지스터들이 8 비트 크기이다. 그러나, 단지 VRAM의 8개의

데이터 입력들 중 4개들만이 기입될 바이트들을 선택하기 위하여 사용된다. 따라서 데이터 버스의 반이 사용되지 않으며 단지 32 바이트만이 단 한번의 액세스에서 기입될 수 있다. 그러나, D[63:0]에 실제적으로 출력된 데이터가 동일하다는 것을 주목하여야 한다. VRAM들은 단순히 모든 다른 니블들을 무시한다. 빅 엔디언 모드에서는, 소스 비트 0 - 7들이 각각 외부 데이터 버스의 비트들 63 및 59, 55 및 51, 47 및 43, 39 및 35, 31 및 27, 23 및 19, 15 및 11, 및 7 및 3으로 맵핑된다. 8× 빅 엔디언 블럭-기입에서와 같이, 데이터 버스는 올바른 동작을 위하여서는 반전된 순서로 VRAM들에 접속되어야만 한다.

블럭-기입을 지원하지 않는 메모리 장치인 경우는, 전송 제어기(80)가 시뮬레이팅된 블럭-기입을 제공한다. 이러한 모드에서는, 패킷 전송 파라메타에 들어있는 64 비트 컬러 레지스터 값이 데이터 버스 상에 출력되며 각각의 소스 데이터 비트는 사이클 동안 어드레싱된 8개의 바이트들의 하나를 사용 가능하게 또는 가능하지 않게 하기 위하여 $\overline{CAS}$컬럼 어드레스 스트로브 핀들을 제어하여 바이트 선택으로서 기능한다. 따라서, 블럭-기입들은 컬러 레지스터 값들이 필 밸류가 되는 경우 필-위드-밸류 형태 전송으로 본질적으로 변형된다. 따라서 데스티네이션 액세스는 특정 바이트들에 대한 기입들이 소스 데이터에 의하여 규정된 바와 같이 사용 가능하지 않게 되는 정상, 페이지-모드 64 비트 기입 사이클이 된다.

4x 및 8x 블럭-기입 사이클들이 수행될 수 있기 전에, VRAM의 컬러 레지스터는 올바른 값으로 로드되어야만 한다. 전송 제어기(80)는 이러한 동작을 패킷 전송 파라메타에 들어있는 컬러 레지스터 값을 사용하는 로드 컬러 레지스터 LCR 사이클을 수행함으로써 하게 된다. 블럭-기입 패킷 전송이 호스트 액세스 또는 VRAM 컬러 레지스터들을 변경시킬 수 있는 다른 블럭-기입 패킷 전송과 같은 높은 우선 순위 요청에 의하여 인터럽트될 수 있기 때문에, 로드 컬러 레지스터 사이클은 역시 블럭-기입 패킷 전송이 다시 시작하는 때마다 수행되어야만 한다. 따라서, 로드 컬러 레지스터 사이클은, 4x 또는 8x 블럭-기입 패킷 전송이 지연된 다음 다시 시작할 때; 또는 호스트가 영상 시스템 버스를 사용한 다음 복귀한 후에 블럭-기입 패킷 전송이 계속할 때마다 발생한다. 로드 컬러 레지스터 사이클은 액세스되고 있는 메모리가 시뮬레이티드 블럭-기입을 요청한다면 수행되지 않을 것이다.

일단 컬러 래치들이 로드되었다면, 다른 로드 컬러 레지스터 사이클이 위의 상태 중 어느 하나가 생기지 않는 한 수행되지 않을 것이다. 예를 들어, 블럭-기입이 8x 모드로 시작하고, 그 후 시뮬레이티드와 8× 모드사이에서 교대한다면, 로드 컬러 레지스터가 8× 모드로 진입하는 각각의 시간에서는 반복되지 않을 것이다.

블럭-기입 패킷 전송을 위한 프로세스의 시퀀스는 다음과 같다. 전송 제어기(80)는 로드 컬러 레지스터 상태 코드를 출력하며 수행될 제1 블럭-기입을 어드레싱한다. 다음으로 전송 제어기(80)는 BS[1:0] 상에 입력된 값을 판독한다. BS[1:0]이 " 10" 또는 " 11" 과 같다면, 로드 컬러 레지스터 사이클은 패킷 전송 파라메타에 들어 있는 64 비트 컬러 레지스터 값을 사용하는 것을 완료한다. 그후 4× 또는 8× 블럭-기입 사이클들이 패킷 전송을 종료시키기 위하여 발생된다. 다른 한편으로, BS[1:0]이 " 00" 과 같다면, 로드 컬러 레지스터 사이클은 패킷 전송 파라메타에 들어 있는 64 비트 컬러 레지스터 값을 데이터로 소스 데이터 비트들을 바이트 선택들로서 사용하는 정상 페이지-모드 기입이 된다. 새로운 로우 액세스가 페이지 변화 또는 높은 우선 순위 사이클로부터의 인터럽션 때문에 시작되었다면, 로드 컬러 레지스터 상태 코드와 수행될 다음 블럭-기입의 어드레스를 출력하는 단계가 다음 데스티네이션 어드레스를 위하여 반복된다.

일단 로드 컬러 레지스터 사이클이 수행되었다면, 시퀀스는 다음과 같이 계속된다. 전송 제어기(80)는 블럭-기입 상태 코드와 수행될 다음 블럭-기입의 어드레스를 출력한다. BS[1:0] = " 10" 또는 " 11" 이라면, 블럭-기입 페이지-모드 사이클은 소스 데이터 비트를 사용하는 것을 완료한다. 새로운 로우 액세스가 시작되었다면, 블럭-기입 상태 코드와 수행될 다음 블럭-기입의 어드레스를 출력하는 단계가 반복된다. 한편으로는, BS[1:0]이 " 00" 과 같다면, 블럭-기입

사이클은 패킷 전송에 들어있는 64 비트 컬러 레지스터 값을 데이터로, 또 소스 데이터 비트들을 바이트 선택으로서 사용하는 정상 페이지-모드 기입이 된다. 다시, 새로운 로우 액세스가 시작되었다면, 블럭-기입 상태 코드와 수행될 다음 블럭-기입의 어드레스를 출력하는 단계가 다음 데스티네이션 어드레스를 위하여 반복된다.

전송 제어기(80)는 언제나 " 실제" 4× 또는 8× 블럭-기입을 수행하려고 시도할 것이다. 따라서 시뮬레이티드 블럭-기입 모드 동안 생기는 정상 기입 사이클들은 항상 로드 컬러 레지스터 또는 블럭-기입 상태 코드 중의 어느 하나를 가질 것이다.

패킷 전송 옵션 필드의 패킷 액세스 모드가 직렬 레지스터 전송 모드를 선택하는 " 011" 이라면, 전송 제어기(80)의 데이터 전송 모드가 사용 불가능해 진다. 전송 제어기(80)는 어드레스들과 $\overline{CAS}$들을 출력하기만 할뿐이다. 그것은 역시 " 판독 전송" 또는 " 기입 전송" VRAM 모드 중 하나를 선택하기 위하여 로우 시간에 TRG,W 및 DSF[1:0] 핀들을 구동한다. 이러한 동작들 중의 하나를 수행할 때에, D[63:0] 핀들은 VRAM 전송 마스크들을 사용 불가능하게 하기 위하여 Hex " FFFFFFFFFFFFFFFF" 의 마스크 값을 출력한다.

소스 파라메타들이 판독 전송을 수행하기 위한 어드레스들을 발생시키는 데에 사용되는데, 이러한 것들은 레지스터 전송에 대한 메모리이며, 데스티네이션 파라메타들이 기입 전송을 수행하기 위하여 사용되는데, 이러한 것들은 메모리 전송들에 대한 레지스터들이다. 소스 및 데스티네이션 둘다에 의하여 수행되는 각각의 액세스들은 단 한번의 로우 액세스들이다. 전송 제어기(80)를 통하여 어떠한 데이터도 전송되지 않기 때문에, 모든 소스 액세스들이 데스티네이션 액세스 전에 수행될 것이다. 각각의 소스 액세스들은 VRAM 로우가 VRAM 시프트 레지스터 안으로 전송되도록 야기한다. 실제로, 소스 파라메타들은 보통 정렬된. 어드레스들, 1인 A 카운트, 0인 B 카운트 및 0인 C 카운트들로 셋업된다. 따라서, 단지 한번의 전송이 수행된다. 모든 $\overline{CAS}$라인들이 판독 프로세스를 위하여 활성 상태이기 때문에, 소스 버스 크기와 같게 또는 그보다 작게 A 카운트를 설정시키는 것은 단 한번의 전송으로 이루어진다. 따라서, 1인 소스 A 카운트가 대부분의 상황에 대해 이상적이다.

각각의 데스티네이션 액세스들은 VRAM 시프트 레지스터가 VRAM 메모리 어레이 로우 내로 전송되도록 만든다. 정상적으로는, 각각의 데스티네이션 액세스는 다음 시퀀셜 로우 어드레스로의 정렬된 전송이다. 이는 데스티네이션 파라메타들을 정렬된 시작 어드레스들, 1인 A 카운트, (로우의 수) - 1인 B 카운트 및 VRAM 로우 어드레스 피치와 같은 B 피치들로 셋업시킴으로써 이룰 수 있다. 다시, 모든 $\overline{CAS}$라인들이 A 카운트를 데스티네이션 버스 크기와 같게 또는 그보다 작게 설정하는 것이 라인마다 단 한번의 전송으로 이루어질 수 있도록 기입 전송 동안 활성화된다. 따라서 1인 데스티네이션 A 카운트는 대부분의 상황에 대해 이상적이다.

시프트 레지스터 전송 패킷 전송이 VRAM 시프트 레지스터의 내용들을 변경시킬 지도 모르는 높은 우선 순위에 의하여 인터럽트될 수 있기 때문에, VRAM 시프트 레지스터가 내용을 잃어버렸을 때(corrupted)마다 판독 전송 사이클을 수행할 필요가 있게 된다. 따라서, 시프트 레지스터 전송 패킷 전송 소스 동작, 판독 전송들이, 시프트 레지스터 전송 패킷 전송이 시작한 때; 시프트 레지스터 전송 패킷 전송이 지연된 다음 다시 시작한 때; 및 시프트 레지스터 전송 패킷 전송이 호스트가 영상 버스를 사용한 후 복귀한 다음 계속된 때마다 수행될 것이다.

직렬 레지스터 전송 패킷 전송의 간단한 예가 제36도에 도시되었다. 로우 0은 원하는 패턴으로 설정되었다고 가정된다. 한편 패킷 전송은 이러한 패턴을 스트립된 효과(striped effect)를 발생하기 위하여 로우 2, 4, 6, 8 및 10으로 복사한다.

투과 모드들은 패킷 전송 옵션 필드의 패킷 전송 액세스 모드 비트들을 " 1XX" 로 설정시킴으로써 사용 가능해진다. 투과 옵션들 중의 하나를 규정하는 것은 " 소스 상의 투과" 동작을 가능케 만든다. 소스 데이터는 패킷 전송 파라메타에 규정된 64 비트 투과 값에 비교된다. 투과는 8, 16, 32, 또는 64 비트 데이터 크기로 규정될 수 있다. 따라서, 하나의 64 비트, 2개의 32 비트, 4개의 16 비트, 또는 8개의 8 비트 비교가 행해진다. 비교 중 어떤 것이라도 참이라면, 투과 및 바이트 기입 회로(353)(제38도에 도시되었음)는 데스티네이션 바이트들이 기입되는 것을 막기 위하여 해당 바이트-스트로브들을 사용 불가능하게 만든다. 투과는 4개의 오프칩 데스티네이션만을 위하여 지원된다. 온칩 데스티네이션을 위하여 투과를 규정하는 것은 패킷 전송이 오류 상태로 지연되도록 만든다.

투과 감지는 소스 데이터가 데스티네이션 및 외부 버스 크기에 정렬된 후 투과 및 바이트 기입 회로(353)(제38도에 도시됨)에 의하여 인가된다. 한편 모든 8 데이터의 바이트들은 버스 크기가 64 비트보다 작은 경우일 지라도 해당 8 투과 값의 바이트들과 비교된다. 8개의 비교들은 투과 데이터 크기에 따라서 그룹지어 진다. 그룹 안의 모든 비교된 바이트들이 매칭된다면, 그룹과 관련된 바이트 스트로브$\overline{CAS}$신호들은 사용 불가능해지며, 상기 그룹 내의 어떠한 바이트들도 기입되는 것이 금해진다. 제37a, 37b, 37c 및 37d도들은 각각 64 비트, 32 비트, 16 비트 및 8 비트 투과 데이터 크기에 대해 행해진 비교들을 도시한다. " & " 부호는 그룹을 형성하기 위하여 함께 엔딩(ANDing)된 바이트 비교들을 도시한다. 제37a도는 64 비트 투과 크기가 행해질 하나의 64 비트 비교를 도시한다. 소스 데이터 및 투과 값들이 같다면, 모든$\overline{CAS}$스트로브들은 사용 불가능해 진다. 그렇지 않다면, 모든 8 바이트들이 기입될 것이다. 제37b도는 32 비트 투과 크기에서 각각$\overline{CAS}$[7:4] 및$\overline{CAS}$[3:0]에 의하여 제어된 2개의 32 비트 비교를 도시한다.

