(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl. [7]
G06F 12/06

(11) 공개번호 특2001-0086035
(43) 공개일자 2001년09월07일

(21) 출원번호 10-2001-7006145
(22) 출원일자 2001년05월16일
번역문 제출일자 2001년05월16일
(86) 국제출원번호 PCT/US1999/27021
(86) 국제출원출원일자 1999년11월15일
(87) 국제공개번호 WO 2000/29957
(87) 국제공개일자 2000년05월25일

(81) 지정국 국내특허 : 중국, 독일, 영국, 일본, 대한민국, 미국,

(30) 우선권주장 60/108,930 1998년11월16일 미국(US)
09/439,303 1999년11월12일 미국(US)

(71) 출원인 인피네온 테크놀로지스 아게
마이클 골위저, 호레스트 쉐퍼
독일, 뮌헨 데-81669, 세인트-마틴-슈트라쎄 53

(72) 발명자 스트라코프스키,헨리
미국,캘리포니아95118,산호세,체리애비뉴3962
스자벨스키,피오트르
미국,캘리포니아95054,산타클라라,밀크리크레인590,아파트먼트103

(74) 대리인 강명구

심사청구 : 없음

(54) 메모리 장치의 물리적 페이지 상태를 탐색하기 위해페이지 태그 레지스터를 이용하는 방법

요약

메모리 제어기(904)에 의해 멀티뱅크형 메모리(908)에 포함되는 메모리 페이지의 메모리 접근 속도를 향상시키는 방법이 공개된다. 요청된 뱅크 번호와 요청된 페이지 번호를 포함하는 메모리 제어기(904)에 의해 유입 시스템 어드레스 요청이 수신된다. 요청된 뱅크 번호에 상응하는 페이지 레지스터가 비교기에 의해 위치하며, 위치한 페이지 레지스터에 포함된 저장된 페이지 어드레스는 요청된 페이지 어드레스와 비교된다. 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치할 때, 요청된 뱅크 번호에 상응하는 메모리 뱅크의 요청 페이지에 접근이 이루어진다.

대표도
도 9b

명세서

기술분야

본 발명은 연산 시스템에 관한 것이다. 특히, 본 발명은 멀티-프로세서 컴퓨터 시스템 등과 같은 연산 시스템의 공유 리소스에 접근을 제공하는 것에 관한 것이다. 특히, 메모리 제어기에 의해 멀티뱅크형 메모리에 포함되는 메모리 페이지의 메모리 접근 속도를 향상시키는 방법이 기술된다.

배경기술

기본 컴퓨터 시스템에서, 중앙 처리 장치(CPU)는 메모리에 저장된 명령 세트나 지정 프로그램에 따라 작동한다. 프로세서가 작동할 때 저장된 명령 세트나 프로그램에 추가하여, 처리 중 CPU의 정보 조작을 용이하게 하기 위해 프로세서 메모리나 관련 부가 메모리 내의 메모리 공간이 제공된다. 추가적인 메모리는 프로세서에 의해 생성되는 정보 저장 공간을 제공할 뿐 아니라, 프로그램 실행을 위해 프로세서가 사용하는 일시나 " 스크래치패드(scratchpad)" 원칙 상의 정보 저장 공간을 제공한다. 추가적으로, 관련 메모리는 시스템의 출력 장치로 이용하기 위해 명령 세트를 작동시키는 프로세서의 출력 정보가 놓이는 위치를 제공한다.

메모리에 접근하기 위해 여러 부품(프로세서, 하드 드라이브 등)이 공통 버스를 공유하여야하는 시스템에서, 메모리 접근시 충돌 가능성이 상존한다. 멀티프로세서 컴퓨터 시스템의 경우에 특히, 다른 프로세서를 이용하는 시스템이 동시에 작동할 경우, 메모리나 다른 공유 리소스에 접근하는 것은 복잡해진다. 각각의 프로세서나 프로세서 시스템이 동일 메모리에 동시 접근을 원하기 쉽기 때문에, 프로세서간의 충돌을 피할 수 없다. 특히, 멀티프로세서 컴퓨터 시스템에서, 두 개 이상의 프로세서나 프로세서 시스템의 작동이 공통 메모리나 그외 다른 공유 리소스에 대해 메모리 명령의 중복을 주기적으로 가져온다.

공유 메모리로의 메모리 접근 요청의 충돌 문제를 해결하는 기본 접근법은 프로세서 시스템의 고립과, 프로세서 각각에 사용되는 메모리의 완전한 리던던시를 포함한다. 그러나, 이 접근법은 멀티프로세서 시스템의 의도된 장점을 저하시키는 경우가 잦다. 이러한 멀티프로세서 시스템은 한 개의 프로세서가 다른 한 프로세서의 작동을 지원할 때 동일한 데이터에 병렬 연산 작동을 제공하는 방식으로 작동할 경우 가장 효율적이다. 일반적으로, 이러한 프로세서 시스템은 메모리와 같은 공유 리소스에 접근을 위해 프로세서가 경쟁할 때 시간 공유 방식을 취하거나, 각각의 프로세서가 고유 메모리 버스를 가질 때 프로세서 시스템이 이중 포트를 취할 수 있다. 가령, 한 포트는 대기 행렬에, 한 포트는 접근을 취하게 할 수 있다.

앞서 기술한 충돌 문제를 방지하기 위해 여러 접근법이 사용되고 있다. 한가지 접근법에서, 프로세서를 순차적으로 작동시키거나 프로세서의 시간 공유를 행함으로서 충돌을 회피할 수 있다. 이 방식으로, 프로세서는 충돌 방지를 위해 공유 리소스 접근에 간단히 " 순서부여" 를 한다. 자주 사용되는 이러한 시스템은 " 링 통과" 나 " 토큰 시스템" 을 포함한다. 이 경우에, 사용자 그룹에 관한 링을 지나는 것과 유사하게 지정 순서에 따라 시스템에 의해 충돌 가능성이 있는 프로세서가 간단하게 추첨된다.

불행하게도, 순차적 프로세서 접근 방식의 이용은 전체 컴퓨터 시스템의 작동에 상당한 제한을 가한다. 경쟁 프로세서의 폴링(polling)시에 시스템에 의해 순차적 시간이 사용된다는 사실로부터 앞서의 제한점이 발생한다. 추가적으로, 단일 프로세서가 작동하고 공유 메모리 접근을 요청하는 경우, 예를 들어 시스템이 순서를 따라 진행되면 각각의 메모리 사이클을 따라 공유 리소스로의 프로세서 접근간 지연이 생성된다.

충돌 방지를 위한 또다른 기존 접근법은 컴퓨터 시스템의 프로세서 사이에 우선순위를 제시하는 것이다. 한가지 이러한 배열은 모든 시스템에 시스템 중요도에 따라 우선순위를 할당한다. 메모리 제어기는 충돌이 발생할 때마다 가장 높은 우선순위의 프로세서에 접근을 제공한다. 가령, 두 개의 프로세서를 가지는 시스템에서, 저장 데이터 관리를 위해 메모리의 주기적 리프레싱을 필요로하는 DRAM 메모리 장치인 공유 메모리에 제 1, 2 프로세서가 접근한다. 일반적으로, DRAM 메모리는 분리된 독립 리프레시 시스템에 의해 리프레싱된다. 이러한 멀티프로세서 시스템에서, 프로세서와 리프레시 시스템은 공통 메모리에 접근하고자 경쟁한다. 프로세서와 리프레시 시스템에 할당된 여러 우선순위에 따라, 시스템 메모리 제어기는 메모리 접근 요청 충돌이나 명령을 처리할 것이다. 이러한 시스템이 충돌을 해결하고, 순수한 순차적 충돌 방지 시스템에 비해 보다 효율적인지만, 이는 또한 유동성 부족의 문제를 가진다.

충돌 해결을 위한 또한가지 접근법은 메모리 제어기에 통합된 결정 능력을 포함한다. 불행하게도, 메모리 제어기의 결정부는 클럭 시스템의 타이밍과 제어하에서 작동되기 때문에, 메모리 제어기가 공통 메모리로의 접근을 허락할 수 있기 전에 실제 결정 과정을 실행하기 위해 실질적인 시간이 사용된다는 점에서 문제가 생긴다.

불행하게도, 실제 결정 과정의 실행에서의 이러한 문제는 다중 뱅크형 메모리 시스템으로의 접근을 허락하는 기존 메모리 제어기의 능력을 손상시킨다. 다중 뱅크형 메모리 시스템에서, 실제 메모리 핵은 특정 영역이나 뱅크로 나뉘어지고, 여기에 검색될 데이터가 저장된다. 더 빠르고 효율적인 메모리 접근을 제공함에도 불구하고, 다중 뱅크 메모리 장치를 처리하는 기존 메모리 제어기에 요구되는 복잡도는 시스템의 전체 접근 시간을 총체적으로 느리게한다.

앞서의 관점에서, 메모리 제어기에 의해 멀티뱅크형 메모리에 포함되는 메모리 페이지의 메모리 접근 속도를 향상시키는 방법이 요구된다.

발명의 상세한 설명

본 발명에 따라, 메모리 제어기에 의해 멀티뱅크형 메모리에 포함되는 메모리 페이지의 메모리 접근 속도를 향상시키는 방법이 공개된다. 한 개의 실시예에서, 메모리 제어기는 다수의 페이지 레지스터를 포함하고, 각각의 페이지 레지스터는 다수의 메모리 뱅크 중 한 개와 연계된다. 이때 특정 뱅크와 연계된 페이지 레지스터는 특정 뱅크 번호, 특정 뱅크 번호 내에 위치하는 열린 페이지 주소, 그리고 열린 페이지 상태를 저장하도록 배열된다. 메모리 제어기는 다수의 페이지 레지스터 각각에 연결된 조절가능한 비교기 유닛을 또한 포함한다. 유입 시스템 어드레스 요청은 요청된 뱅크 번호와 요청된 페이지 번호를 포함하는 메모리 제어기에 의해 수신된다. 요청된 뱅크 번호에 상응하는 페이지 레지스터는 비교기에 의해 위치하며, 위치한 페이지 레지스터에 포함된 저장된 페이지 어드레스는 요청된 페이지 어드레스와 비교된다. 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치할 때, 요청된 뱅크 번호에 상응하는 메모리 뱅크의 요청 페이지에 접근이 이루어진다.

도면의 간단한 설명

도 1A는 발명의 한 실시예에 따르는 범용 제어기의 폭넓은 구현을 나타내는 도면.

도 1B는 도 1A에 도시되는 범용 제어기의 특정 구현을 나타내는 도면.

도 1C는 발명의 한 실시예에 따르는 범용 제어기에 연결되는 어드레스 공간 제어기의 도면.

도 1D는 도 1C에 도시되는 어드레스 공간 제어기의 특정 구현의 도면.

도 1E는 발명의 한 실시예에 따르는 일례의 요청/응답 ID 번호를 도시하는 도면.

도 2A는 발명의 한 실시예에 따르는 범용 명령의 도면.

도 2B는 메모리 페이지 판독 명령을 요청하기에 적절한 도 2A에 도시되는 종류의 전용 명령의 도면.

도 2C는 도 2B에 도시되는 일례의 명령의 명령 성분 사이에 적절한 타이밍 구간을 제공함으로서 형성되는 순서 명령의 예의 도면.

도 3은 발명의 한 실시예에 따르는 리소스 태그의 도면.

도 4는 발명의 한 실시예에 따라 범용 제어기가 공유 리소스에 접근하는 과정의 상세한 순서도.

도 5는 발명의 한 실시예에 따라 실행되는 작동의 순서와 리소스 상태를 범용 제어기가 결정하는 처리 과정의 도면.

도 6은 발명의 한 실시예에 따르는 처리 과정에 따라 작동 순서 사이에 적절한 타이밍을 범용 제어기가 결정하는 처리 과정의 도면.

도 7A와 7B는 발명의 한 실시예에 따르는 페이지 히트/미스(hit/miss) 제어기의 도면.

도 8은 발명의 한 실시예에 따르는 뱅크 접근 제어기의 도면.

도 9A는 발명의 한 실시예에 따르는 SLDRAM 기반의 멀티프로세서 시스템의 도면.

도 9B는 도 9A에 도시되는 멀티프로세서 시스템에 따라 SLDRAM 버스 트랜잭션을 도시하는 타이밍도표.

도 10은 발명의 한 실시예에 따르는 메모리 제어기의 블록도표.

도 11은 발명의 한 실시예에 따르는 제한 블록의 블록도표.

도 12는 발명의 한 실시예에 따르는 SLDRAM 명령 타이밍도표.

도 13A-13C는 본 발명의 특정 실시예에 따라 메모리 명령의 재정렬을 나타내는 시간표.

도 14는 발명의 특정 실시예에 따라 고안된 메모리 제어기 일부의 블록도표.

도 15는 발명의 특정 실시예에 따라 고안된 재정렬 회로의 블록도표.

도 16는 도 15의 재정렬 회로의 상세한 블록도표.

도 17은 발명의 특정 실시예에 따른 명령 큐 요소의 내용의 다이어그램.

도 18은 어드레스 시프터의 특정 실시예의 블록도표.

도 19는 발명의 특정 실시예에 따른 데이터 큐 요소의 내용의 다이어그램.

도 20은 도 15에 도시되는 충돌 감지 시스템의 또다른 구현인 충돌 감지 시스템의 도면.

도 21은 타겟 장치로의 모든 판독/기록 명령이 데이터 패킷 트랜스퍼를 이와 연계시키는 방법을 도시하는 타이밍 다이어그램.

도 22는 특정 페이지에서의 마지막 명령과, 상기 메모리의 다음 예측된 접근 사이의 시간을 저장하는 N 페이지 타이머를 가지는 예측 시스템의 도면.

도 23은 발명의 한 실시예에 따라 장치 접근 우선순위화 장치를 가지는 장치 제어기의 도면.

도 4는 발명의 한 실시예에 따라 제한 블록에 의해 실행되는 스케쥴 과정을 요약한 표 4의 도면.

실시예

프로세서와 같은 여러 장치가 공통 리소스를 공유하는 시스템에서, 두 개 이상의 장치가 공유 리소스에 접근을 요청할 때 일반적으로 충돌을 방지하기 위해 여러 접근법이 사용되고 있다. 한 접근법에서, 프로세서를 순차적으로 작동시킴으로서, 또는 프로세서의 시간을 공유함으로서, 충돌을 회피할 수 있다. 이 방식으로, 프로세서는 충돌 방지를 위해 공유 리소스 접근에 " 순서부여" 를 한다. 흔히 사용되는 이러한 시스템은 " 링 통과" 나 " 토큰 시스템" 을 포함한다. 이 경우에, 사용자 그룹에 관한 링을 지나는 것과 유사하게 지정 순서에 따라 시스템에 의해 충돌 가능성이 있는 프로세서가 간단하게 추첨된다.

불행하게도, 이 순차적 접근 방식은 프로세서의 경쟁시 상당한 시간이 사용되기 때문에 전체 컴퓨터 시스템의 작동에 상당한 제한을 가한다.

충돌 방지를 위한 또다른 기존 접근법은 컴퓨터 시스템의 프로세서간 우선순위를 부여하는 과정에 의존한다. 이러한 배열은 시스템 중요도에 따라 모든 프로세서에 우선순위를 부여한다. 이러한 시스템이 충돌을 해결하고 순수한 순차적 충돌 해결 시스템에 비해 보다 효율적이지만, 이는 유동성 부족의 문제점을 가진다.

충돌 해결을 위한 또다른 기존 접근법은 제어기형태 장치에 통합되는 결정 논리 장치를 포함한다. 불행하게도, 제어기가 공유 메모리로의 접근을 허락받기 전에 실제 결정 과정 수행에 상당한 시간이 사용되어야 한다는 점을 결정 논리 장치의 복잡도가 필요로한다.

시스템 성능을 느리게하는 복잡한 논리 장치의 문제점은 각각 다른 작동 특성을 가지는 다수의 상호연결 메모리 장치 사이에 분산되는 메모리를 가지는 멀티-칩 모듈형 메모리 시스템에서 악화된다. 기존 논리 장치가 여러 메모리 장치에 내재하는 서로 다른 접근 특성을 각각 보상하도록 하는 구조를 가질 수 없기 때문에, 전체 시스템 성능이 저하된다.

넓은 범주에서 도 1A에 도시되는 바와 같이, 발명은 공유 리소스(108)에 접근을 제공하도록 시스템 버스(106)를 통해 범용 장치 제어기(104)에 각각 연결되는 요청 장치(102)를 가지는 시스템(100)을 들어 설명될 수 있다. 한 실시예에서, 시스템 버스(106)는 관련 시스템 인터페이스층(110)을 통해 범용 제어기(104)에 연결되지만, 범용 제어기(104)는 공유 리소스 인터페이스(109)를 통해 공유 리소스(108)에 연결된다. 넓은 범주에서, 범용 제어기(104)는 요청 시스템에 의해 발생된 공유 리소스 요청과, 공유 리소스 작동 특성 매개변수(113)를 바탕으로 공유 리소스(108)의 상태를 결정한다. 요청 시스템(102)이 나머지 프로세서에 의해 공유되는 메모리 장치(108)로 공유 리소스(108)로의 접근을 요청하는 멀티프로세서 시스템의 한 개의 프로세서인 경우에, 범용 제어기(104)는 필요한 리소스 요청을 완료하기 위해 실행될 연산의 순서를 결정한다. 메모리 장치(108)가 SDRAM일 경우, 일반적으로 작동은 프리차지(pre-charge), 페이지클로즈(page close), 페이지 오픈(page open), 페이지 판독, 또는 페이지 기록을 포함한다.

특정 순서의 작동이 결정될 경우, 범용 제어기(104)는 데이터 충돌이나 다른 형태의 충돌을 방지하기 위하여 작동 순서 사이에 적절한 타이밍을 결정한다. 선호되는 실시예에서, 타이밍은 탐색표 등에 저장되는 공유 메모리 장치의 작동 특성을 바탕으로 한다. 적절하게 순서화된 접근 명령은 범용 제어기에 의해 처리되고, 그후, 공유 메모리에 의해 응답된다.

본 발명의 실시예에서, 발명의 완전한 이해를 돕기 위해 수많은 구체적 실시예가 설명된다. 그러나, 이 특정 세부사항을 사용하지 않으면서 대안의 요소나 과정을 이용해도 본 발명이 구현될 수 있다. 또다른 예에서, 본 발명의 태양을 불필요하게 흐리지 않기 위해, 공지된 공정, 과정, 성분, 회로는 상세하게 설명되지 않는다.

발명은 프로세서와 공유 메모리 사이의 접촉으로 작용하도록 배열되는 메모리 제어기를 들어 이제부터 설명될 것이다. 그러나, 어떤 리소스(공유 방식이든, 아니든)로의 접근을 제어할 수 있는 범용 제어기로 발명이 구현될 수 있다는 것에 주목하여야 한다. 이러한 리소스가 반드시 메모리일 필요는 없다. 실제로, 버스 접근 호출을 감소시킴으로서 유효 시스템 버스 대역폭을 증가시키도록, 멀티프로세서 시스템의 트래픽 제어를 제공하는 것과 같이, 공유 시스템 버스로의 접근을 제어하기 위해 발명이 또한 사용될 수 있다.