제37c도는 16 비트 투과 크기에서 각각$\overline{CAS}$[7:6],$\overline{CAS}$[5:4],$\overline{CAS}$[3:2] 및$\overline{CAS}$[1:0]에 의하여 제어된 4개의 16 비트 비를 도시한다. 제37d도는 8 비트 투과 크기에서 각각 분리되어$\overline{CAS}$[7],$\overline{CAS}$[6],$\overline{CAS}$[5],$\overline{CAS}$[4],$\overline{CAS}$[4],$\overline{CAS}$[3],$\overline{CAS}$[2],$\overline{CAS}$[3:2] 및$\overline{CAS}$[1:0]에 의하여 제어된 8개의 8비트 비교들을 도시한다.$\overline{CAS}$[7:0] 스트로브들은 항상 엔디언 동작에 무관하게 데이터 버스 상의 같은 비트들과 동일하다는 점을 주목해야 한다. 따라서 제37도는 빅 및 리틀 엔디언 포맷 모두에 적용된다. 투과 비교는 외부 버스에 대한 정렬 후에 발생한다. 따라서 외부 버스 크기가 32 비트라면, 데이터는 64 비트 투과 모드가 선택된 경우일 지라도 언제나 투과 값의 비트 31 - 0(빅-엔디언인 경우는 64 - 32)들과 비교된다. 투과 메카니즘은 비교 그룹의 정수(integral number)로 나누어질 수 있는 현재 버스 크기와 함께 운영될 수 있도록 설계된다. 따라서 버스 크기는 언제나 투과 크기와 같거나 또는 그보다 커야만 한다.

제38도는 투과 및 바이트 기입 논리(353)의 구조를 도시하며, 이는 데스티네이션 멀티플렉서 및 정렬 논리(350)의 일부분이다. 투과 레지스터(601)는 패킷 전송 파라메타로부터의 64 비트 투과 값을 저장한다. 일련의 8 비트 비교기(611, 612, 613, 614, 615, 616, 617 및 618)들은 투과 레지스터(601)에 저장된 투과 값의 개개의 비트들을 데스티네이션 멀티플렉서(351)로부터의 정렬된 데스티네이션 데이터의 해당 비트들과 비교한다. 각각의 비교기(611, 612, 613, 614, 615, 616, 617, 및 618)들은 해당 비트들이 같은가를 나타내는 바이트 동일 신호를 발생시킨다. 해당 1/2 워드로부터의 1/2 워드 AND 게이트들의 집합(621, 622, 623 및 624)은 각각 바이트 0 및 1, 바이트 2 및 3, 바이트 4 및 5 및 바이트 6 및 7을 위한 신호들과 동일하다. 2개의 워드 AND 게이트(631 및 632)들은 각각 1/2 워드 게이트(621, 622, 623 및 624)들과 바이트 0 내지 3, 및 바이트 4 내지 7을 위한 해당 워드 동일 신호들로부터 접속된다. 마지막으로, 더블 워드 AND 게이트(641)가 워드 AND 게이트(631 및 632)와 접속되며 모든 64 비트 더블 워드 동일 신호를 형성한다.

멀티플렉서(645)는 8개의 바이트 동일 신호들, 4개의 1/2 워드 동일 신호들, 2개의 워드 동일 신호들 및 더블 워드 동일 신호들을 수신한다. 멀티플렉서(645)는 또한 패킷 전송 옵션 필드의 패킷 전송 액세스 모드 필드의 비트 17 - 16들에 따라서 선택된 투과 크기 표시를 받아들인다. 이러한 비트들은 표(19)에 도시된 바와 같이 엔코딩된다. 멀티플렉서(645)는 각각의 입력들과 투과 크기에 근거하여 바이트 기입 스트로브 신호들, 실제적으로는 $\overline{CAS}$ 신호들을 출력한다. 8 비트의 투과 크기가 선택되었다면, 바이트 동일 신호들은 각각의 바이트 기입 스트로브들을 제어한다. 투과 컬러 값과 해당 데이터 바이트의 동일 표시는 바이트 기입 스트로브내에 있게 된다. 따라서 데이터는 메모리 안으로 기입되지 않는다. 16 비트의 투과 크기를 선택한 경우, 각각의 1/2 워드 동일 신호는 2개의 해당 바이트 기입 스트로브를 제어하며, 따라서 각각의 1/2 워드 동일 신호들은 2개의 바이트 기입 스트로브들을 제어한다. 32 비트의 투과 크기를 선택한 경우는, 각각의 워드 동일 신호들이 4개의 해당 바이트 기입 스트로브들을 제어한다. 64 비트 투과 크기를 선택한 경우는, 모든 8개의 바이트 기입 스트로브들이 더블 워드 동일 신호에 의하여 제어된다. 따라서 선택된 투과 크기에 근거하여, 투과 컬러 값을 갖는 투과 크기와 같은 데이터 부분의 동일성은 데스티네이션 안으로의 상기 데이터 기입을 중단시킨다. 이러한 기능은 많은 그래픽 응용에 있어서 매우 유용하다.

주변 장치 전송들은 패킷 전송 옵션 필드의 패킷 전송 액세스 모드가 " 001" 과 같은 경우 주변 장치가 선택된 때에 수행된다. 이러한 모드는 주변 장치가 멀티프로세서 집적 회로(100)의 외부 메모리에 기입 또는 메모리로부터 판독하기 위하여 전송 제어기(80)의 메모리 제어기를 사용할 수 있게 허용하도록 설계된다. 주변 장치 전송이 생길 때에는, 전송 제어기(80)가 정상적으로 메모리 어드레스 및 제어 라인들을 구동시키나, 데이터를 기입하거나 판독하지는 않는다. 이는 영상 시스템 버스에 접속된 호스트 시스템(1)과 같은 주변 장치가 데이터를 기입하거나 판독할 수 있게 한다. 주변 전송의 판독 또는 기입은 패킷 전송 파라메타들이 프로그래밍되는 방법과 같은 방법에 의하여 수행된다.

메모리로부터 주변 장치로의 주변 판독 전송은 주변 장치에 의하여 필요한 메모리 데이터를 액세스하기 위하여 패킷 전송의 소스 파라메타들을 프로그래밍하여 발생된다. 소스 액세스 모드는 차원화 또는 가이드되나 필 - 위드 - 밸류일 수는 없다. 데스티네이션 전송은 " 000" 으로 데스티네이션 전송 모드를 설정시키고 데스티네이션 A 카운트를 0으로 설정시킴에 의하여 사용 불가능해져야만 한다.

메모리에서 주변 장치로의 기입인 주변 기입 전송은, 주변 장치가 기입을 필요로 하는 메모리 영역을 액세스하도록 데스티네이션을 프로그래밍하여 발생된다. 데스티네이션 액세스 모드는 차원화 또는 차원화될 수 있다. 소스 전송은 소스 전송 모드를 " 000" 으로 소스 A 카운트를 0으로 설정시킴에 의하여 사용 불가능해 져야만 한다. 패킷 전송 오류는 바이트의 데스티네이션 수가 바이트의 소스의 수를 초과할 때일지라도 이러한 경우에는 발생하지 않는다.

주변 장치 패킷 전송 요청은 어떠한 프로세서에 의해서라도 어떠한 우선 순위로도 제출될 수 있으며 정상 우선 순위 방법을 사용하여 제공될 수 있을 것이다. 그러나 정상의 동작은 데이터의 판독 또는 기입을 원하는 때에, 주변 장치로 전송을 개시시킬 것이다. 이는 외부 인터럽트들을 사용하여 이룰 수 있다. 주변 장치가 서비스를 요청할 때에는, 외부 인터럽트 입력인 멀티프로세서 집적 회로(100) 중의 하나를 통하여 마스터 프로세서(60)를 인터럽트시킬 수 있다. 한편 마스터 프로세서(60)는 인터럽트 서비스 루틴의 부분으로서 주변 장치 패킷 전송을 제출할 수 있다. 서비스 루틴 내에서 인터럽트들이 사용 불가능하다면, 전송은 " 긴급" 우선 순위를 발행함으로써 디지탈 연상/그래픽 프로세서(71, 72, 73 및 74) 캐시 요청들보다 우선하는 우선 순위가 될 수 있다.

일단 주변 장치가 마스터 프로세서(60)를 인터럽트했다면, 마스터 프로세서는 그것이 데이터를 판독 또는 기입할 수 있기 전에 패킷 전송을 시작할 때까지 기다려야만 한다. 주변 장치 전송의 시작은 로우 시간에서 STATUS[4:0] 상의 특별 사이클 형식 코드에 의하여 신호를 받는다. 값 " 00100" 이 판독을 위하여 사용되며, 값 " 00101" 이 기입을 위하여 사용된다. 주변 장치 전송이 시작과 끝은 항상 로우 액세스를 강제하는 LASTPAGE 레지스터(360)를 무효로 만든다. 주변 장치는 데이터 전송을 시작한 때를 결정하기 위하여 STATUS[4:0] 핀들을 모니터링하여야 한다. 메모리 사

이클 버스의 크기, 컬럼 타이밍, 기타 전송 제어기(80)에 의하여 발생된 것들의 포맷은 버스 크기 선택 BS[1:0], 컬럼 타이밍 선택 CT[1:0], 기타 사이클의 시작시의 메모리 동일 입력들로 어떠한 것들이 선택되더라도 일치할 것이다.

따라서 주변 장치는 자신의 데이터 전송을$\overline{CAS}$[7:0], CLKOUT 등을 사용하는 이하 순차적인 컬럼 액세스들에 동기화시킬 수 있다.

메모리 어드레싱이 패킷 전송 파라메타들에 의하여 발생되기 때문에, 전송하는 주변 장치는 전송 제어기(80)에 의하여 액세스된 순서에 따라서 데이터를 보내거나 받아들일 준비가 되어 있어야만 한다. 주변 장치는 역시 전송 제어기(80)의 전송 속도에 적합할 수 있거나 전송을 늦추기 위한 대기 상태를 삽입시킬 수 있어야 한다.

주변 장치가 전송을 위하여 영상 시스템을 사용하기 때문에, 그것은 주변 전송이 시작할 때까지 버스를 구동시키면 안된다. 이는 주변 장치와 멀티프로세서 집적 회로(100) 데이터 버스들 사이에 송수신기를 두고 주변 전송 동안에만 그들을 사용 가능하게 함으로써 이룰 수 있다. 멀티프로세서 집적 회로(100)는 자신의 데이터 버스를 전송 동안 고임피던스 상태로 두며 역시 자신의 외부 수신기가 있다면 그를 사용 불가능하게 만들기 위하여 DBEN을 비활성 하이 상태로 구동시킬 것이다.

전송 제어기(80)는 빅 엔디언 또는 리틀 엔디언 포맷 중 어느 하나의 포맷으로 데이터를 액세스할 수 있다. 엔디언 모드는 바이트들이 액세스되는 방법을 선택한다. 리틀 엔디언 모드에서, 바이트 0은 워드 내의 최우측의 바이트이며 연속된 바이트들은 좌측 방향으로 번호가 부여된다. 빅 엔디언에서는, 바이트 0이 워드 내의 최좌측의 바이트이며 연속된 바이트들은 우측 방향으로 번호가 부여된다.

어드레스의 3개의 최하위 비트들 및 전송될 바이트의 수들은 유효 데이터 바이트들의 위치들을 결정한다. 표 24a와 24b는 리틀 엔디언 모드에서의 64 비트 버스 전송들을 위한 바이트 위치들을 도시한다. 표 24a와 24b에서, V들은 유효 바이트 위치들을 나타내며 0들은 비유효 바이트들을 나타낸다. 대쉬(dash)들은 동작이 수행될 수 없음을 나타낸다.

[표 24a]

3개의 최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트	3 바이트	4 바이트
0 0 0	0000000V	000000VV	00000VVV	0000VVVV
0 0 1	000000V0	00000VV0	0000VVV0	000VVVV0
0 1 0	00000V00	0000VV00	000VVV00	00VVVV00
0 1 1	0000V000	000VV000	00VVV000	0VVVV000
1 0 0	000V0000	00VV0000	0VVV0000	VVVV0000
1 0 1	00V00000	0VV00000	VVV00000	--
1 1 0	0V000000	VV000000	--	--
1 1 1	V0000000	--	--	--

[표 24b]

3개의 최하위 어드레스 비트	바이트 번호 5 바이트	6 바이트	7 바이트	8 바이트
0 0 0	000VVVVV	00VVVVVV	0VVVVVVV	VVVVVVVV
0 0 1	00VVVVV0	0VVVVVV0	VVVVVVV0	--
0 1 0	0VVVVV00	VVVVVV00	--	
0 1 1	VVVVV000	--	--	
1 0 0	--	--	--	
1 0 1	--	--	--	
1 1 0	--	--	--	
1 1 1	--	--	--	

표 25a와 25b는 빅 엔디언 모드에서의 64 비트 버스 전송들을 위한 바이트 위치들을 도시한다.