도 1B에서, 시스템(100)은 시스템 버스(106)를 통해 범용 제어기(104)에 연결되는 프로세서와 같은 요청 장치(102)를 가진다. 제어기(104)는 DRAM, SDRAM, SLDRAM, EDO, FPM, RDRAM, 등과 같이 여러 형태를 취할 수 있은 메모리(108)와 같은 공유 리소스(108)에 연결된다. 기술되는 실시예에서, 시스템 버스(106)는 범용 프로세서(102)에 의해 발생된 메모리 어드레스 요청을 제어기(104)로 보내도록 배열되는 일방향 어드레스 버스(106-1)를 포함한다. 시스템 버스(106)는 어드레스 버스(106-1)와 연계하여 메모리 어드레스와 관련된 명령을 실행하는 일방향 명령 버스(106-2)를 또한 포함한다. 가령, 프로세서(102)가 메모리(108)의 특정 메모리 위치에 저장된 실행 명령을 필요로할 때, 프로세서는 어드레스 버스(106-1) 상에서 상응하는 메모리 어드레스 요청(시스템 어드레스로 불림)과 동시에 순서대로 판독 요청(시스템 명령이라고 불림)을 명령 버스(106-2)로 출력한다. 시스템 어드레스 및 시스템 명령은 제어기(104)에 포함된 컨피거러블(configurable) 시스템 인터페이스(110)에 의해 수신된다. " 컨피거러블" 이라는 용어는, 시스템 인터페이스(110)가 메모리(108)에 의해 요구되는 방식과 형태가 무엇이든 간에 수신된 시스템 명령 및 어드레스를 처리하도록 배열될 수 있다는 것을 의미한다. 이 방식으로, 프로세서(102)에 의해 요청된 데이터는, 각각의 메모리 장치에 대해 프로세서(102)가 주문 요청 발생을 받지 않으면서, 제어기(104)에 연결된 수많은 종류의 메모리 장치에 저장될 수 있다.

기술되는 실시예에서, 시스템 인터페이스(110)는 수신 시스템 명령과 시스템 어드레스를 범용 명령(200)으로 불리는 것으로 변환한다. 그 예가 도 2A에 도시된다. 한가지 구현에서, 공유 리소스가 DRAM형 메모리 장치(SLDRAM, SDRAM, EDO DRAM 등)일 때, 범용 명령(200)은 메모리(108)의 어떤 메모리 접근도 실행하기 위하여 요청된 모든 작동을 포함한 5개의 데이터 필드로 형성된다. 이러한 작동은 특정 행이 프리-차지되어야 하는 지를 표시하기 위해 사용되는 프리-차지 데이터 필드(202)로 표시되는 프리-차지 작동을 포함한다. 다른 작동은 활성화 데이터 필드(204), 판독 데이터 필드(206), 기록 데이터 필드(208), 리프레시 데이터 필드(210)를 포함한다. 메모리가 현재 활성화 상태(판독이나 기록후 열린 상태)인 메모리 뱅크 1의 메모리 페이지 1을 가지는 경우에, 이어지는 프로세서 명령은 메모리 뱅크 1의 페이지 2에 저장된 데이터가 프로세서(102)에서 판독되고 출력될 것을 요구한다. 이러한 경우에, 프로세서(102)에 의해 요청 명령을 실행하기 위해, 페이지 1은 닫혀져야하고, 페이지 2는 활성화되어야 하며, 활성화가 종료된 후, 페이지 2가 판독된다. 따라서, 도 2B에 도시되는 범용 명령(212)은 " 관련 연산 실행" 을 표시하기 위해 " 1" 로 설정되는 데이터 필드(202, 204, 206)와, " 관련 작동을 실행안함(NOP)" 을 표시하는 " 0" 으로 설정되는 데이터 필드(208, 210)를 가지도록, 범용 명령 발생기(110)에 의해 발생된다.

도 1B에서, 다수의 서로 다른 요청 장치가 메모리(108)로의 접근을 공유한다는 점에서 메모리(108)로의 접근이 동적이기 때문에, 메모리(108)의 상태는 일정하게 변화한다. 메모리 상태란, 특정 메모리 위치에서 특정 작동을 성공적으로 실행하기 위해 상기 메모리 위치의 상태가 공지되어야한다는 것을 의미한다. 가령, 특정 메모리 페이지가 닫힌 경우에, 판독 작동 실행을 위해, 상기 메모리 페이지가 반드시 열려야 한다. 따라서, 특정 어드레스 위치의 현 상태를 확인하기 위해, 특정 메모리 위치에서 실행된 가장 현재의 작동은 도 3에 도시되는 바와 같이 리소스 태그(300)로 식별된다. 발명의 한 실시예에서, 리소스 태그(300)는 특정 메모리 어드레스 위치 식별에 사용되는 어드레스 필드(302), 어드레스 필드(302)에서 식별된 어드레스에 대한 가장 최근의 명령 식별에 사용되는 최근 명령 필드(304), 그리고 최근 명령 데이터의 시간 필드(306)를 포함한다. 가령, 메모리 어드레스 ADD_5에 대한 리소스 태그(308)는 페이지 판독이 시간 5Φ에서 처리되었다는 것을 나타내며, 리소스 태그(310)는 동일한 메모리 어드레스 ADD_5의 경우에 ADD_5에서의 메모리 페이지 상에서 페이지 기록이 시간 10Φ에 실행될 것임을 나타낸다. 메모리 어드레스 ADD_5의 상태를 탐색함으로서, 범용 제어기(104)는 ADD_5에서의 메모리 페이지가 이미 열려 있고 따라서 페이지를 여는 작동이 필요하지 않음을 안다.

리소스 태그 버퍼(114)에 저장되는 태그(300)에 의해 제공되는 리소스 상태 정보를 이용하여, 컨피거러블 시스템 인터페이스(112)에 연결된 명령 순서기(114)는 명령 성분(202-204)(204-206) 사이의 타이밍 구간 t1과 t2를 각각 가지는 도 2C에 도시되는 순서 명령(220)을 제공하기 위해 범용 명령(200)의 명령 성분(2022-210) 사이의 적절한 타이밍 구간을 제공한다. 명령 성분(208, 210)이 "NOP" 형 필드이기 때문에, 순서 명령(220)은 이들 필드에 대한 어떤 기준도 포함하지 않으며, 이와 같이 t1+t2와 같은 시간 주기와, 성분(202-206)에 대해 필요한 클럭 사이클과 같은 시간 주기만을 필요로 한다. 이 방식으로, 명령 순서기(114)는 프로세서(102)와 메모리(108)간의 최적의 명령 및 데이터 흐름을 제공할 수 있다.

발명의 또다른 실시예에서, 공유 리소스(108)가 SDRAM과 같은 멀티뱅크형 메모리 장치일 때, 또는, 공유 리소스가 멀티칩 모듈과 같은 멀티소자 메모리 장치일 때, 리소스 태그 버퍼(114)는 특정 뱅크나 소자의 모든 열린 페이지에 대해 리소스 태그를 저장할 수 있다. 한가지 구현에서, 비교기(도시되지 않음)는 시스템 어드레스의 뱅크 번호나 장치 식별자를 감지하고, 페이지 어드레스 및 시스템 어드레스를 태그 버퍼(114)의 내용과 비교한다. 비교가 "히트(hit)"가 아닌 경우에(즉, 어드레스가 일치하지 않는 경우), 범용 제어기(104)는 태그 버퍼(114)로부터의 어드레스를 이용하여 구페이지를 닫아야하고, 새로운 시스템 명령에 따라 새 페이지를 열어야 한다.

범용 제어기(104)에 의해 서비스되는 다수의 서로 다른 장치가 존재하는 경우에, 유입 시스템 어드레스가 관련된 특정 장치와만 연계된 작동 매개변수를 선택할 수 있는 것이 바람직하다. 범용 제어기가 다수의 서로 다른 장치를 서비스할 경우, 범용 제어기(104)에 연결되는 어드레스 공간 제어기(120)는 도 1C에서 도시된다. 기술되는 실시예에서, 어드레스 공간 제어기(120)는 유입 시스템 어드레스와 관련된 한 개의 장치에 대한 장치의 특정 매개변수를 선택할 수 있는 능력을 제공한다. 도 1D에 도시되는 특정 구현에서, 어드레스 공간 제어기(120)는 유입 어드레스가 어느 장치와 관련된 지를 식별하는 영역 어드레스 범위 버퍼(124)에 관한 내용과 유입 시스템 어드레스를 비교하도록 배열되는 비교기(122)를 포함한다. 특정 장치나 영역이 식별되면, 장치 매개변수 레지스터(126, 127) 그룹 중 하나가 선택된다. 그 각각은 범위 버퍼(124)에 연결되고, 특정 장치에 대한 장치의 매개변수를 포함한다. 선택된 장치 매개변수 레지스터는 시스템 어드레스에 상응하는 장치와 관련된 특정 작동 매개변수를 제공한다. 몇몇 실시예에서, 선택된 장치 매개변수 레지스터의 내용은 LUT(118)로 입력된다. 이 방식으로, 다수의 서로 다른 장치가 범용 제어기(14)에 의해 서비스되어, 각 장치의 특정 작동 매개변수가 식별되고, 상응하는 범용 명령을 최적으로 순서화하기 위해 사용된다.

범용 제어기에 연결되는 장치 중 하나가 바빠서 새로운 명령을 수용할 수 없는 일부 경우에, 명령 큐에서 대기하는 그외 다른 명령을 선택할 수 있는 것이 바람직할 것이다. 발명의 일부 실시예에서, 범용 제어기에 의한 요청과 장치에 의한 모든 응답은 도 1E에 도시되는 바와 같이 5비트의 데이터 워드 길이인 관련 ID 번호(150)를 가진다. ID 번호(150)는

3비트 길이의 요청 번호 필드(153)와 2비트 길이의 그룹 선택기 필드(152)를 포함한다. 그룹 선택기(GS)는 특정 시스템 요청이 어느 그룹에 속하는 지를 결정한다. 이때 요청 번호(RN)는 동일한 송수신기로부터의 일련의 요청이 일련의 요청 번호를 가지도록 그룹 선택기 필드(152)에 의해 식별되는 관련 그룹과의 요청수나 응답수를 나타낸다.

일부 실시예에서, 그룹 우선순위 선택기 레지스터(154)는 높은 우선순위를 가지는 요청이나 응답 그룹이 낮은 우선순위에 비해 선행하도록 각각의 요청 그룹이나 응답 그룹에 대해 우선순위값을 가진다. 이 방식으로, 높은 우선순위의 응답이나 요청은 낮은 우선도의 요청이나 응답이 다음 클럭 사이클에서 처리될 수 없을 때 낮은 우선순위의 경우를 회피할 수 있다. 지속로크(livelock)로 명명되는 것을 방지하기 위해, 지속 로크 카운터 레지스터(156)는 높은 우선순위를 가지는 일련의 요청(또는 응답)의 수에 관한 정보를 포함하며, 상기 우선순위를 가지는 일련의 요청(또는 응답)은 낮은 우선순위의 요청(또는 응답)을 회피할 수 있다. 이 방식으로, 낮은 우선순위의 요청(또는 응답)은 클럭 사이클의 본질적 수를 갈망할 수 없다.

명령 및 데이터 흐름의 제어를 최적화하기 위하여, 각각의 공유 리소스가 작동 특성 세트를 관련시킨다는 사실이 인지된다. 두 개 이상의 공유 리소스가 범용 제어기(104)에 의해 서비스되는 경우에, 각각의 공유 리소스는 명령 순서기(116)에 연결되는 탐색표(LUT)(118)에 저장되는 다른 세트의 작동 특성을 가진다. 명령 순서기(116)는 순서 명령(220) 형성을 위해 명령 성분(202-210)을 적절히 순서화하는 리소스 태그 버퍼(114)에 저장된 리소스 태그와 연계하여 LUT(118)에 의해 제공되는 정보를 이용한다. 이는 공유 리소스가 멀티 칩 모듈과 같이 메모리 장치의 그룹일 경우 특히 사실이고, 각각의 장치는 서로 다른 작동 특성을 가질 수 있다.

도 4에서, 발명의 실시예에 따라 공유 리소스에 접근하는 범용 제어기에 대한 과정 400의 순서도가 도시된다. 과정 400은 공유 리소스에 대한 접근 명령을 시스템이 발생시킴으로서 시작된다(402). 공유 리소스가 DRAM 기반의 메모리 장치일 경우, 이러한 작동은 프리-차지, 리프레시, 닫힘, 열림, 판독, 기록을 포함한다. 가령, 프로세서는 요청 페이지가 저장된 메모리 위치를 표시하는 관련 시스템 어드레스와 시스템 명령(즉, 페이지 판독)을 발생시킴으로서 공유 메모리에 저장되는 메모리 페이지를 요청한다. 선호되는 실시예에서, 리소스 상태는 공유 메모리의 능동 메모리 위치와 연관된 리소스 태그를 이용하여 결정된다(404). 그 다음에, 공유 리소스의 필요 요청 실행을 위해 작동 순서의 결정이 이루어진다(406). 요구된 요청 실행에 필요한 작동 순서를 바탕으로 하는 범용 명령을 범용 제어기가 발생시킨다(408). 가령, 페이지 판독 작동 실행을 위해, 이전에 열린 페이지가 닫혀져야 하고, 새 페이지가 활성화되고 판독 작동이 실행되어야 하며, 이 모두는 단일 범용 명령 구조로 해석된다. 범용 명령이 범용 제어기에 의해 만들어진 경우, 공유 리소스에 대한 특정 작동 특성 데이터와 리소스 태그를 이용하여, 범용 제어기는 범용 명령의 여러 명령 성분 사이에서 적절한 타이밍을 결정한다(410). 순서 명령은 일부 실시예에서 공유 리소스로의 물리적 단계를 이용하여 처리된다(412). 마지막으로, 시스템 어드레스로 표시되는 위치에 저장된 데이터를 제공함으로서 공유 리소스가 순서화 명령에 응답한다(414).

발명의 한 실시예에서, 범용 제어기는 도 5에 도시되는 과정 500을 이용하여 리소스 상태(402)와 실행될 작동 순서(404)를 결정한다. 과정 500은 리소스 파티션 식별자(즉, 메모리 어드레스 레지스터)를 리소스 식별자(즉, 리소스 태그 어드레스 필드(202))와 비교함으로서 시작된다(502). " 히트" 가 발생한다고 결정될 경우(504), 다음 명령이 처리된다(506). 이와 달리, 새 명령의 어드레스가 현재 태그 어드레스 필드와 일치하지 않을 경우(즉, 히트가 아닐 경우), 구 페이지가 열렸는 지에 관한 결정이 이루어진다(508). 구페이지가 열렸으면, 구페이지는 닫히고(510) 새페이지가 열린다(512). 그러나, 구페이지가 열리지 않으면(508), 새 페이지가 열리고(512), 어느 경우에도, 새 페이지가 열리면, 다음 명령(데이터 작동)이 처리된다(506).

발명의 한 실시예에서, 범용 제어기는 도 6에 도시되는 과정 600에 따라 작동 순서(410) 사이의 적절한 타이밍을 결정한다. 과정 600은 특정 리소스에 대해, 명령의 새 순서의 첫 번째 명령을, 가장 최근의 이전 명령 순서의 최근 명령과 LQRY하는 범용 제어기에 의해 시작된다(602). 범용 제어기는 가장 최근의 이전 범용 명령의 최근 명령 성분과 새 범용 명령의 제 1 명령 성분을 비교함으로서 범용 명령 성분간의 시간 제약을 결정한다(604). 다른 실시예에서, 범용 제어기는 어레이의 제 1 행이 구명령(가장 최근의 이전 명령)을 나타내고 제 1 열이 새 명령을 나타내는 표 1에 도시되는 바와 같이 2차원 어레이 형태의 2지표 탐색표(LUT)를 이용한다. 가령, 표 1에서, 구명령이 판독된 페이지고 새 명령이 닫힌 페이지이면, 닫힌 새 명령 페이지와 판독된 새명령 페이지의 교차(즉, 5Φ)는 두 작동간 최소의 가능한 시간양(즉, 최소의 물리적 처리 시간)을 제공한다. 일반적으로, LUT에 저장된 정보는 공유 리소스 제작자에 의해 제공된다.

표 1

구명령

새명령		닫힌페이지	열린페이지	판독	기록
	닫힌페이지			5Φ	
	열린페이지				
	판독				
	기록				

리소스의 물리적 제약이 특정 범용 명령 성분에 대해 결정되면, 범용 명령에 추가 명령 성분이 포함되는 지가 결정된다(606). 추가 명령 성분이 없을 경우, 범용 명령 및 그 관련 성분 타이밍 설명이 저장된다(608). 다른 한편, 범용 명령에 추가 명령 성분이 포함될 경우, 제어는 단계 604로 복귀하여, 상기 성분에 대한 상응하는 물리적 타이밍 제약이 결정된다.

그러나, 다수의 메모리 뱅크를 가지는 공유 메모리(108)의 물리적 페이지의 상태 탐색을 위해, 다수의 캐시 메모리를 필요로하는 다수의 리소스 태그가 리소스 태그 버퍼(114)에 제공된다. 이는 범용 제어기(104)의 성능의 속도를 저하시킨다. 왜냐하면, 서로 다른 위치에 각각 존재할 수 있은 메모리의 특정 페이지에 대한 특정 리소스 태그 검색을 위해 상당한 시간이 필요하기 때문이다. 도 7A의 한 실시예에서, 페이지 히트/미스 제어기(702)가 범용 제어기(104)에 포함되어, 멀티뱅크 메모리(706)의 메모리 뱅크의 수 N보다 작은 M개의 페이지 레지스터(704)를 감소시키도록 배열된다. 왜냐하면, 모든 뱅크가 M 페이지 레지스터(704)에 나타나지는 않기 때문이다. 작동 중에, 각각의 M 페이지 레지스터(704)는 열린 페이지의 어드레스 및 상태 데이터를 저장하고, 임의 페이지 레지스터 번호 발생기(708)는 열린 페이지의 상태로 대체되어야하는 페이지 레지스터에 상응하는 M 이하의 임의 정수값을 발생시킨다. 비교기(710)는 다음의 네 개의 가능한 결과와 함께 모든 M 레지스터의 페이지 어드레스와 뱅크 번호를 유입 시스템 어드레스와 비교한다.

1) 비교기(710)가 히트를 표시하면, 요청 뱅크의 필요한 페이지는 열리고 접근 가능하다.

2) 뱅크 히트가 있고 페이지 미스가 있다고 비교기(710)가 표시하면, 범용 제어기(104)는 페이지 레지스터로부터의 페이지 어드레스를 이용하여 구페이지를 닫아야하고, 시스템 어드레스로부터의 페이지 어드레스를 이용하여 새 페이지를 열어야 한다.

3) 비교기(710)가 뱅크 및 페이지 미스를 나타내면, 범용 제어기(104)는 임의 페이지 번호 발생기에 의해 주어지는 번호를 가지는 뱅크의 구페이지를 반드시 닫아야하고, 시스템 어드레스를 이용하여 새 페이지를 열어야하며, 마지막으로 요청된 뱅크에 접근한다.

4) 뱅크와 페이지 미스, 그러나 한 개 이상의 페이지 레지스터가 사용되지 않으면, 이 레지스터가 사용될 것이고, 새 페이지는 열릴 것이다.

일부 실시예에서, 임의 번호 페이지 발생기(708)는 M 레지스터(704) 중 어느 것이 가장 오랜 시간 동안 사용되지 않았는 지를 결정하기 위해 도 7B에 도시되는 바와 같이 " 최근에 거의 사용되지 않은(LRU)" 비교기(712)로 대체된다.

멀티뱅크 메모리(704)의 물리적 페이지의 상태 탐색에 추가하여, 도 8에 도시되는 뱅크 접근 제어기(800)는 멀티뱅크 메모리(704)에 포함된 메모리 뱅크의 수 N에 상응하는 N 뱅크 레지스터(802)를 포함한다. 뱅크 레지스터(802)는 관련 뱅크의 정보가 저장된 뱅크의 식별 번호를 표시하는 뱅크 번호 필드(804)를 포함한다. 뱅크 레지스터(802)는 뱅크 번호 필드(804)의 뱅크 번호로 식별되는 특정 뱅크의 상태를 표시하는 뱅크 상태 필드(804)를 또한 포함한다. 특정 실시예에서, 뱅크 상태 필드(806)는 표 2에 제시되는 것과 같은 값을 취할 수 있다.