[표 25a]

3개의 최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트	3 바이트	4 바이트
0 0 0	V0000000	VV000000	VVV00000	VVVV0000
0 0 1	0V000000	0VV00000	0VVV0000	0VVVV000
0 1 0	00V00000	00VV0000	00VVV000	00VVVV00
0 1 1	000V0000	000VV000	000VVV00	000VVVV0
1 0 0	0000V000	0000VV00	0000VVV0	0000VVVV
1 0 1	00000V00	00000VV0	00000VVV	--
1 1 0	000000V0	000000VV	--	--
1 1 1	0000000V	--	--	--

[표 25b]

3개의 최하위 어드레스 비트	바이트 번호 5 바이트	6 바이트	7 바이트	8 바이트
0 0 0	VVVVV000	VVVVVV00	VVVVVVV0	VVVVVVVV
0 0 1	0VVVVV00	0VVVVVV0	0VVVVVVV	--
0 1 0	00VVVVV0	00VVVVVV	--	--
0 1 1	000VVVVV	--	--	--
1 0 0	--	--	--	--
1 0 1	--	--	--	--
1 1 0	--	--	--	--
1 1 1	--	--	--	--

외부 버스가 32 비트로 제한된 경우는, 버스의 단지 4개 최하위 바이트들이 데이터를 전송하기 위하여 사용된다. 이는 D[31:0]들이 리틀 엔디언 전송에서 사용되며 D[63:32]들이 빅 엔디언 전송들을 위하여 사용된다. 리틀 엔디언을 위한 어드레스의 2개의 최하위 비트들에 근거된 바이트 위치들이 표 26에 나타나 있다. X들은 무시된 64 비트 버스의 바이트들을 나타낸다. 대쉬들은 다시 수행될 수 없는 전송들을 나타낸다.

[표 26]

2개의 최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트	3 바이트	4 바이트
0 0	XXXX000V	XXXX00VV	XXXX0VVV	XXXXVVVV
0 1	XXXX00V0	XXXX0VV0	XXXXVVV0	--
1 0	XXXX0V00	XXXXVV00	--	--
1 1	XXXXV000	--	--	--

빅 엔디언을 위한 어드레스의 2개의 최하위 비트들에 근거된 바이트 위치들은 표 27에 도시된다.

[표 27]

2개의 최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트	3 바이트	4 바이트
0 0	000VXXXX	00VVXXXX	0VVVXXXX	VVVVXXXX
0 1	00V0XXXX	0VV0XXXX	VVV0XXXX	--
1 0	0V00XXXX	VV00XXXX	--	--
1 1	V000XXXX	--	--	--

외부 버스가 16 비트로 제한된 때에는, 단지 리틀 엔디언을 위한 2개의 최하위 바이트 D[15:0] 또는 빅 엔디언을 위한 D[63:48]들이 데이터를 전송시키기 위하여 사용된다. 표 28은 어드레스의 최하위 비트에 근거된 리틀 엔디언 모드를 위한 바이트 위치들을 도시한다.

[표 28]

최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트
0	XXXXXX0V	XXXXXXVV
1	XXXXXXV0	--

표 29는 어드레스의 최하위 비트에 근거된 리틀 엔디언 모드를 위한 바이트 위치들을 도시한다.

[표 29]

최하위 어드레스 비트	바이트 번호 1 바이트	2 바이트
0	V0XXXXXX	VVXXXXXX
1	0VXXXXXX	--

외부 버스가 8 비트들을 위하여 구성된 경우에는, 단지 리틀 엔디언에 대해서는 최하위 바이트 D[7:0] 또는 빅 엔디언 모드에 대해서는 D[63:56]이 데이터를 전송하기 위하여 사용된다. 따라서 이러한 버스 크기에 대하여는 바이트의 위치들이 고정된다.

내부 크로스바 버스 상으로의 32 비트 외부 데이터 전송의 경우, 전송 제어기(80)는 표 24a와 24b 및 표 25a와 25b로부터의 " 1 바이트" , " 2 바이트" , " 3 바이트" 및 " 4 바이트" 의 바이트 위치들을 사용하는 일반적인 64 비트 액세스들을 수행한다. 내부 크로스바 버스 상으로의 16 비트 외부 데이터를 전송하는 경우에는, 전송 제어기(80)는 표 24a와 24b 및 표 25a와 25b로부터의 " 1 바이트" 및 " 2 바이트" 의 바이트 위치들을 사용하는 일반적인 64 비트 액세스들을 수행한다.

멀티프로세서 집적 회로(100)를 위한 엔디언 모드는$\overline{UTME}$ 입력을 사용하는 재설정시 선택된다. 멀티프로세서 집적 회로(100)는 RESET 입력 상의 상승 에지 전의 클릭 사이클 상의 $\overline{UTME}$ 의 값을 샘플링하여 래치한다.$\overline{UTME}$ 이 재설정의 끝에서 로우(" 0")로 샘플링되었다면, 멀티프로세서 집적 회로(100)는 다음의 하드웨어 재설정이 생길 때까지 빅 엔디언 모드로 동작할 것이다.$\overline{UTME}$ 이 하이(" 1")로 샘플링되었다면, 멀티프로세서 집적 회로(100)는 리틀 엔디언 모드로 동작할 것이다.

위에서 검토된 바와 같이, 패킷 전송 파라메타들은 워드(32 비트) 레벨인 경우에만 엔디언에 독립적이다. 전송 제어기

(80)는 언제나 패킷 전송 파라메타들을 더블-워드(64 비트) 전송들로서 페치하여 저장한다. 전송 제어기(80)는 선택된 엔디언에 따라서 32 비트 워드들을 교환한다. A 카운트 및 B 카운트와 같은 워드 내의 어떠한 16 비트 크기들은 이러한 값들을 갖는 필드가 단 하나의 32 비트 크기로 여겨지기 때문에 엔디언에 따라 교환되지 않는다. 마찬가지로, 64 비트 투과 워드 및 64 비트 컬러 레지스터 값 필드는 언제나 단 하나의 64 비트 크기로 다루어지며 그것의 바이트들은 엔디언에 따라서 교환되지 않는다.

로컬 메모리 판독 및 기입 사이클들은 메모리와 멀티프로세서 집적 회로(100) 사이의 데이터 및 명령들을 전송하기 위하여 사용된다. 이러한 사이클들은 패킷 전송, 캐시 요청 또는 전송 제어기(80)로의 직접 외부 액세스 요청 등의 결과로서 생길 수 있다. 판독 사이클들은 메모리로부터 멀티프로세서 집적 회로(100)로 데이터를 전송시킨다. 전송 제어기(80)는 판독이 일어났음을 나타내기 위하여 사이클의 시작시 STATUS[4:0] 상에 " 00000" 을 출력한다. 사이클 동안, WE는 비활성 하이 상태로, TRG는 메모리 출력 구동기를 사용 가능하게 하기 위하여 $\overline{RAS}$의 하강 후에 로우 상태로 구동되며 DDIN은 데이터 송수신기가 멀티프로세서 집적 회로(100) 안으로 구동할 수 있도록 사이클 동안 액티브 로우 상태가 된다. 전송 제어기(80)는 D[63:0]을 그것이 메모리로부터 구동되며 적절한 컬럼 상태 동안 입력 데이터를 래치하는 것을 허용하기 위하여 하이 임피던스 상태로 스위칭한다. 전송 제어기(80)는 언제나 64 비트 더블-워드를 판독하며 그 다음으로 적절한 데이터 바이트들을 추출하여 정렬시킨다. 따라서 64 비트보다 작은 버스 크기에 대하여는 유효하지 않은 바이트들이 무시된다.

제39도는 본 발명의 멀티프로세서 집적 회로(100)의 고해상도 텔레비젼 시스템에서의 사용을 도시한다. 제39도는 고해상도 텔레비젼 신호의 3개의 소스들을 도시한다. 이러한 것들은 방송 텔레비젼 신호, 컴팩트 디스크 롬(read only memory) 신호 및 케이블 텔레비젼 신호들이다.

안테나(801)는 고해상도 텔레비젼 신호를 포함하는 방송 무선 주파수 신호들을 수신한다. 텔레비젼 튜너(802)는 특정의 무선 주파수 신호를 선택하는 튜닝된 수신기, 무선 주파수 신호 상에 엔코딩된 영상 데이터를 추출하는 복조기 및 아날로그 대 디지탈 변환기 등을 포함한다. 한편 텔레비젼 튜너(802)는 고해상도 텔레비젼 영상에 따르는 디지탈 신호를 발생시킨다. 이러한 디지탈 신호들은 역시 영상 시스템 버스에 접속된 버퍼(803) 내에 일시적으로 저장된다.

컴팩트 디스크 롬(CD ROM) 플레이어(811)는 컴팩트 디스크 상에 영구적으로 저장된 데이터들을 판독한다. 이러한 것들은 원하는 고해상도 텔레비젼 프로그램에 따르는 영상 데이터들을 포함한다. 컴팩트 디스크 롬 플레이어(811)는 일시적인 저장을 위하여 컴팩트 디스크로부터 버퍼(812)로 판독된 디지탈 데이터를 제공한다. 버퍼(812)는 역시 영상 시스템 버스에 접속된다.

케이블 시스템 박스(821)는 케이블 시스템에 양방향 접속된다. 이러한 양방향 접속은 케이블 시스템으로부터 사용자에게 고해상도 텔레비젼 신호의 트랜스미션과 사용자로부터 케이블 시스템으로의 요청, 퀴어리 및 그와 유사한 것들의 트랜스미션을 허용한다. 케이블 시스템은 사용자에게 디지탈 트랜스미션을 사용할 수 있거나 또는 위에 언급된 무선 주파수 방송과 유사한 무선 주파수 트랜스미션을 포함할 수 있다. 케이블 시스템 박스(821)는 일시적인 저장을 위하여 버퍼(822)로 디지탈 영상 데이터를 제공하기 위하여 어떠한 필요 변환 회로라도 포함할 수 있다. 버퍼(822)가 역시 영상 시스템 버스에 접속된다는 점을 주목해야 한다.

고해상도 텔레비젼은 트랜스미션의 모드에 상관없이 데이터 압축 포맷으로 매우 유사하게 전송될 것이다. 멀티프로세서 집적 회로(100)는 압축된 데이터를 수신하기 위하여, 이러한 데이터를 개개의 텔레비젼 프레임으로 펼치기 위하여, 또 비디오 램(random access memory, 6)으로 프레임 데이터를 제공하기 위하여 프로그래밍된다. 위에 이미 설명된 바와 같이, 이러한 데이터는 비디오 램(6)으로부터 재호출되며 비디오 팔레트(7)에 제공된다. 비디오 팔레트(7)은 비디오 디스플레이(8)에 적절한 비디오 신호들을 발생시킨다. 디스플레이를 위하여 데이터를 제공할때 영상 데이터를 펼치는 이러한 처리는 멀티프로세서 집적 회로(100) 안에서 또 영상 시스템 버스를 따르는 많은 데이터 이동을 포함한다.

전송 제어기(80)는 이러한 데이터 이동을 제어하기 위하여 마스터 프로세서(60) 및 디지탈 영상/ 그래픽 프로세서(71, 72, 73 및 74)로부터의 패킷 전송을 위한 요청에 응하며, 이러한 데이터 이동을 메모리 리프레시와 같은 다른 영상 시스템 버스 사용과 동등하게 한다.

제40도는 본 발명의 다른 시스템 실시예를 도시한다. 제40도에서는 멀티프로세서 집적 회로(101)이 마스터 프로세서(60) 및 단 하나의 디지탈 영상/그래픽 프로세서(71)을 포함한다. 멀티프로세서 집적 회로(101)는 멀티프로세서 집적 회로(100)보다 작은 실리콘 기판 영역을 필요로 하며 결과적으로 적은 비용으로 구현될 수 있다. 멀티프로세서 집적 회로(101)는 멀티프로세서 집적 회로(100)의 구현에 이미 인용된 바와 같은 방법을 사용하여 구현된다. 각각의 디지탈 영상/그래픽 프로세서의 폭이 자신의 해당 메모리 및 크로스바(50)의 관련 부분의 폭과 같기 때문에, 멀티프로세서 집적 회로(100)는 멀티프로세서 집적 회로(101)의 설계를 얻기 위하여 디지탈 영상/그래픽 프로세서(71 및 72) 사이에서 짤릴 수 있다. 멀티 프로세서 집적 회로(101)는 4개의 디지탈 영상/그래픽 프로세서의 처리 용량이 필요하지 않을 때에 응용에 사용될 수 있다.

멀티프로세서 집적 회로(101)가 제40도에 컬러 팩시밀리 장치의 부분으로서 도시된다. 모뎀(1301)은 발신과 수신을 위하여 전화선과 양방향 결합된다. 모뎀(1301)은 버퍼(1302)에 일시적으로 저장되는 이러한 신호들을 복조한다. 전송 제어기(80)는 디지탈 영상/그래픽 프로세서(71)에 의한 처리를 위해 데이터를 데이터 메모리(22, 23 및 24)로 전송함으로써 버퍼(1302)에 서비스한다. 디지탈 영상/그래픽 프로세서(71)가 들어오는 데이터의 선단을 따를 수 없는 경우에는, 전송 제어기(80)는 역시 버퍼(1302)로부터 메모리(9)로 데이터를 전송시킬 수 있다. 디지탈 영상/그래픽 프로세서(71)는 들어온 팩시밀리의 영상 데이터를 처리한다. 이러한 것은 영상 펼침, 잡음 제거, 오류 보정, 컬러 베이스 보정 및 이러한 것들을 포함한다. 일단 처리되었다면, 전송 제어기(80)는 데이터 메모리(22, 23 및 24)로부터 비디오 램(VRAM, 1303)으로 영상 데이터들을 전송한다. 프린터 제어기(1304)는 프레임 제어기(90)의 제어 아래 영상 데이터를 재호출하며 그것을 하드 카피(hard copy)를 형성하는 컬러 프린터(1305)로 제공한다.