표 2

뱅크 레지스터 요소	설명
뱅크 번호	뱅크 레지스터의 정보가 저장된 뱅크를 식별함
뱅크 상태	뱅크상태를 표시함" 00" - 무효 엔트리" 01" - 뱅크 카운터값이 0에 이를때까지 감소함" 10" - 뱅크가 닫힘" 11" - 뱅크 카운터값이 0에 이를때까지 감소함 뱅크 카운터가 0보다 크면, 메모리의 모든 뱅크로의 접근이 금지됨
뱅크 타이머	뱅크 카운터가 0보다 크면, 뱅크상태값에 따른 메모리로의 접근이 금지됨

400-800Mb/s/pin 범위의 버스 데이터 속도를 운반하는 동기식 링크 DRAM(SLDRAM)와 같은 고속 패킷 지향 메모리의 개발로, 메모리 접근시 충돌에 의한 문제가 크게 증가되었다. 도 9A에서, SLDRAM을 기반으로 하는 멀티프로세서 시스템(900)은 시스템 버스(906)를 이용하여 제어기(904)에 연결되는 프로세서(902)를 포함한다. 범용 제어기(904)는 일방향 명령 버스(912)와 양방향 데이터 버스(914)로 구성되는 SLDRAM 버스를 통해 동기식 링크 DRAM(SLDRAM)(908)과 SLDRAM(910)에 연결된다. 단 두 개의 SLDRAM만이 도 9A에 도시되었으나, 버스(912, 914)를 통해 범용 제어기(904)에 여러개의 SLDRAM이 연결될 수 있다. 일부 경우에, SLDRAM은 SLDRAM(908)과 같이 적절한 수의 SLDRAM을 포함하는 버퍼 모듈의 형태를 취할 수 있다. 각각의 SLDRAM(908)에 범용 제어기(904)를 연결하는 초기화/동기화(I/S) 버스(916)는 범용 제어기(904)에 의해 발생되는 동기화 신호 및 초기화 신호의 신호 경로를 제공한다.

발명의 한 실시예에서, 패킷형 명령, 어드레스, 범용 제어기(904)로부터의 제어 정보는 명령 버스(912) 상에서 SLDRAM(908, 910)으로 선택적으로 보내진다. 데이터 버스(914)는 범용 제어기(904)로부터 SLDRAM(908, 910) 중 선택된 SLDRAM까지 패킷형 기록 데이터를 전송하도록 배열된다. 이의 대안으로, 데이터 버스(914)는 SLDRAM(908, 910) 중 선택된 SLDRAM으로부터 범용 제어기(904)까지 패킷형 판독 데이터를 전송하도록 또한 배열된다. 명령 버스(912)와 데이터 버스(914)는 같은 속도로 작동하는 것이 일반적이다. 400MB/s/p, 600MB/s/p, 800MB/s/p 등등이 그 예이다.

범용 제어기(904)에 의해 발생되고 명령 버스(912)에 의해 실행되는 다수의 제어 신호는 가령, 차이로부터 자유로운 구동 클럭 신호(CCLK), FLAG 신호, 명령 어드레스 신호 CA, LISTEN 신호, LINKON 신호, RESET 신호를 포함한다. 일반적으로, 패킷 명령은 명령의 제 1 워드가 FLAG 신호의 제 1 비트의 '1'로 표시되는 4개의 10-비트 워드로 형성된다. 선호되는 실시예에서, 차이로부터 자유로운 구동 클럭 CCLK의 두 변부는 명령 워드 래치를 위해 SLDRAM(908, 910)에 의해 사용된다. SLDRAM(908, 910)은 명령 유입을 위한 명령 버스(912)를 감시함으로서 HIGH인 LISTEN 신호에 응답한다. 또는, SLDRAM(908, 910)은 전력 절약 대기 모드에 진입함으로서 LOW인 LISTEN 신호에 응답한다. LINKON 신호와 RESET 신호는 SDLRAM(908, 910) 중 공지된 상태의 선택된 SLDRAM까지 각각 전원 오프나 온하기 위해 사용된다.

이 설명의 나머지 부분에서, SLDRAM(908)은 어떤 수의 SLDRAM도 범용 제어기(904)에 적절하게 연결될 수 있다는 전제하에서 기술될 것이다. 앞서 언급한 바와 같이, SLDRAM(904)와 같은 전형적인 SLDRAM 장치는 메모리 뱅크, 열, 행, 비트, 메모리 영역내부에 의해 계급을 가지는 형태로 구성된다. 각각의 계급 수준이 서로 다른 작동 특성을 가지는 것으로 관측될 수 있다는 사실이 중요하다. 이러한 작동 특성은 메모리 접근 시간, 칩 작동 시간, 데이터 검색 시간 등과 같은 매개변수를 포함하지만 이러한 사항만으로 제한되는 것은 아니다. 멀티뱅크 메모리 내의 뱅크가 동일한 작동 특성을 가질 것이지만, 서로 다른 명령 및 데이터 호출을 각각 가지는 서로 다른 메모리 형태나 서로 다른 메모리 그룹과 같이 영역이 다른 장치로 정해진다. 가령, 국부 메모리 그룹은 보드에 위치하는 제 2 비국부 메모리 그룹과 메모리 제어기에 직접 연결될 수 있다. 이때 상기 보드에서는, 드라이버 간섭이 국부 메모리 그룹에 대해 명령 및 데이터 호출을 증가시킨다. 다른 경우에, 멀티칩 모듈을 형성하고자 하는 여러 메모리 칩 각각은 다른 메모리 영역이라고 간주될 수 있다.

도 9A의 시스템에서 특히, SLDRAM(908)은 4 메모리 칩 A, B, C, d를 가지는 멀티칩 모듈이고, 각각의 메모리 칩은 명령 버스(912), 데이터 버스(914), I/S 버스(916)에 의해 개별적으로 접근될 수 있다. 메모리 칩 A-D 각각기 서로 다른 작동 특성을 가질 수 있기 때문에, 명령 및 데이터 패킷을 최적으로 스케쥴링하기 위하여, 이에 따라 범용 제어기(904)는 특정 계급 구조나 메모리 영역의 작동 특성을 이용할 수 있다.

한 예로, 도 9B는 도 9에 도시되는 멀티프로세서 시스템(900)에 따른 SLDRAM 버스 트랜잭션에 대한 대표적 타이밍 다이어그램을 도시한다. 작동 중, 프로세서는 SLDRAM(908)의 적절한 메모리 뱅크가 응답하는 판독 명령(950)과 기록 명령(952)과같은 프로세서 명령 패킷을 발생시킬 것이다. 일반적으로, 판독 명령(950) 및 기록 명령(952)은 프로세서의 특정 요청을 바탕으로 시스템 버스(906) 상에서 파이프라인 처리되고, 이로부터 명령(950, 952)이 발생하지만, 최적의 SLDRAM 성능에 대한 것은 아니다. 시스템 클럭 CLKsys(도시되지 않음)는 필요한 타이밍 신호를 제공한다.

본 예의 경우에, 프로세서(902a)는 SLDRAM(908)의 메모리 칩 A에 위치하는 메모리 어드레스 MA1을 가지는 판독 명령(950)을 발생시킨다. 프로세서(902b)는 SLDRAM(908)의 메모리 칩 A에 또한 위치하는 메모리 어드레스 MA2를 가지는 기록 명령(952)을 발생시킨다. 본 예에서, 판독 명령(950)은 기록 명령(952)의 출력 이전에 시스템 버스(906)에 출력된다. 범용 제어기(904)는 먼저 판독 명령(950)을 수신하고, 범용 제어기(904) 내에 저장된 수신지 어드레스 정보를 이용하여 명령 자체 및 명령 어드레스 MA1을 바탕으로 상기 명령을 처리한다. 최소의 처리 시간이 결정되면, 범용 제어기(904)는 수신된 프로세서 명령(950)에 상응하는 SLDRAM 명령 패킷 " 판독" (960)을 발생시키고, 이를 명령 버스(912)로 보낸다.

일반적으로, SLDRAM 명령 패킷은 8개의 뱅크, 1024 행 어드레스, 128 열 어드레스로 64M SLDRAM을 나타내는 표 3에 도시되는 바와 같은 네 개의 10 비트 워드로 구성된다. 도시되는 바와 같이, 뱅크 어드레스(BNK)에 대해 3비트, 행 어드레스(ROW)에 대해 10비트, 열 어드레스(COL)에 대해 7비트가 존재한다. 여러 다른 조직 및 밀도가 가능하고, 기술되는 40비트 포맷 내에서, 그리고 적절하다고 결정될 수 있은 그외 다른 포맷 내에서 수용될 수 있다. 전력이 온상태일 때, 범용 제어기(904)는 뱅크의 수, 행, 열, 관련 작동 특성과 같은 인자에 대한 SLDRAM 폴링을 바탕으로 명령 패킷을 구성한다. 그후 명령 패킷은 범용 제어기(904)에 의해 저장된다.

명령 패킷의 제 1 워드는 칩 ID 비트를 포함한다. SLDRAM은 국부 ID와 일치하지 않는 어떤 명령도 무시한다. 칩 ID는 초기화 및 동기화 신호를 이용하여 전력-온 중 범용 제어기(904)에 의해 할당된다. 이 방식으로, 범용 제어기(904)는 분리된 칩 작동 신호나 글루 논리 장치(glue logic)를 발생시키면서, 멀티프로세서 시스템(900)의 각각의 SLDRAM을 독자적으로 처리한다.

표 3

SLDRAM 명령 패킷 구조SLDRAM 명령 패킷 구조

FLAG	CA9	CA8	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0
1	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	CMD5
0	CMD4	CMD3	CMD2	CMD1	CMD0	BNK2	BNK1	BNK0	RW9	RW8
0	ROW7	ROW6	ROW5	ROW4	ROW3	ROW2	ROW1	ROW0	0	0
0	0	0	0	COL6	COL5	COL4	COL3	COL2	COL1	COL0

판독 명령(950) 및 기록 명령(952)이 파이프라인식으로 처리되기 때문에, 범용 제어기(904)는 판독 명령(950)의 수신 후 얼마의 시간동안 기록 명령(952)을 수신하고, 기록 명령(952)에 상응하는 SLDRAM 명령 패킷 " 기록" (962)을 순차적으로 처리한다. 동일한 뱅크(A)가 두 명령(960, 962)에 의해 처리되고 있기 때문에 이전에 처리된 판독 명령(960)과의 간섭을 방지하기 위해 기록(962)에 대한 데이터 오프셋과 최소 처리 시간을 발생시키도록 범용 제어기(904)는 판독 명령(960)의 처리 시간과, MA2 특성화 데이터를 이용한다.

이 방식으로, 범용 제어기(904)는 명령 및 데이터 패킷 스트림의 현 상태와 특정 수신지 어드레스 장치 작동 특성을 바탕으로 SLDRAM 명령 패킷의 처리를 동적으로 스케쥴링할 수 있다.

도 10은 발명의 한 실시예에 따르는 메모리 제어기의 블록 도표이다. 메모리 제어기(1000)는 도 1에 도시되는 범용 제어기의 한가지 가능한 실시예일 뿐, 발명의 범위를 제한하는 것으로 간주되어서는 안된다. 메모리 제어기(1000)는 시스템 버스(906)를 이용하여 프로세서(902)를 메모리 스케쥴러(1006)에 연결하는 시스템 인터페이스(1002)를 포함한다. 발명의 한 실시예에서, 시스템 인터페이스(1002)는 메모리 명령 패킷과, 프로세서(902)에 의해 발생되는 관련 기록 데이터 패킷을 메모리 명령 패킷 스케쥴러(1004)에 전송하는 과정을 제공한다. 모든 내부 버퍼가 차있고 새 명령이 수용될 수 없음을 스케쥴러(1006)가 표시하는 상황에서, 시스템 인터페이스(1002)는 시간 스케쥴러(1006)가 새 명령 수용 준비가 됨을 표시할 때까지 새 명령을 유지한다.

동기화 링크 매체 접근 제어기(SLiMAC)(1008)는 스케쥴러(1006)와 SLDRAM(908) 사이에 물리적 인터페이스를 제공한다. 특히, SLiMAC(1008)는 명령 버스(912)와 데이터 버스(914)를 각각 이용하여 SLiMAC(1008)를 SLDRAM(908)에 연결하는 명령 인터페이스(1010)와 데이터 인터페이스(1012)를 포함한다. 발명의 선호되는 실시예에서, 명령 인터페이스(1010)는 SLiMAC(1008)로부터 SLDRAM(908)까지 관련 명령 클럭 CCLK와 동반하여 메모리 명령을 전송한다. 일부 실시예에서, SLiMAC(1008)은 200MHz에서 구동되는 명령 클럭 신호 CCLK를 발생시키기 위해 인터페이스 클럭 신호 ICLK(100MHz에서 구동될 수 있음)를 이용하는 클럭 더블러를 병합한다.

발명의 한 실시예에서, 데이터 인터페이스(1012)는 데이터 버스(914) 상에서 데이터를 수신하고 송신한다. 데이터 버스(914)의 폭은 요구되는 SLDRAM의 수를 지원하기에 필요한만큼 클 수 있다. 따라서 필요한 대역폭 제공을 위해, 필요한 수의 데이터 인터페이스가 SLiMAC(1008)에 포함될 수 있다. 예를 들어, 데이터 버스(914)가 32비트 폭일 경우(가령 SLDRAM 당 16비트), SLIiMAC(1008)는 특정 SLDRAM과 연관된 16비트를 각각 조작할 수 있는 2개의 데이

터 인터페이스를 포함한다. 이 방식으로, SLiMAC(1008)에 포함된 데이터 인터페이스의 크기는 연결된 SLDRAM의 특정 구조와 가깝게 부합될 수 있다.

명령 인터페이스(1010)와 상당히 비슷한 방식으로, SLiMAC(1008)는 SLDRAM(908)으로부터 SLiMAC(1008)까지 전송되는 판독 데이터를 동반하는 데이터 클럭 신호 DCLK를 제공할 수 있다. 발명의 한 실시예에서, 대략 100MHz에서 1000MHz까지 인터페이스 클럭 ICLK 주사수를 두배화하는 클럭 더블러를 이용함으로서, 데이터 클럭 DCLK가 발생된다. 인터페이스 클럭 신호 ICLK, 명령 클럭 신호 CCLK, 데이터 클럭 신호 CDLK는 모두 위상 동기적이다.

발명의 선호되는 실시예에서, 스케쥴러(1006)는 시스템 인터페이스(1002)로부터 시스템 명령과 관련 시스템 어드레스 데이터를 수신하도록 배열되는 수신 블록(1016)을 포함한다. 수신 블록(1016)은 SLDRAM 명령 패킷 데이터와 관련 타이밍 정보를 재정렬 블록(1018)에 제공한다. 기록 버퍼(1020)는 시스템 인터페이스(1002)로부터 기록 데이터를 수신한다. 스케쥴러(1006)에 의해 설정되는 바와 같이, 판독 데이터는 데이터 인터페이스(1012)로부터, 데이터 버스(914)에 연결되는 판독 버퍼(1022)를 통해 전송되고, 판독 데이터를 시스템 인터페이스(1002)에 제공하도록 배열된다. 초기화/동기화(I/S) 버스(916)에 연결되는 I/S 블록(1024)은 적절한 초기화/동기화 신호를 SLDRAM(908)에 제공한다.

작동 중에, 스케쥴러(1006)는 프로세서(902)에 의해 발생되는 파이프라인식 메모리 명령 패킷을 수신한다. 일반적으로, 메모리 명령 패킷은 메모리 명령과 관련 메모리 어드레스로 이루어진다. 발명의 한 실시예에서, 스케쥴러(1006)는 메모리 명령 및 관련 데이터 패킷이 향하는 수신지 어드레스를 결정하기 위해 수신된 새 명령과 연관된 메모리 어드레스를 해독한다. 해독되며, 스케쥴러(1006)는 새 SLDRAM 명령 패킷을 처리하기 위해 방금 처리된 메모리 명령과 연관된 정보와, 저장된 수신지 어드레스 특정 장치 특성화 데이터를 이용한다. 새로운 SLDRAM 명령 패킷은 명령 버스(912)에 출력되고, 결국에는, SLDRAM 명령 패킷에 포함된 CHIP ID에 의해 식별되는 SLDRAM에 출력된다.

스케쥴링 과정의 일부로, 스케쥴러(1006)는 새 명령 처리 이전에 필요한 방금 처리된 명령의 처리 후 최소의 시간을 결정한다. 앞서 언급한 바와 같이 SLDRAM의 메모리 뱅크와 같은 각각의 계급 레벨이 서로 다른 작동 특성을 가질 수 있기 때문에, 스케쥴러(1006)는 초기화 중 서비스하는 각각의 SLDRAM을 폴링한다. 일부 실시예에서, 메모리 특정 매개변수(타이밍 등)는 연결된 메모리 장치가 작동 특성 결정을 위해 폴링을 허락하지 않을 경우 제한 블록 레지스터(1016)로 직접 기록될 수 있다. SLDRAM이 폴링되면, 스케쥴러(1006)는 적절한 스케쥴링 프로토콜 개발에 사용할 장치 특정 정보를 저장한다. 이 방식으로, 스케쥴러(1006)는 하드기록이나 다른 시간을 소모하거나 비용을 발생시키는 과정에 의지하지 않으면서 여러 종류와 수의 SLDRAM에 스케쥴링 서비스를 제공할 수 있다.

도 11은 발명의 한 실시예에 따르는 제한 블록(1100)을 도시한다. 제한 블록(1100)은 도 10에 도시되는 제한 블록(1016)의 한가지 가능한 실시예에 지나지 않는다. 제한 블록(1100)은 프로세서(902)에 의해 발생된 새 메모리 명령과 연계하여 수신된 새 어드레스 신호를 해독하도록 배열되는 시스템 인터페이스(102)에 연결되는 어드레스 디코더(1102)를 포함한다. 해독된 새 어드레스 신호는 모든(또는 일부의) 적절한 SLDRAM 메모리 뱅크에 대한 상태와 그외 다른 관련 정보가 저장되는 어레이 태그 레지스터(1104)에 입력을 제공한다. 어레이 태그 레지스터(1104)는 탐색표(LUT)(1108)에 해독된 새 명령 어드레스를 바탕으로 선택된 가상 뱅크에 대한 관련 데이터를 통과시키는 선택기(1106)에 입력을 제공한다.

제한 블록(1100)은 새 명령 어드레스가 위치하는 메모리 영역을 나타내는 영역 식별자 제공을 위해 수신된 새 어드레

스 신호를 사용하도록 배열되는 시스템 인터페이스(1002)에 또한 연결되는 영역 비교기(1110)를 또한 포함한다. 이 방식으로, 제한 블록(1100)은 메모리 영역 특정 특성화 데이터의 일부를 바탕으로 새 메모리 명령에 대한 최적의 스케쥴링 프로토콜을 제공할 수 있다. 영역 비교기(1110)는 새 명령 신호와 함께 입력으로 LUT(1108)에 영역 식별자를 제공한다. LUT(1108)는 새 명령 및 관련 새 어드레스를 SLDRAM 명령 패킷으로 변환하기 위해 사용되는 최소의 델타 처리 시간 및 데이터 오프셋을 제공한다. 최소의 델타 처리 시간은 방금 처리된 구 명령과 연계하여 새 명령을 처리하는 (클럭 사이클로 표시되는) 델타 시간을 표시한다. 데이터 오프셋 시간은 새 명령처리 후 새 명령과 연계된 판독 데이터 패킷을 수신하기 위해 클럭 사이클로 델타 시간을 표시한다.

발명의 한 실시예에 따라, 제한 블록(1100)은 16개의 어레이 태그 뱅크 레지스터를 포함하고, LUT(1108)는 16개의 관련 레지스터를 각각 가지는 네 개의 타이밍 영역에 대한 네 개의 서로 다른 매개변수를 저장할 수 있다.

도 12는 발명의 한 실시예에 따라 수신된 프로세서 명령에 응답하는 SLDRAM 버스 신호의 타이밍 도표(1200)의 도면이다. 표 4는 여러 발생 신호를 식별함으로서 제한 블록(1100)에 의해 실행되는 스케쥴링 과정을 요약한다. 메모리 명령은 " 명령" 이 실행될 명령을 나타내고 " 어드레스" 가 관련 메모리 위치를 나타내는, {명령, 어드레스}의 형태를 취한다.