제40도의 장치는 역시 컬러 팩시밀리를 송신할 수 있다. 영상 장치(3)는 소스 문서를 스캐닝한다. 영상 장치(3)는 프레임 제어기(90)의 제어 아래 동작하는 영상 캡쳐 제어기(4)에 처리되지 않은 영상 데이터들을 제공한다. 이러한 영상 데이터는 비디오 램(1303)에 저장된다. 제40도에 도시된 실시예는 분리된 비디오 램을 사용하는 제1도의 실시예와는 달리, 영상 캡쳐 및 영상 디스플레이 모두를 위한 비디오 램(1303)을 공유한다는 것을 주목하여야 한다. 전송 제어기(80)는 이러한 영상 데이터를 데이터 메모리(22, 23 및 24)로 전송한다. 한편 디지탈 영상/그래픽 프로세서(71)는 영상 압축, 오류 보정 리던던시, 컬러 베이스 보정 및 이와 같은 것들을 위하여 영상 데이터를 처리한다. 처리된 데이터는 팩시밀리 트랜스미션을 지원할 필요가 있을 때에, 전송 제어기(80)에 의하여 버퍼(1302)로 전송된다. 상대적인 데이터 율에 좌우되어, 전송 제어기(80)는 버퍼(1303)로의 전송 전에 데이터를 메모리(9)에 일시적으로 저장시킬 수 있다. 이러한 버퍼(1302)에 있는 영상 데이터는 모뎀(1301)에 의하여 변조되어 전화선을 통하여 전송된다.

같은 시스템에 영상 장치 및 컬러 프린터가 함께 있는 것은 이러한 시스템이 역시 컬러 복사기로서 동작할 수 있도록 허용한다. 이러한 경우, 데이터 비교 및 펼침은 필요 없을 수 있다. 그러나, 디지탈 영상/그래픽 프로세서(71)는 여전히 잡음 제거 및 컬러 베이스 보정을 위하여 유용하다. 또한 디지탈 영상/그래픽 프로세서(71)가 원래와는 달리 다른 컬러를 갖는 복사가 가능하도록 하기 위하여 컬러들을 미묘하게 변경(deliberately shifting cokor)시킬 수 있도록 프로그래밍되는 것이 가능하다. 폴스 컬러링(false coloring)으로 알려진 이러한 기법은 데이터의 유동적인 범위를 사용 가능한 프린트 컬러의 유동 범위로 맞추는 데 유용하다.

(57) 청구의 범위

청구항 1.

데이터 프로세서에 있어서,

복수의 저장 위치를 갖는 내부 메모리;

투과 데이터를 저장하는 투과 레지스터(transparency resister) - 상기 투과레지스터는 전송될 데이터의 최소량의 정수배를 저장하고, 상기 투과 데이터는 상기 투과 레지스터 내에서 상기 정수배만큼 반복됨 - ;

상기 메모리에 결합되어, 소스 어드레스를 계산하고, 상기 메모리로부터 전송될 데이터를 상기 소스 어드레스에 대응하는 저장 위치에서 재현(recall)하는 소스 어드레스 제어기;

상기 투과 레지스터 및 상기 소스 어드레스 제어기에 결합되어, 상기 전송될 데이터를 상기 투과 레지스터내에 저장된 상기 투과 데이터와 비교하고, 상기 전송될 데이터가 상기 메모리에 기입될지를 표시하며, 상기 투과 레지스터와 동일한 크기를 갖는 상기 전송될 데이터의 블록을 상기 전송될 데이터의 상기 최소량의 단위로 상기 투과 레지스터에 저장된 상기 투과 데이터와 비교하고, 데이터가 상기 메모리에 기입될지를 표시하는 복수의 표시(indications)를 상기 전송될 데이터의 상기 최소량의 각 단위별로 하나씩 제공하는 비교기; 및

상기 비교기에 결합되어 데스티네이션 어드레스(destination address)를 계산하고, 상기 비교기가 상기 전송될 데이터가 상기 메모리에 기입될 것임을 표시하는 경우에는 상기 전송될 데이터를 상기 데스티네이션 어드레스에 대응하는 저장 위치에서 상기 메모리 내에 기입하고, 상기 비교기가 상기 전송될 데이터가 상기 메모리에 기입될 것임을 표시하지 않는 경우에는 어떠한 데이터도 상기 데스티네이션 어드레스에 대응하는 저장 위치에서 상기 메모리 내에 기입하지 않는 데스티네이션 어드레스 제어기

를 포함하는 데이터 프로세서.

청구항 2.

제1항에 있어서,

상기 메모리는 미리 정해진 데이터폭의 데이터 버스를 갖고,

상기 투과 레지스터는 상기 데이터 버스의 상기 미리 정해진 데이터 버스폭과 동일한 크기를 갖는 데이터 프로세서.

청구항 3.

제1항에 있어서,

상기 전송될 데이터는, 상기 전송될 데이터의 상기 최소량의 정수배인 선택 가능한 크기를 갖는 데이터 워드로 구성되고,

상기 투과 레지스터는, 적어도 데이터 워드의 선택 가능한 크기 중 최대의 크기만큼 큰 크기를 가지며,

상기 비교기는,

각각이 상기 전송될 데이터의 상기 최소량과 동일한 크기의 데이터 단위를 상기 투과 레지스터 내에 저장된 상기 투과 데이터의 대응 부분과 비교하고, 상기 전송될 데이터의 상기 단위가 상기 투과 레지스터 내에 저장된 상기 투과 데이터 중 상기 대응 부분과 일치하는지의 여부를 검출하는 복수의 데이터 비교기; 및

상기 복수의 데이터 비교기에 접속되어, 상기 전송될 데이터가 상기 투과 레지스터 범위에 적당한 상기 선택된 크기의 상기 데이터 워드의 다수배와 동일한 크기의 상기 메모리에 기입될지를 표시하는 상기 복수의 표시를 제공하는 멀티플렉서 - 상기 복수의 표시는 상기 검출된 일치 및 상기 선택된 데이터 크기로부터 형성됨 -

를 포함하는 데이터 프로세서.

청구항 4.

제3항에 있어서,

상기 선택된 데이터 크기는 바이트(8비트), 하프워드(16비트), 워드(32비트) 및 더블워드(64비트) 중에서 선택되고,

상기 투과 레지스터는 64비트를 저장하는 데이터 프로세서.

청구항 5.

제1항에 있어서,

상기 소스 어드레스 제어기 및 상기 데스티네이션 어드레스 제어기는 상기 메모리 내의 미리 정해진 메모리 위치에서 시작하는 상기 각각의 소스 및 데스티네이션 어드레스의 제어를 위한 메모리 전송 제어 데이터를 수신하는 데이터 프로세서.

청구항 6.

제5항에 있어서,

상기 데이터 프로세서는 상기 미리 정해진 메모리 위치에서 시작하는 상기 내부 메모리 내에 저장된 상기 메모리 전송 제어 데이터를 지정하는 데이터 프로세서.

청구항 7.

제1항에 있어서,

상기 내부 메모리, 상기 소스 어드레스 제어기, 상기 비교기 및 상기 데스티네이션 어드레스 제어기를 포함하여, 단일 반도체 칩 상에 배치된 집적 회로를 더 포함하며,

상기 집적 회로는 외부 메모리로의 데이터 통신을 위한 외부 메모리 인터페이스를 더 포함하고,

상기 데이터 프로세서는 상기 메모리 내에 저장된 데이터의 데이터 조작을 위해 상기 메모리에 접속되는 데이터 프로세서.

청구항 8.

제7항에 있어서,

상기 내부 메모리 및 상기 외부 메모리 인터페이스는, 상기 내부 메모리 전용의 제1 어드레스 세트 및 상기 외부 메모리 전용의 다른 제2 어드레스 세트를 갖는 단일 통합 어드레스 공간을 포함하는 데이터 프로세서.

청구항 9.

제8항에 있어서,

상기 소스 어드레스 제어기는 상기 제1 어드레스 세트 내에서 상기 소스 어드레스를 계산하고,

상기 데스티네이션 어드레스 제어기는 상기 제1 어드레스 세트 내에서 상기 데스티네이션 어드레스를 계산하는 데이터 프로세서.

청구항 10.

제8항에 있어서,

상기 소스 어드레스 제어기는 상기 제1 어드레스 세트 내에서 상기 소스 어드레스를 계산하고,

상기 데스티네이션 어드레스 제어기는 상기 제2 어드레스 세트 내에서 상기 데스티네이션 어드레스를 계산하는 데이터 프로세서.

청구항 11.

제8항에 있어서,

상기 소스 어드레스 제어기는 상기 제2 어드레스 세트 내에서 상기 소스 어드레스를 계산하고,

상기 데스티네이션 어드레스 제어기는 상기 제1 어드레스 세트 내에서 상기 데스티네이션 어드레스를 계산하는 데이터 프로세서.

청구항 12.

제8항에 있어서,

상기 소스 어드레스 제어기는 상기 제2 어드레스 세트 내에서 상기 소스 어드레스를 계산하고,

상기 데스티네이션 어드레스 제어기는 상기 제2 어드레스 세트 내에서 상기 데스티네이션 어드레스를 계산하는 데이터 프로세서.

청구항 13.

제7항에 있어서,

상기 데이터 프로세서는 상기 투과 레지스터내에 저장된 상기 투과 데이터를 지정하는 데이터 프로세서.

청구항 14.

제7항에 있어서,

상기 집적 회로는, 상기 내부 메모리 내에 저장된 데이터의 데이터 조작을 위해 상기 내부 메모리에 접속된 적어도 하나의 추가 데이터 프로세서; 및

상기 적어도 하나의 추가 데이터 프로세서 각각에 접속되어, 상기 데이터 프로세서와 동일한 방식으로 상기 적어도 하나의 추가 데이터 프로세서 각각에 응답하는 데이터 전송 제어기

를 상기 단일 반도체 칩 상에 더 포함하는 데이터 프로세서.

청구항 15.

제1항에 있어서,

상기 메모리 및 상기 비교기에 접속되어, 상기 비교기가 상기 전송될 대응 데이터가 상기 메모리에 기입될 것으로 표시하는 경우, 상기 전송될 데이터의 상기 최소량의 각 단위마다 상기 전송될 데이터가 상기 메모리에 기입되도록 하게 하는 기입 신호를 생성하기 위한 메모리 기입 신호 발생기

를 더 포함하는 데이터 프로세서.

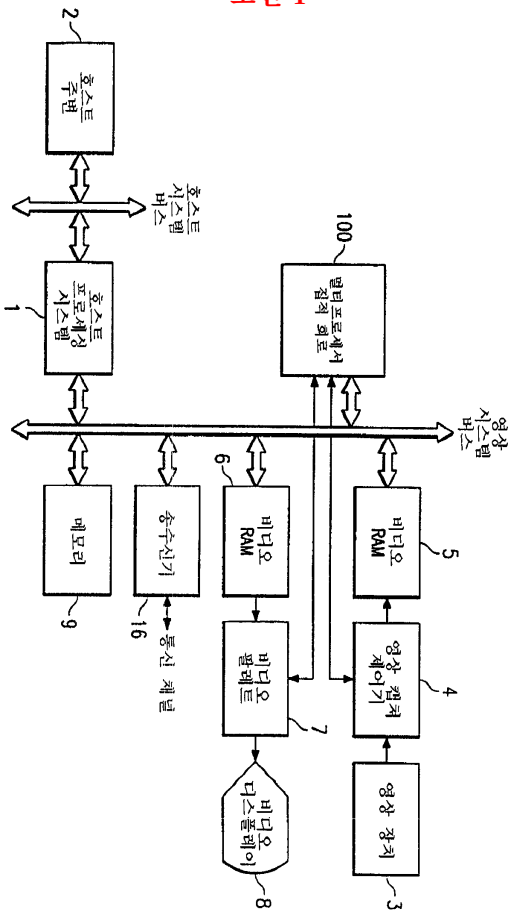
청구항 16.

제15항에 있어서,

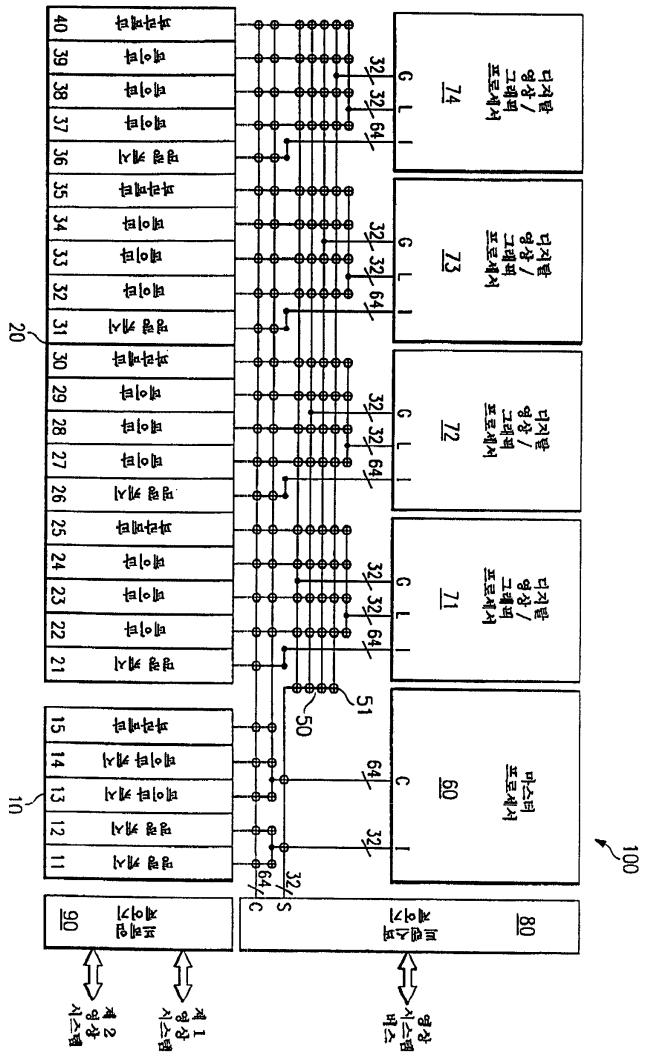
상기 비교기는, 상기 전송될 데이터가 상기 투과 데이터와 일치하는 경우, 상기 전송될 데이터가 상기 메모리에 기입되지 않음을 나타내는 데이터 프로세서.