표 4와 도 12에서, 시스템 클럭 사이클 $\varnothing_1$ 중, 제 1 {OPENPAGE,1000} 명령이 어드레스 디코더(302)와 영역 비교기(1110)에서 수신된다. 이 예의 경우에, 어드레스 디코더(1102)는 OPENPAGE 명령 어드레스 " 1000" 을 " 100" 과 " 400" 으로 해독한다. " 100" 과 " 400" 은 메모리 영역 0 내에 포함된다고 영역 비교기(1110)가 결정한다. OPENPAGE 명령이 수신되는 첫 번째 명령이기 때문에, 가상 뱅크 B0-13과는 어떠한 " 히트" 도 없으며, 상응하는 대체 카운터는 " 0" 으로 설정된다. 기술되는 실시예에서, 대체 카운터는 슈도-임의 카운팅 기법을 바탕으로 갱신되지만, 다른 실시예에서는 임의 카운팅이나 그외 다른 적절한 기법이 사용될 수 있다. 첫 번째 {OPENPAGE, 1000} 명령이 개방형 명령이기 때문에, 어떤 관련된 최소 델타 처리 시간이나 데이터 오프셋도 없으며, 따라서 어드레스(1000)에서의 페이지는 첫 번째 명령 클럭 사이클 $\varnothing C_1$ 에서 열린다.

다음 시스템 클럭 사이클 $\varnothing_2$ 중, {READ,1000} 명령이 제한 블록(1100)에서 수신되고, 어드레스 디코더(1102)는 이를 100과 400으로 해독하며(즉, 이전 클럭 사이클로부터 메모리 어드레스 위치(1000)에서 열린 페이지를 판독함), 이는 다시 영역 비교기(1110)로 하여금 영역 식별자를 REGION1로 설정하게 한다. 그러나 이 경우에, 이전, 또는 B0 레지스터에 저장된 " 구명령" 은 B0에서 " 히트" 를 유발하고, 이는 선택기로 하여금 " READ" 를, LUT(1108)에 입력되는 " 구명령" 으로 출력하게 한다. 추가 입력은 " READ" 로 " 새명령" 입력과 영역 비교기(1104)에 의해 발생된 영역 식별자 REGION1을 포함한다. LUT(1108)는 3개의 명령 클럭 사이클 $\varnothing$ 3의 최소 델타 처리 시간을 발생시키기 위해 저장된 특성화 데이터를 이용하며, 이는 최소한 3개의 명령 클럭 사이클이 {PAGEOPEN, 1000} 명령 및 관련 {READ, 1000} 명령의 처리를 분리시켜야한다는 것을 의미한다.

이 방식으로, 제한 블록(1100)에서 수신된 각각의 메모리 명령 패킷은 방금 처리된 명령의 일부나 LUT(1108) 내에 저장된 특성화 데이터에 따라 처리된다. 발명의 특정 실시예에 따라 제한 블록으로부터 수신된 명령의 재정렬이 이제부터 기술될 것이다. 도 13A-C는 본 발명의 특정 실시예에 따라 메모리 명령을 재정렬함으로서 실현될 수 있는 일부 장점을 설명하기 위해 간단한 명령 재정렬 예를 통한 타임라인(1302, 1304)이다. 각각의 타임라인은 두 개의 다른 메모리 뱅크에 상응하는 네 개의 판독 명령을 보여준다. CMD0과 CMD1은 관련 메모리의 뱅크 1을 향한 판독 명령이다. CMD2와 CMD3은 관련 메모리의 뱅크 2를 향한 판독 명령이다. 타임라인(1302)은 시스템 프로세서로부터 메모리 제어기에 의해 명령이 수신된 순서로 메모리 제어기와 메모리를 연결하는 명령 버스에 배열되는 메모리 명령을 도시한다. CMD0은 시간 슬롯 0, CMD1은 시간 슬롯 3, CMD2는 시간 슬롯 4, CMD3은 시간 슬롯 7을 점유한다. 각각의 시간 슬롯은 한 개의 클럭 사이클을 나타낸다.

앞서 언급한 바와 같이, 동일한 메모리 뱅크로의 명령은 이전에 처리된 명령의 서비스를 수용하기 위해 처리간 최소의 지연을 가져야 한다. 이는 각 쌍의 명령 사이 두 시간 슬롯에 의해 도 13A에 나타난다. 도시되는 바와 같이, 네 개의 판독 명령이 도 13A에 도시되는 순으로 메모리로 보내질 경우, 명령 버스는 시간 슬롯 1, 2, 5, 6의 네 가용 클럭 사이클 중 비사용 상태로 진행될 것이다. 본 발명에 따라 명령을 재정렬함으로서 이 비효율성의 일부가 개선될 것이다.

도 13B와 13C의 타임라인(1304, 1306)은 발명의 특정 실시예에 따라 도 13A의 명령을 재정렬하는 것을 도시하고, 이에 의해 얻은 장점을 도시한다. 본 예에서, 데이터 버스 상의 충돌은 단순화로 인해 고려되지 않는다. 그러나 아래에 설명되는 바와 같이, 메모리 명령의 효율적 재정렬을 위해 이러한 고려에 대해 관심이 주어져야 한다. CMD2와 CMD3가 CMD0과 CMD1과는 다른 메모리 뱅크를 향하기 때문에, 두 쌍의 명령간 메모리 접근 호출은 관련성이 없고 무시될 수 있다. 즉, CMD0 바로 다음의 시간 슬롯 1에 CMD2를 위치시키고 CMD1 바로 다음의 시간 슬롯 4에 CMD3을 위치시키도록 타임 라인(1304)에 도시되는 바와 같이 명령이 재정렬될 수 있다. 이는 CMD0과 CMD2의 처리 사이, 또는 CMD1과 CMD3의 처리 사이에 어떤 지연도 필요하지 않기 때문이다. 그 이유는 이들이 다른 뱅크의 메모리를 지향하기 때문이다. 그러나, 도 13C에 도시되는 바와 같이, 같은 뱅크로 향하는 명령 쌍 사이에서는 최소 지연 시간, 예를 들어 2클럭 사이클이 유지되어야 한다. 즉, 명령 재정렬이 동일한 메모리 뱅크로의 일련의 명령 사이 지연시간을 감소시키려는 시도를 포함하지 않을 수 있다.

명령의 재정렬 결과가 도 13C에 도시되고, 여기서 네 개의 명령이 5개의 클럭 사이클에서 처리되고 이때 시간 슬롯 2는 사용되지 않는 상태를 유지한다. 또다른 메모리 뱅크로의 제 5 메모리 명령이 시간 슬롯 2에 삽입되어, 명령 버스가 사용되는 효율을 최대화시킬 수 있다.

도 14는 발명의 특정 실시예에 따라 고안된 메모리 제어기의 일부의 블록 도표이다. 재정렬 회로(1400)는 유입 메모리 명령의 순서, 1, 2, 3을 시스템 프로세서로부터 수신한다. 특정 실시예에 따라, 메모리 명령은 제한 회로(도시되지 않음)를 통해 재정렬 회로(1400)로 전송된다. 상기 제한 회로는 앞서 기술한 바와 같이 관련 메모리의 동일한 논리 뱅크로 향하는 타명령에 대해 선택된 명령 상의 처리 시간 제약을 부과한다. 명령은 명령 큐(1402)에서 재정렬되고, 이로부터 명령이 메모리로 처리된다. 본 예에서, 명령은 순서 1, 3, 2로 재정렬된다.

원래의 메모리 명령 순서 1, 2, 3은 데이터 판독 회로(1406)의 FIFO 메모리(1404)에 저장된다. FIFO(1404)의 순서는 메모리 제어기에 의해 명령이 원래 수신된 순서에 상응하도록 메모리로부터 수신된 데이터를 재정렬하는 데 사용된다. 그러나, 과정의 일부가 정렬된 데이터를 기대함에 반해, 일부는 정렬되지 않은 데이터를 기대한다. 따라서, FIFO(1404)를 필요한 대로 ON/OFF 스위칭함으로서, 어떤 종류의 데이터 정렬도 지원할 수 있다. 이는 프로세서가 메모리 제어기에 원래 명령을 전송한 순서에 상응하는 순서로 데이터를 수신하는 것을 " 기대" 하기 때문이다.

추가적으로, 프로세서가 메모리 명령을 전송한 원래 순서에 상응하지 않는 순서로 메모리 제어기에 의해 메모리로부터 데이터가 수신될 수 있기 때문에, 제 3 순서가 데이터 큐(1408)에 저장된다. 본 순서(본 예에서 3, 1, 2)는 명령 순서 1, 3, 2에 상응하는 데이터가 데이터 판독 회로(1406)에 의해 수신될 순서를 나타낸다. 데이터 큐 순서는 메모리의 여러 논리 뱅크와 연관된 공지된 호출과 명령 큐 순서를 바탕으로 회로(1400)를 재정렬함으로서 연산된다. 데이터 큐(1408)에 저장된 순서(3,1,2)로 메모리가 데이터를 메모리 제어기에 전송할 때, 원래의 명령 순서 1, 2, 3에 상응하는 순서로 데이터가 프로세서에 전송되도록, 데이터는 판독 데이터 버퍼(1410)에 저장되고, FIFO(1044)와 데이터 큐(1408)를 바탕으로 재정렬된다.

도 15는 발명의 특정 실시예에 따라 고안된 메모리 제어기의 재정렬 회로(1500)의 블록도표이다. 재정렬 회로(1500)는 시스템 프로세서로부터 수신된 명령을 저장하고 재정렬하는 명령 큐(1502)를 포함한다. 명령 큐(1502)는 각각의 명령에 대한 처리 시간을 계산하고, 명령을 처리하며, 큐로부터 처리 명령을 제거한다. 이때 메모리의 동일한 논리 뱅크로의 명령과 연관된 명령 처리 시간 제약뿐 아니라 데이터 버스 이용 제약까지를 이용한다.

데이터 큐(1504)는 처리된 메모리 명령에 상응하는 데이터 발생 시간을 나타내는 데이터 요소를 저장하고, 큐의 각각의 새 엔트리에 대한 새 데이터 발생 시간을 나타내며, 상응하는 메모리 트랜잭션이 완료될 때 큐 엔트리를 제거한다.

비교기 매트릭스(1506)는 명령 큐(1502)로부터 처리될 준비가 된 명령의 데이터 발생 시간을 데이터 큐(1504)에 나타나는 바와 같이 이전에 처리된 명령의 데이터 발생 시간과 비교하는 충돌 감지 함수를 실행한다. 충돌이 감지되면, 명령 처리가 지연된다.