도면

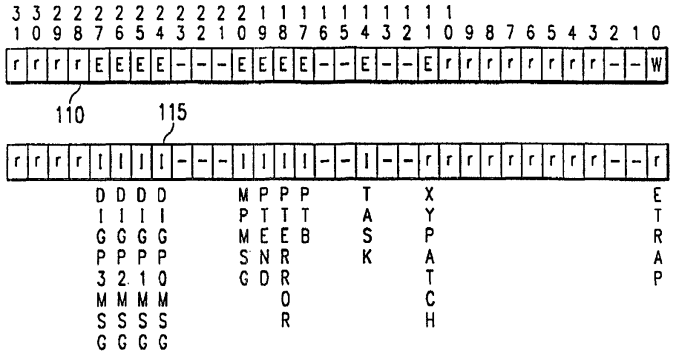
도면 1


2
호스트 주변
호스트 시스템 버스
1
호스트 프로세싱 시스템
100
멀티프로세서 컴퓨팅 회로
영상 시스템 버스
메모리
9
송수신기
16
통신 채널
6
비디오 RAM
5
비디오 RAM
비디오 팔레트
7
영상 캡쳐 제어기
4
비디오 디스플레이
8
영상 장치
3

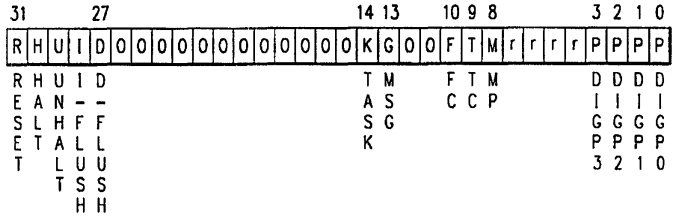
도면 2


디지탈 영상/그래픽 프로세서
74
73
72
71
마스터 프로세서
60
80
90
20
10
100
50
51
32
64
G
L
I
C
S
제1 영상 시스템
제2 영상 시스템
영상 시스템

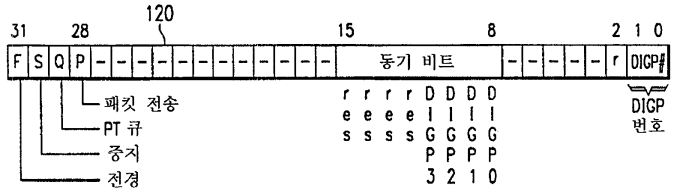
도면 3


31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
r r r r E E E E - - - E E E - - E - - E r r r r r r r - - W
110
115
r r r r I I I I - - - I I I - - I - - r r r r r r r - - r
DIGP3MSG
DIGP2MSG
DIGP1MSG
DIGP0MSG
MPMSG
PTENDR
PTERO
PTB
TASK
XYPATCH
ETRAP

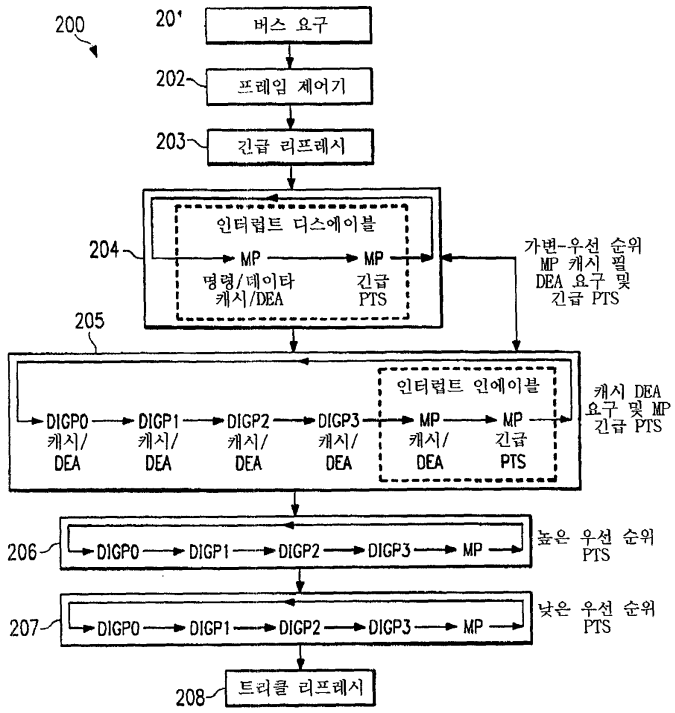
도면 4


31 27 14 13 10 9 8 3 2 1 0
R H U I D 0 0 0 0 0 0 0 0 0 0 0 0 0 0 K G 0 0 F T M r r r P P P P
RESET
HANGLT
UNHALTSH
IFLUSH
DFLUSH
TASK
MSG
FCC
TCP
MP
DIGP3
DIGP2
DIGP1
DIGP0

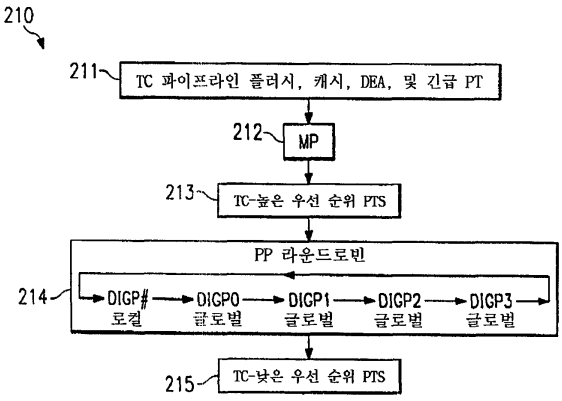
도면 5


120
31 28 15 8 2 1 0
F S Q P - - - - - - - - - - - - - - 동기 비트 - - - - - r DIGP#
패킷 전송
PT 큐
중지
전경
reserved
reserved
reserved
reserved
DIGP3
DIGP2
DIGP1
DIGP0
DIGP 번호

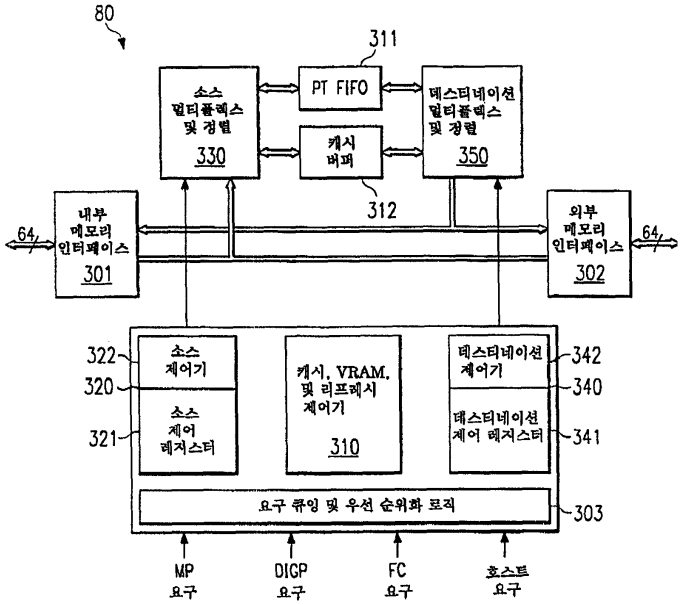
도면 6


200
20'
버스 요구
202
프레임 제어기
203
긴급 리프레시
204
인터럽트 디스에이블
MP
MP
명령/데이타 캐시/DEA
긴급 PTS
가변-우선 순위 MP 캐시 필 DEA 요구 및 긴급 PTS
205
인터럽트 인에이블
DIGP0 캐시/ DEA
DIGP1 캐시/ DEA
DIGP2 캐시/ DEA
DIGP3 캐시/ DEA
MP 캐시/ DEA
MP 긴급 PTS
캐시 DEA 요구 및 MP 긴급 PTS
206
DIGP0 DIGP1 DIGP2 DIGP3 MP
높은 우선 순위 PTS
207
DIGP0 DIGP1 DIGP2 DIGP3 MP
낮은 우선 순위 PTS
208
트리클 리프레시

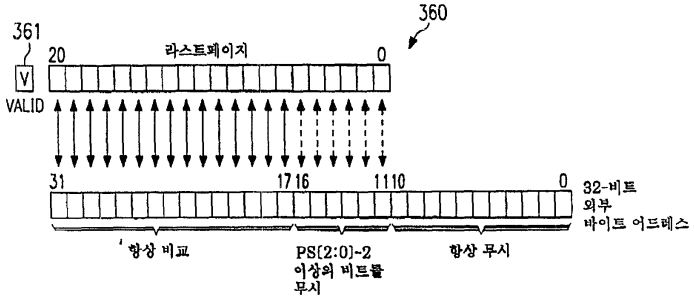
도면 7


210
211
TC 파이프라인 플러시, 캐시, DEA, 및 긴급 PT
212
MP
213
TC-높은 우선 순위 PTS
PP 라운드로빈
214
DIGP# 로컬
DIGP0 글로벌
DIGP1 글로벌
DIGP2 글로벌
DIGP3 글로벌
215
TC-낮은 우선 순위 PTS

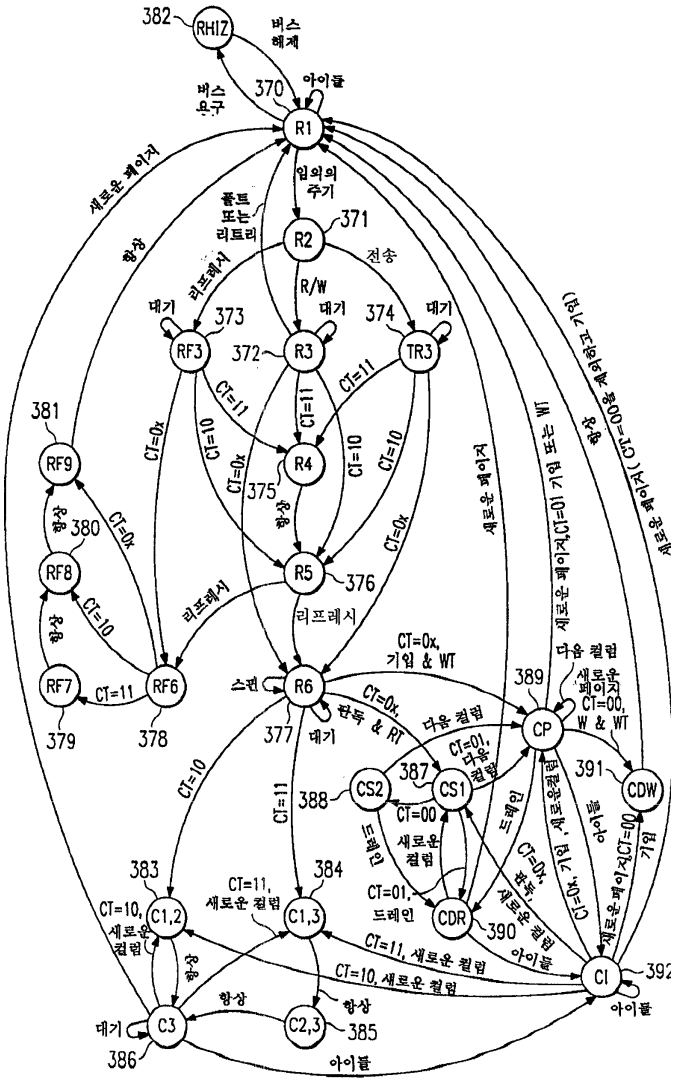
도면 8


80
311
소스 멀티플렉스 및 정렬
330
PT FIFO
캐시 버퍼
데스티네이션 멀티플렉스 및 정렬
350
312
내부 메모리 인터페이스
301
외부 메모리 인터페이스
302
64
64
322
320
321
소스 제어기
소스 제어 레지스터
캐시, VRAM, 및 리프레시 제어기
310
데스티네이션 제어기
데스티네이션 제어 레지스터
342
340
341
요구 큐잉 및 우선 순위화 로직
303
MP 요구
DIGP 요구
FC 요구
호스트 요구

도면 9

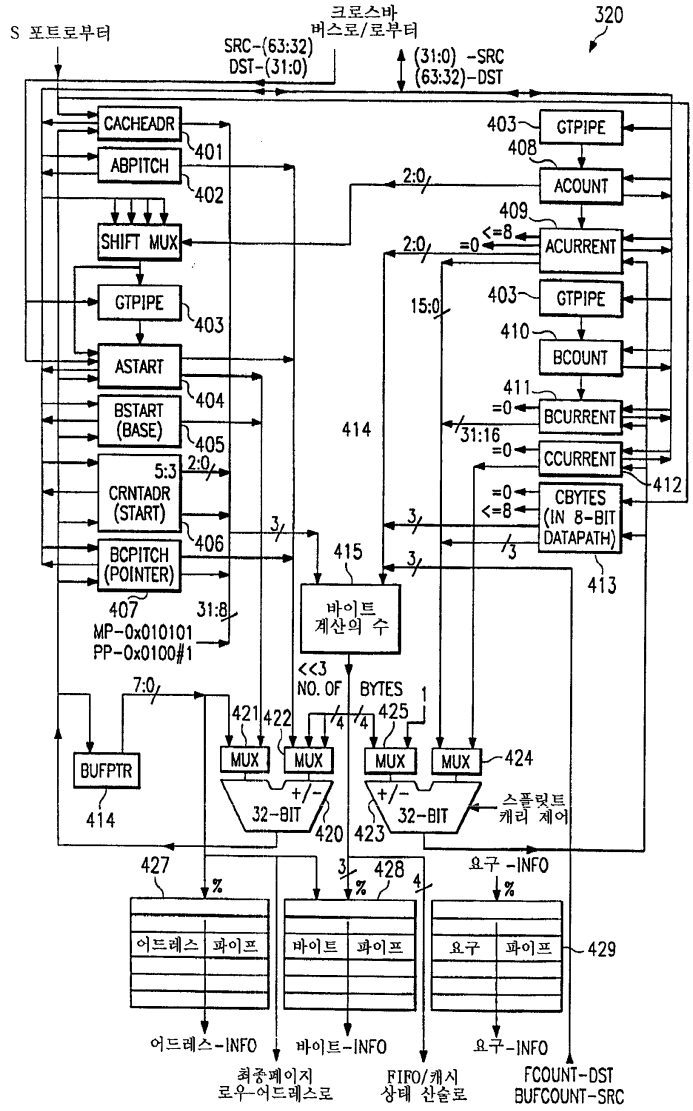

360
361
V
VALID
20
라스트페이지
0
31
17 16
1110
0
32-비트 외부 바이트 어드레스
항상 비교
PS[2:0]~2 이상의 비트를 무시
항상 무시

도면 10

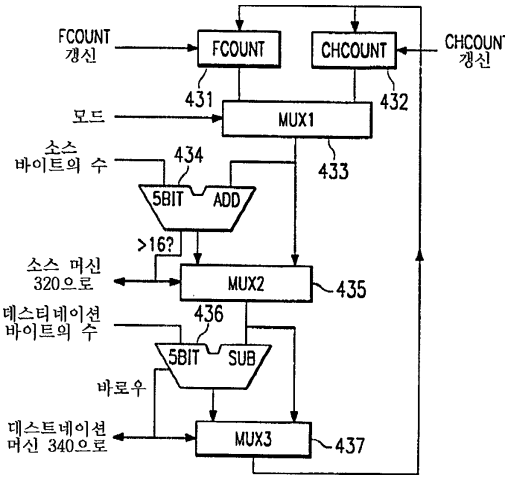


382
RHIZ
버스 해제
370
아이들
버스 요구
R1
새로운 페이지
항상
임의의 주기
폴트 또는 리트리
371
R2
전송
리프레시
R/W
대기
373
RF3
372
R3
374
TR3
CT=11
CT=10
CT=0x
381
RF9
R4
375
항상
380
RF8
R5
376
리프레시
RF7
RF6
CT=11
379
378
스핀
R6
377
대기
CT=0x, 기입 & WT
CT=0x, 판독 & RT
389
CP
다음 컬럼
새로운 페이지
CT=00, W & WT
CT=01, 다음 컬럼
387
CS1
388
CS2
CT=00
새로운 컬럼
391
CDW
드레인
CT=01, 드레인
CDR
390
아이들
383
C1,2
384
C1,3
CT=11, 새로운 컬럼
CT=10, 새로운 컬럼
CT=11, 새로운 컬럼
CT=10, 새로운 컬럼
CI
392
아이들
C3
386
C2,3
385

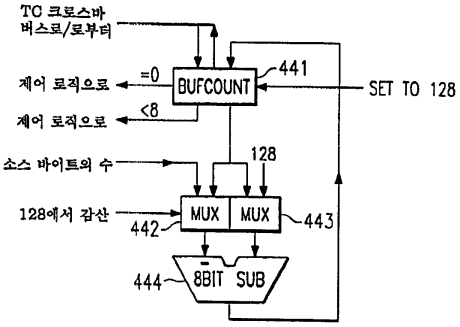
도면 11


S 포트로부터
크로스바
버스로/로부터
SRC-(63:32)
DST-(31:0)
(31:0) -SRC
(63:32)-DST
320
CACHEADR
401
ABPITCH
402
SHIFT MUX
GTPIPE
403
ASTART
404
BSTART
(BASE)
405
CRNTADR
(START)
406
BCPITCH
(POINTER)
407
MP-0x010101
PP-0x0100#1
403
GTPIPE
408
ACOUNT
409
ACURRENT
403
GTPIPE
410
BCOUNT
411
BCURRENT
CCURRENT
412
CBYTES
(IN 8-BIT
DATAPATH)
413
414
415
바이트
계산의 수
<<3
NO. OF BYTES
BUFPTR
414
421
422
425
MUX
424
32-BIT
420
423
스플릿트
캐리 제어
427
428
429
어드레스 파이프
바이트 파이프
요구 파이프
요구 -INFO
어드레스-INFO
바이트-INFO
요구-INFO
최종페이지
로우-어드레스로
FIFO/캐시
상태 산술로
FCOUNT-DST
BUFCOUNT-SRC

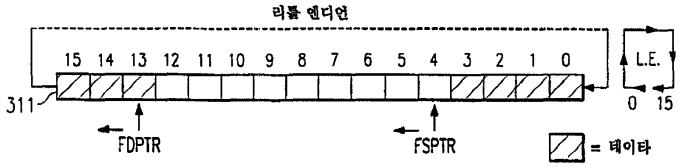
도면 12



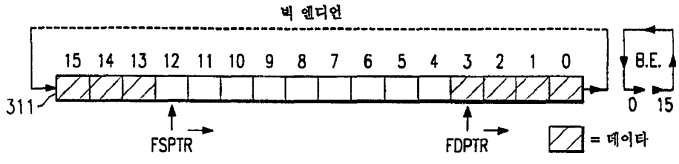
도면 13



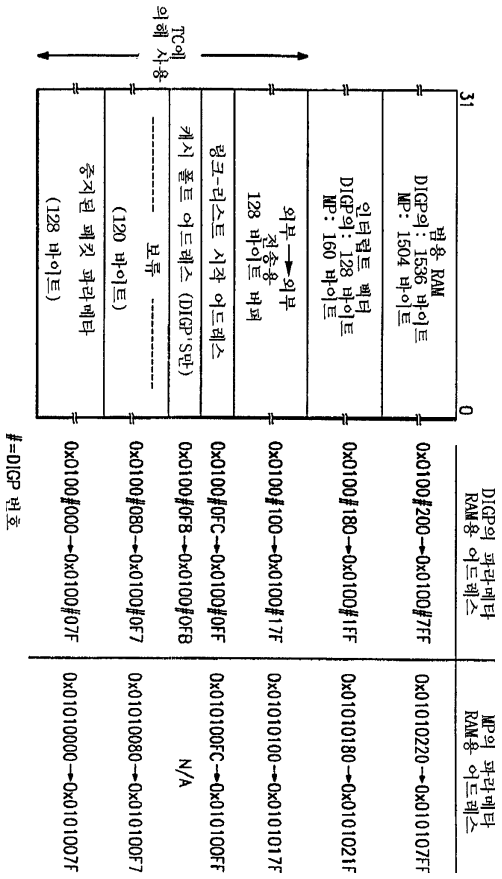
도면 14a



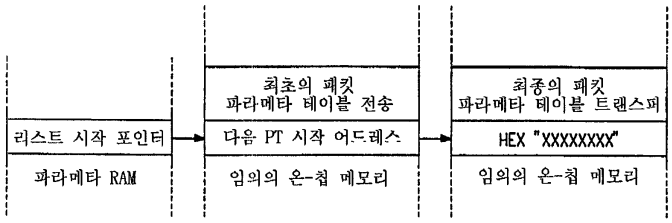
도면 14b



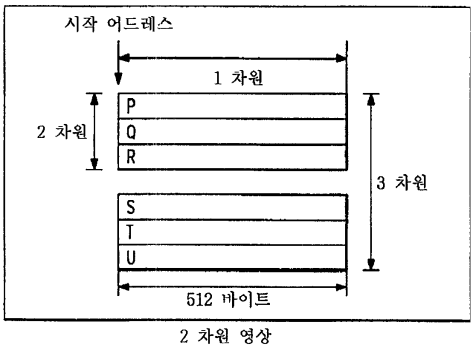
도면 15


31
0
범용 RAM
DIGP의: 1536 바이트
MP: 1504 바이트
인터럽트 벡터
DIGP의: 128 바이트
MP: 160 바이트
외부→외부
전송용
128 바이트 버퍼
링크-리스트 시작 어드레스
캐시 폴트 어드레스 (DIGP'S만)
---------- 보류 ----------
(120 바이트)
중지된 패킷 파라메타
(128 바이트)
TC에
의해 사용
DIGP의 파라메타
RAM용 어드레스
0x0100#200→0x0100#7FF
0x0100#180→0x0100#1FF
0x0100#100→0x0100#17F
0x0100#0FC→0x0100#0FF
0x0100#0F8→0x0100#0FB
0x0100#080→0x0100#0F7
0x0100#000→0x0100#07F
#=DIGP 번호
MP의 파라메타
RAM용 어드레스
0x01010220→0x010107FF
0x01010180→0x0101021F
0x01010100→0x0101017F
0x0101000FC→0x010100FF
N/A
0x01010080→0x010100F7
0x01010000→0x0101007F

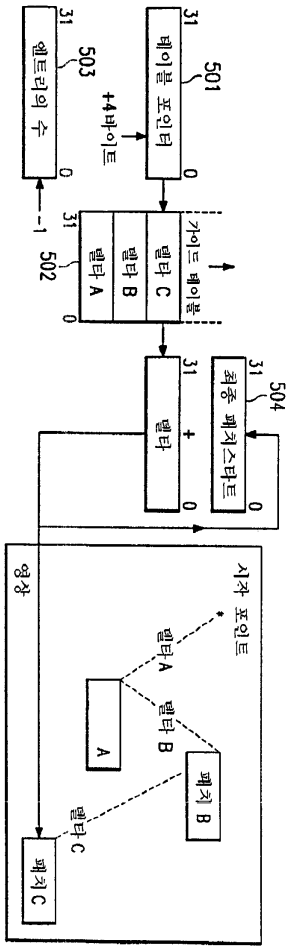
도면 16


최종의 패킷
파라메타 테이블 트랜스퍼
HEX "XXXXXXXX"
임의의 온-칩 메모리
최초의 패킷
파라메타 테이블 전송
다음 PT 시작 어드레스
임의의 온-칩 메모리
리스트 시작 포인터
파라메타 RAM

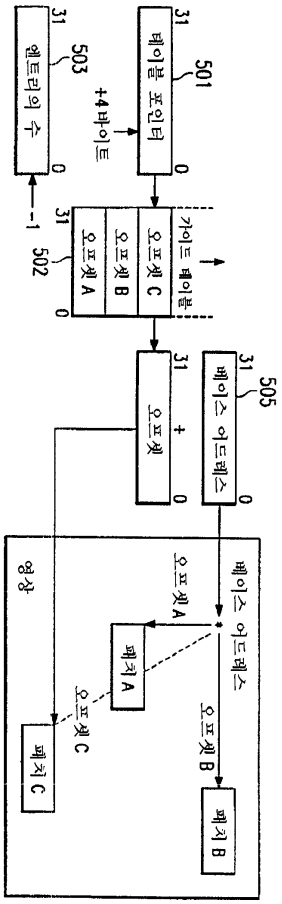
도면 17


시작 어드레스
1 차원
2 차원
3 차원
P
Q
R
S
T
U
512 바이트
2 차원 영상

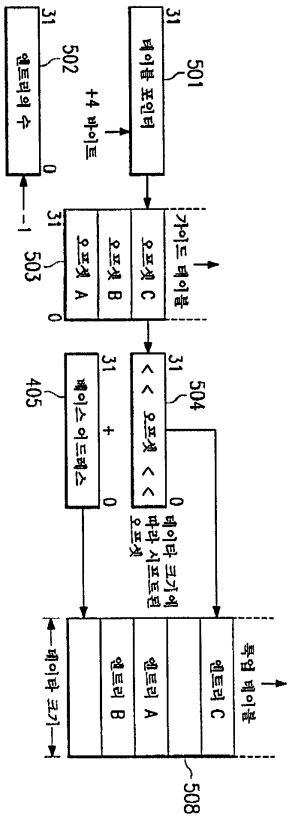
도면 18


31
웹트리의 수
503
0
+4바이트
31
테이블 포인터
501
0
-1
31
델타 A
델타 B
델타 C
502
0
가이드 테이블
31
델타
+
0
31
최종 패치스타트
504
0
영상
시작 포인트
델타A
델타 B
델타 C
A
패치 B
패치 C

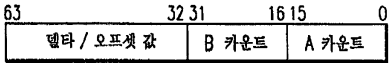
도면 19


31
501
0
테이블 포인터
+4 바이트
31
503
0
엘트리의 수
-1
31
502
0
오프셋 A
오프셋 B
오프셋 C
가이드 테이블
31
0
+
오프셋
31
505
0
베이스 어드레스
영상
오프셋 A
베이스 어드레스
패치 A
오프셋 C
오프셋 B
패치 C
패치 B

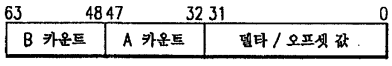
도면 20



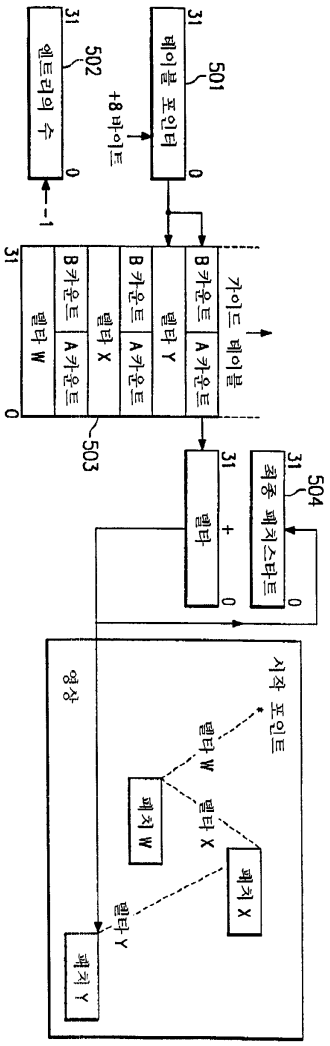
도면 21



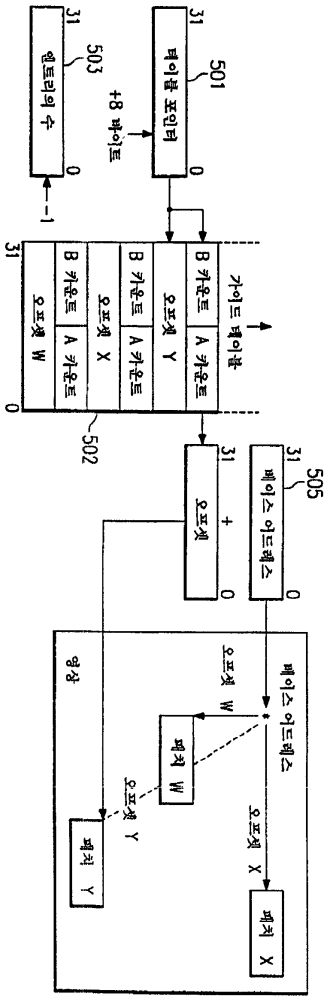
도면 22



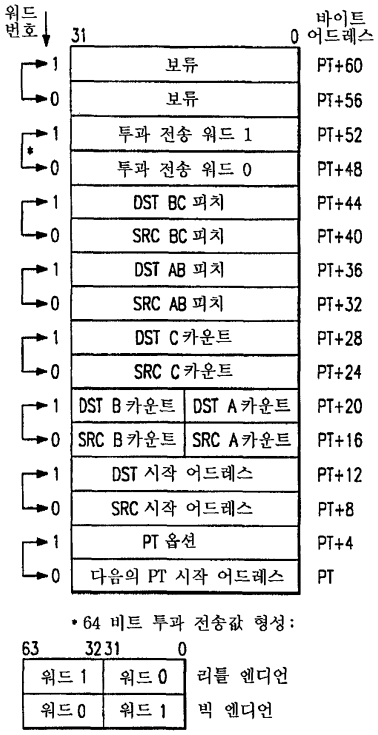
도면 23


엔트리의 수
502
-1
31
0
테이블 포인터
501
+8 바이트
가이드 테이블
B 카운트
A 카운트
밸타 Y
밸타 X
밸타 W
503
최종 패치스타트
504
밸타
+
영상
시작 포인트
밸타 W
밸타 X
밸타 Y
패치 W
패치 X
패치 Y

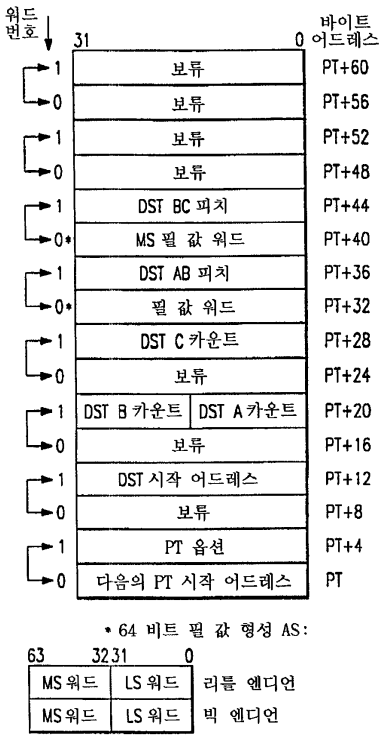
도면 24


31
테이블 포인터
501
0
+8 바이트
31
엔트리의 수
503
0
-1
31
B 카운트
A 카운트
오프셋 Y
B 카운트
A 카운트
오프셋 X
B 카운트
A 카운트
오프셋 W
0
가이드 테이블
502
31
+
오프셋
0
31
베이스 어드레스
505
0
영상
오프셋 W
폐치 W
오프셋 Y
폐치 Y
베이스 어드레스
오프셋 X
폐치 X

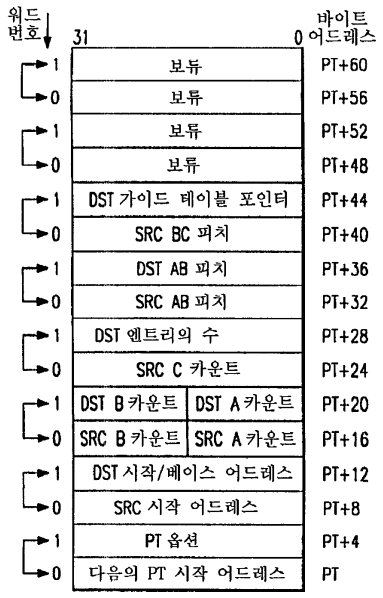
도면 25


워드 번호
31
0
바이트 어드레스
1
0
보류
PT+60
보류
PT+56
투과 전송 워드 1
PT+52
투과 전송 워드 0
PT+48
DST BC 피치
PT+44
SRC BC 피치
PT+40
DST AB 피치
PT+36
SRC AB 피치
PT+32
DST C 카운트
PT+28
SRC C 카운트
PT+24
DST B 카운트
DST A 카운트
PT+20
SRC B 카운트
SRC A 카운트
PT+16
DST 시작 어드레스
PT+12
SRC 시작 어드레스
PT+8
PT 옵션
PT+4
다음의 PT 시작 어드레스
PT
• 64 비트 투과 전송값 형성:
63
32
31
0
워드 1
워드 0
리틀 엔디언
워드 0
워드 1
빅 엔디언

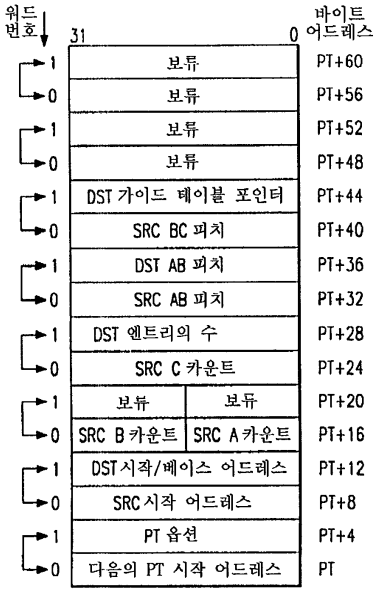
도면 26


워드 번호
31
0
바이트 어드레스
1
보류
PT+60
0
보류
PT+56
1
보류
PT+52
0
보류
PT+48
1
DST BC 피치
PT+44
0*
MS 필 값 워드
PT+40
1
DST AB 피치
PT+36
0*
필 값 워드
PT+32
1
DST C 카운트
PT+28
0
보류
PT+24
1
DST B 카운트
DST A 카운트
PT+20
0
보류
PT+16
1
DST 시작 어드레스
PT+12
0
보류
PT+8
1
PT 옵션
PT+4
0
다음의 PT 시작 어드레스
PT
* 64 비트 필 값 형성 AS:
63
32
31
0
MS 워드
LS 워드
리틀 엔디언
MS 워드
LS 워드
빅 엔디언

도면 27


워드 번호
31
0
바이트 어드레스
1
보류
PT+60
0
보류
PT+56
1
보류
PT+52
0
보류
PT+48
1
DST 가이드 테이블 포인터
PT+44
0
SRC BC 피치
PT+40
1
DST AB 피치
PT+36
0
SRC AB 피치
PT+32
1
DST 엔트리의 수
PT+28
0
SRC C 카운트
PT+24
1
DST B 카운트
DST A 카운트
PT+20
0
SRC B 카운트
SRC A 카운트
PT+16
1
DST 시작/베이스 어드레스
PT+12
0
SRC 시작 어드레스
PT+8
1
PT 옵션
PT+4
0
다음의 PT 시작 어드레스
PT

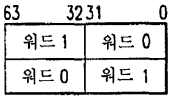
도면 28



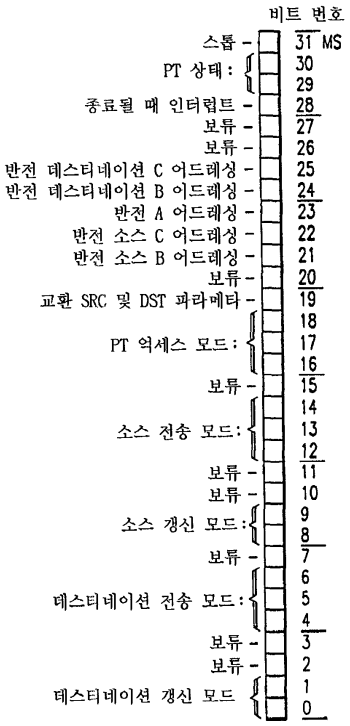
도면 29



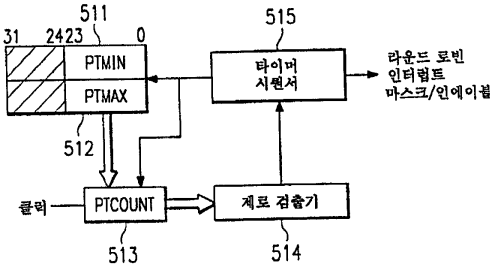
* 64 비트 컬러 레지스터 값 형성 AS :



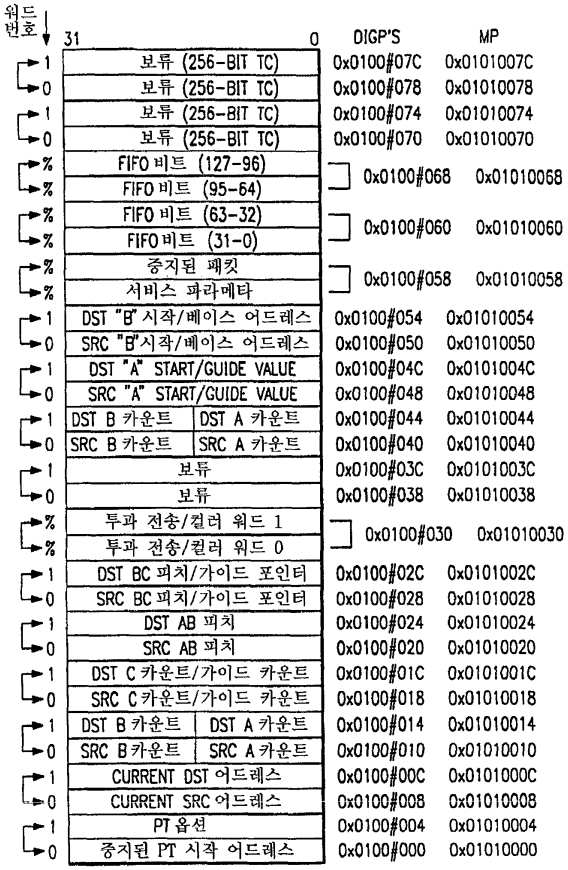
도면 30


비트 번호
스톱
31 MS
PT 상태:
30
29
종료될 때 인터럽트
28
보류
27
보류
26
반전 데스티네이션 C 어드레싱
25
반전 데스티네이션 B 어드레싱
24
반전 A 어드레싱
23
반전 소스 C 어드레싱
22
반전 소스 B 어드레싱
21
보류
20
교환 SRC 및 DST 파라메타
19
18
PT 억세스 모드:
17
16
보류
15
14
소스 전송 모드:
13
12
보류
11
보류
10
9
소스 갱신 모드:
8
보류
7
6
데스티네이션 전송 모드:
5
4
보류
3
보류
2
데스티네이션 갱신 모드
1
0

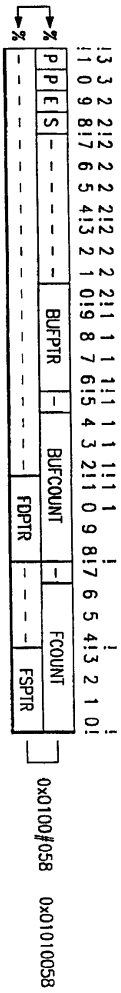
도면 31


511
31
2423
0
PTMIN
PTMAX
512
515
타이머
시퀀서
라운드 로빈
인터럽트
마스크/인에이블
클럭
PTCOUNT
제로 검출기
513
514

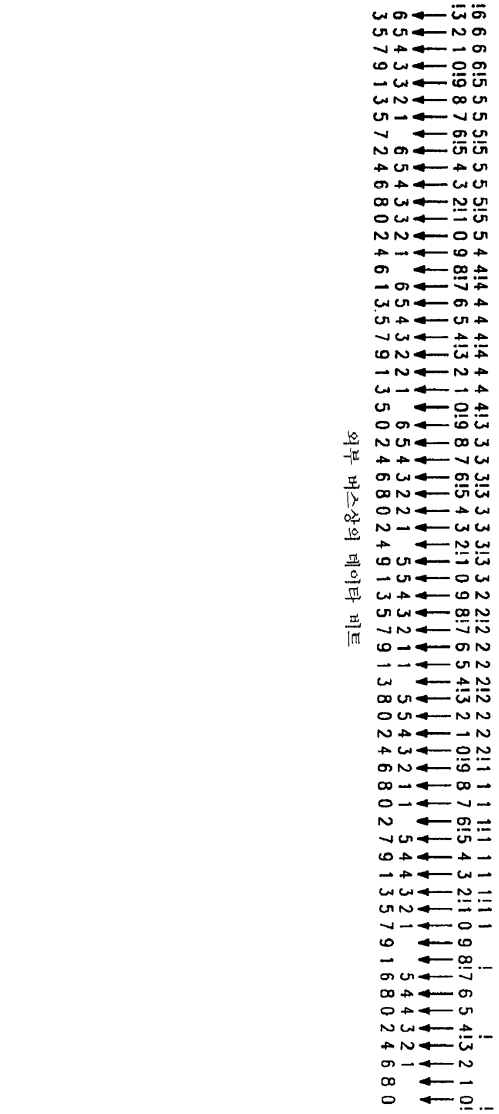
도면 32


워드 번호
31
0
DIGP'S
MP
보류 (256-BIT TC) 0x0100#07C 0x0101007C
보류 (256-BIT TC) 0x0100#078 0x01010078
보류 (256-BIT TC) 0x0100#074 0x01010074
보류 (256-BIT TC) 0x0100#070 0x01010070
FIFO 비트 (127-96)
FIFO 비트 (95-64) 0x0100#068 0x01010068
FIFO 비트 (63-32)
FIFO 비트 (31-0) 0x0100#060 0x01010060
중지된 패킷
서비스 파라메타 0x0100#058 0x01010058
DST "B" 시작/베이스 어드레스 0x0100#054 0x01010054
SRC "B"시작/베이스 어드레스 0x0100#050 0x01010050
DST "A" START/GUIDE VALUE 0x0100#04C 0x0101004C
SRC "A" START/GUIDE VALUE 0x0100#048 0x01010048
DST B 카운트 DST A 카운트 0x0100#044 0x01010044
SRC B 카운트 SRC A 카운트 0x0100#040 0x01010040
보류 0x0100#03C 0x0101003C
보류 0x0100#038 0x01010038
투과 전송/컬러 워드 1
투과 전송/컬러 워드 0 0x0100#030 0x01010030
DST BC 피치/가이드 포인터 0x0100#02C 0x0101002C
SRC BC 피치/가이드 포인터 0x0100#028 0x01010028
DST AB 피치 0x0100#024 0x01010024
SRC AB 피치 0x0100#020 0x01010020
DST C 카운트/가이드 카운트 0x0100#01C 0x0101001C
SRC C 카운트/가이드 카운트 0x0100#018 0x01010018
DST B 카운트 DST A 카운트 0x0100#014 0x01010014
SRC B 카운트 SRC A 카운트 0x0100#010 0x01010010
CURRENT DST 어드레스 0x0100#00C 0x0101000C
CURRENT SRC 어드레스 0x0100#008 0x01010008
PT 옵션 0x0100#004 0x01010004
중지된 PT 시작 어드레스 0x0100#000 0x01010000

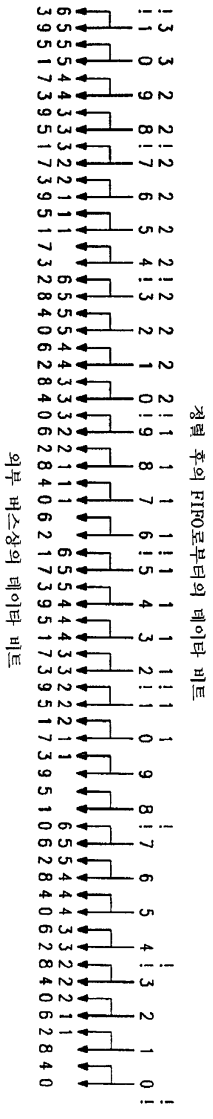
도면 33


31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
PP ES
BUFPTR
BUFCOUNT
FDPTR
FCOUNT
FSPTR
0x0100#058
0x01010058

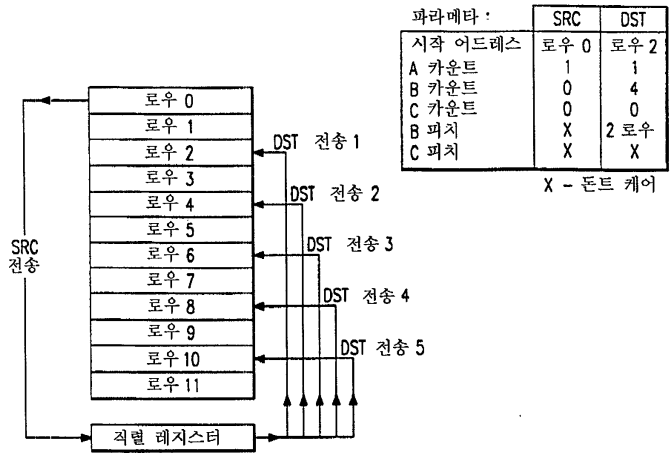
도면 34



도면 35



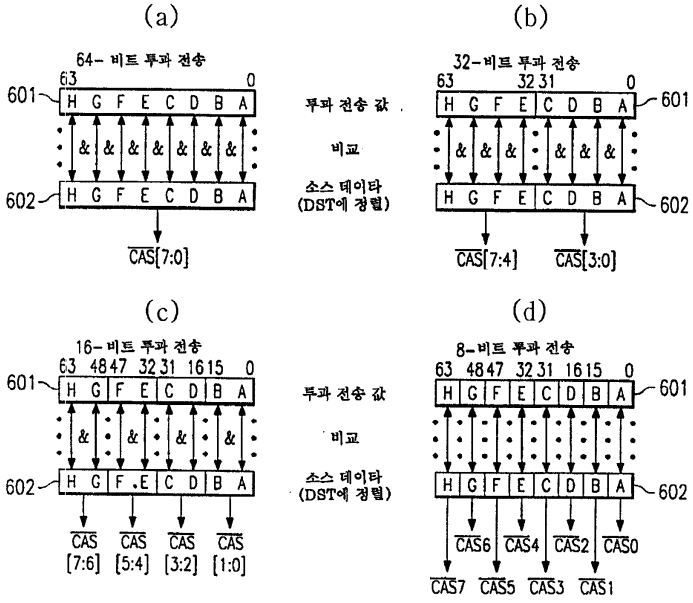
도면 36



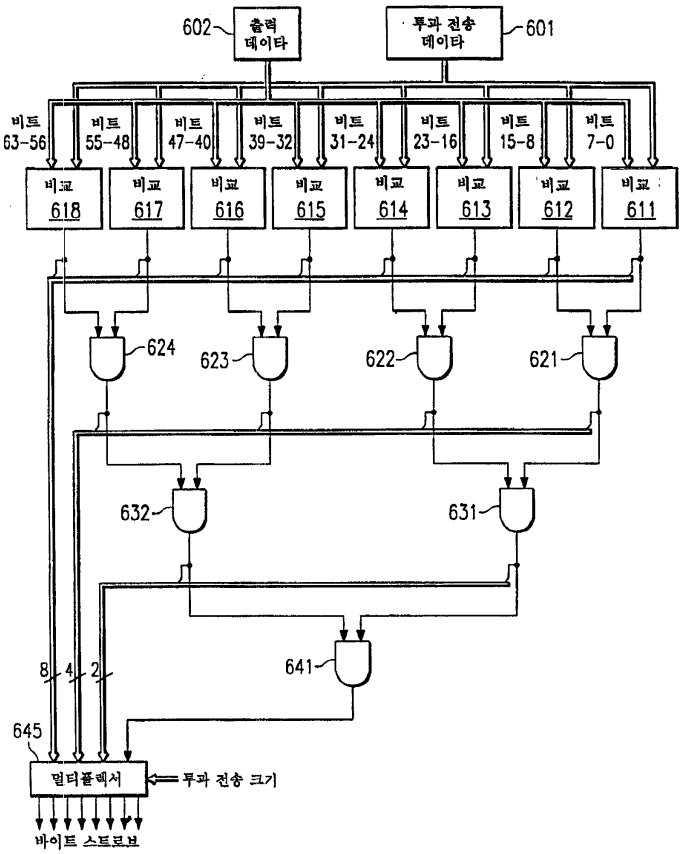
파라메타 :	SRC	DST
시작 어드레스	로우 0	로우 2
A 카운트	1	1
B 카운트	0	4
C 카운트	0	0
B 피치	X	2 로우
C 피치	X	X

X - 돈트 케어

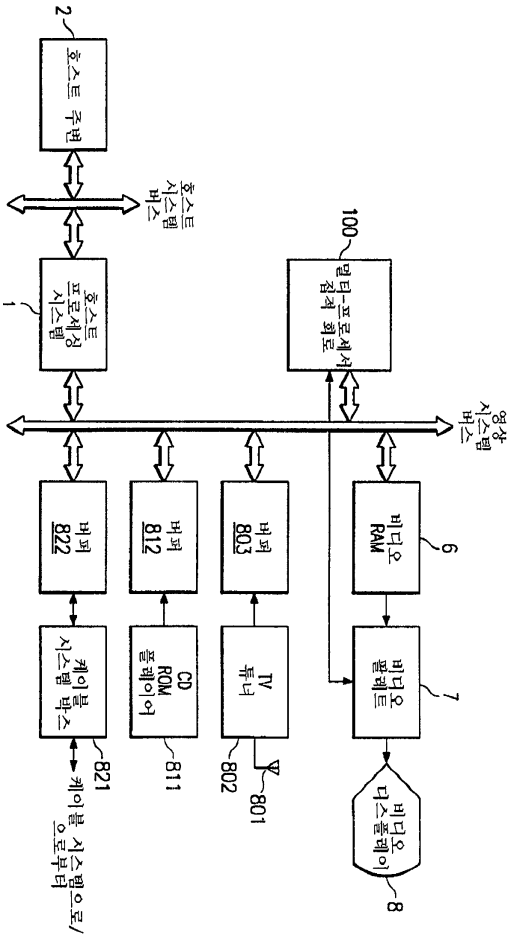
도면 37



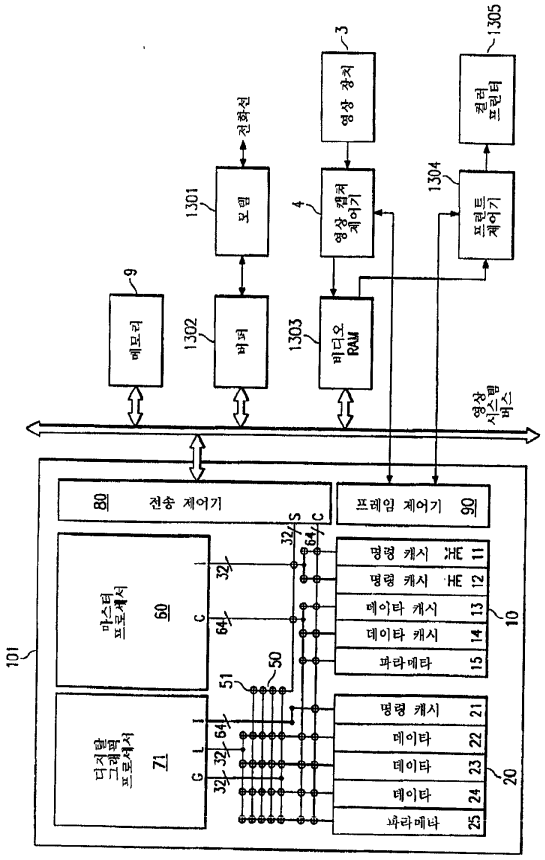
도면 38


602
출력 데이타
투과 전송 데이타
601
비트 63-56
비트 55-48
비트 47-40
비트 39-32
비트 31-24
비트 23-16
비트 15-8
비트 7-0
비교 618
비교 617
비교 616
비교 615
비교 614
비교 613
비교 612
비교 611
624
623
622
621
632
631
641
8
4
2
645
멀티플렉서
투과 전송 크기
바이트 스트로브

도면 39


2
호스트 주변
호스트 시스템 버스
100
멀티-프로세서 회로
1
호스트 프로세싱 시스템
영상 시스템 버스
버퍼 822
버퍼 812
버퍼 803
비디오 RAM
6
케이블 박스 시스템
CD ROM 플레이어
TV 튜너
비디오 믹서
7
821
811
802
801
케이블 시스템으로/ 으로부터
비디오 디스플레이
8

도면 40


메모리
9
버퍼
1302
모뎀
1301
전화선
영상 장치
3
영상 캡쳐 제어기
4
비디오 RAM
1303
프린트 제어기
1304
컬러 프린터
1305
영상 시스템 버스
전송 제어기
80
프레임 제어기
90
마스터 프로세서
60
디지탈 그래픽 프로세서
71
101
명령 캐시
데이타 캐시
파라메타
10
20
51
50