도 16은 도 15의 재정렬 회로(1500)의 보다 상세한 블록도표이다. 명령 큐(1502)는 6개의 명령 큐 요소(1602)로 이루어지고, 그 각각의 요소(1602)는 도 17의 블록도표에 의해 도시되는 바와 같이 특정 메모리 명령에 관한 61비트의 정보를 저장한다. 명령 필드(1702)는 메모리 명령을 구체화하는 40-비트 메모리 명령 패킷을 포함한다. 명령 처리 시간(Cd) 필드(1704)는 명령이 처리되기 전에 클럭 사이클의 델타 시간을 나타내는 6비트 필드이다. 필드(1704)의 값은 앞서 기술한 바와 같이 제한 회로에 의해 결정되고, 메모리의 동일한 논리 뱅크에 상응하는 가장 최근의 메모리 명령에 관한 것이다. 즉, Cd 필드의 값은 동일한 뱅크의 두 명령 사이 호출을 표시한다. 각 뱅크에 대해 필요한 호출에 관한 정보는 제한 회로에 저장되고 메모리의 물리적 특성에 의해 주로 결정된다. 명령 큐 내에서, Cd 필드는 일부 예외 사항을 제외하곤 각 클럭 사이클에 대해 한번씩 감소된다. 가령, 동일한 논리 뱅크로의 일련의 명령간 호출은 변경될 수 없다. 따라서, 특정 뱅크를 지향하는 명령에 대한 Cd 필드가 0에 도달하고 처리되지 않을 경우, 동일한 뱅크로의 모든 다른 명령에 대한 Cd 필드는 제 1 명령이 처리될 때까지 감소할 수 없다.

데이터 발생 시간(Dd) 필드(1706)는 명령 큐로부터 상응하는 데이터의 전송까지 메모리 명령의 처리 사이 클럭 사이클의 델타 시간을 표시하는 6-비트 필드이다. Dd 필드(1706)는 명령 큐에서 변경되지 않을 수 있다. 명령 ID 필드(1708)는 명령 패킷(1702)의 명령을 독자적으로 식별하는 5-비트 필드이다. 이 정보는 데이터 큐와 FIFO의 상응하는 정보와 함께 사용되어, 명령 및 데이터의 재정렬이 실행될 수 있도록 트랙의 어느 패킷이 패킷이고 어느 데이터가 어느 패킷에 상응하는 지를 유지한다. 논리 뱅크(B) 필드(1710)는 명령 패킷이 향하는 메모리의 논리 뱅크를 식별하는 3-비트 필드이다. 마지막으로, 버스트 표시기(burst indicator)(Db) 필드(1712)는 요청되거나 기록된 데이터가 1 클럭 사이클인지 2 클럭 사이클인지를 표시하는 1-비트 필드이다.

도 16에서, 명령 큐의 작동은 명령 큐 제어기(1604)에 의해 제어된다. 제어기(1604)는 트랙의 명령 큐 요소(1602)가 가용하도록 트랙을 유지하고, 자유 위치 표시기(1606)를 통해 특정 큐 요소(1602)로 유입 명령의 삽입을 제어한다. 제어기(1604)는 상응하는 명령이 처리되면, 명령 큐 요소 정보의 데이터 큐(1504)로의 삽입을 용이하게 한다. 특정 실시예에 따라, 명령이나 데이터 버스의 자유 시간 슬롯의 가용성에 관계없이 명령이 명령 큐(1502) 내로 삽입된다.

Cd 카운트가 0이고 표시된 데이터 버스에 충돌이 없을 경우, 멀티플렉서(1608)를 통해 명령 큐 요소(1602) 중 하나로부터 명령 버스까지 명령이 처리될 수 있다. 즉, 명령 버스나 데이터 버스 상의 자유 시간 슬롯이 반드시 식별되어야 한다. 명령이 판독이나 기록이 아닐 경우(따라서 어떤 데이터 버스 리소스도 요구하지 않을 경우), 명령 버스 시간 슬롯만이 필요하다. 명령이 판독이나 기록일 경우, 명령 버스와 데이터 버스 상의 시간 슬롯이 모두 필요하다.

제어기(1604)의 제로 비교기(1610)는 Cd=0인 지를 먼저 결정하기 위해 사용된다. 감산기(1612)는 앞서 기술한 예외가 없을 경우 각각의 클럭 사이클에서 각각의 명령 큐 요소(1602)에 대해 Cd 카운트로부터 " 1" 을 빼기 위해 사용된다. 즉, 처리될 수 없는 특정 명령에 대해 Cd=0이다. 이러한 경우에, 모든 큐 요소에 대한 Cd 및 B 필드를 이용하여, 큐 제어기(1604)는 동일한 논리 뱅크로의 모든 명령에 대해 CD 카운트가 감소하는 것을 방지하는 마스크 신호(M)를

발생시킨다.

특정 실시예에 따라, Cd=0을 가지는 두 개의 큐 요소가 존재할 경우, 높은 우선순위의(가령, 가장 오래된) 큐 요소가 처리된다. 어드레스 시프터(1614)는 큐 명령의 우선순위를 결정한다. 이는 도 18을 참고하여 더욱 상세하게 설명될 것이다. 발명의 또다른 특정 실시예에 따라, Cd 카운트가 이미 0일 때 새 명령이 명령 큐에 도달하면, 새 명령은 멀티플렉서(1608)를 통해 메모리에 직접 전송될 것이다. Cd 카운트가 0이 아니거나, Cd=0이고 높은 우선순위의 명령 큐에 다른 명령이 저장될 경우, 새 명령이 명령 큐 요소(1602)에 저장된다. 그러나, 명령 큐가 비면, 새 명령은 즉시 처리될 것이다(만약 Cd가 0이라면).

판독 또는 기록 명령의 경우에, 처리될 준비가 된 명령을 포함하는 명령 큐 요소(1602)의 Dd와 Db 필드를 이용하여 충돌이 감지된다. 명령에 상응하는 데이터의 듀레이션과 발생 시간은 큐 제어기(1604)에 의해 제어되는 멀티플렉서(1508)를 통해 비교기 매트릭스(1506)에 전송된다. 즉, 큐 제어기(1604)는 명령 처리 시간, 즉 Cd가 0인 큐 요소의 데이터 발생 시간과 듀레이션(1 클럭 사이클 또는 2 클럭 사이클)을 전송하기 위해 멀티플렉서(1508)를 제어한다. 이 듀레이션은 Dd+1에 대해 " 0" (1 클럭 사이클을 표시함)이거나 " 1" (2 클럭 사이클을 표시함)을 도출하도록, 가산기(1616)로 Db 비트를 데이터 발생 시간 Dd에 더함으로서 1 클럭 사이클이나 2 클럭 사이클임이 표시된다. 데이터 발생 시간과 듀레이션은 데이터 큐(1504)에 저장된 5개의 이전 처리된 명령의 데이터 발생 시간 및 듀레이션과 비교기 회로(1506)에서 비교된다. 특정 실시예에 따라, 비교기 매트릭스(1506)는 2*10 병렬 비교기 매트릭스를 포함한다.

도 18은 도 16의 어드레서 시프터(1614)의 특정 실시예의 블록도표이다. 앞서 언급한 바와 같이, 어드레스 시프터(1614)는 명령의 우선순위를 결정한다. 앞서 또한 언급한 바와 같이, 자유 위치 표시기(1606)에 따라 새 명령이 어떤 자유로운 명령 큐 요소(1602)로 삽입된다. 새 명령이 삽입될 명령 큐 요소(1602)의 어드레스는 가장 높은 우선순위로 첫 번째 자유 위치(A0-A5)에 삽입된다. 그 결과는 어드레스 시프터(1614)의 A0 위치가 아직 처리되지 않은 가장 오래된 명령에 대한 큐 요소 어드레스를 저장한다. 명령이 명령 큐로부터 처리될 때, 어드레스 시프터(1614)의 상응하는 엔트리가 제거되고, 낮은 우선순위의 명령에 대한 어드레스가 높은 우선순위의 위치로 이동된다. 앞서 언급한 바와 같이, 명령 큐의 명령에 대한 Cd 카운트가 0에 도달할 경우, 처리될 수 있다. 그러나, Cd=0인 명령이 두 개 이상 있을 경우, 어드레스 시프터(1614)의 어드레스 위치에 의해 표시되는, 가장 오래된 명령, 즉 가장 높은 우선순위를 가지는 명령이 처리된다.

도 16의 데이터 큐(1504)는 다섯개의 큐 요소(1652)를 포함하며, 그 각각의 요소는 도 19의 다이어그램에 표시되는 바와 같이 이전에 처리된 메모리 명령에 관한 12 비트의 정보를 저장한다. 데이터 발생 시간(Dd) 필드(1902)는 상응하는 데이터의 수신과 명령 큐로부터 명령의 처리 사이 클럭 사이클의 델타 시간을 표시하는 6-비트 필드이다. 각각의 데이터 큐 요소(1652)에 대한 Dd 카운트는 그 값이 0에 이를 때까지 감산기(1654) 중 하나를 이용하여 모든 클럭 사이클에서 감소한다. Dd=0일 때, 상응하는 데이터는 데이터 버스 상에 있다. 따라서, 오직 한 개의 데이터 큐 요소(1652)만이 주어진 시간에 Dd=0을 가질 수 있다. Dd 카운트가 0에 도달하며, 상응하는 데이터 큐 요소의 정보가 데이터 큐(1504)로부터 제거된다.

명령 ID 필드(1904)는 데이터와 대응하는 처리 명령을 독자적으로 식별하는 5-비트 필드이다. 이 정보는 메모리 제어기로 원래 전송된 명령의 순서에 상응하도록 데이터를 재정렬시키는 데 유용하다. 마지막으로, 버스트 표시기(Db) 필드(1906)는 데이터가 1 클럭 사이클을 점유하는 지 2클럭 사이클을 점유하는 지를 표시하는 1비트 필드이다.

도 16과 앞서 기술한 내용에서, 각각의 데이터 큐 요소(1652)에 대한 듀레이션과 데이터 발생 시간(Dd)는 Cd=0인 처리될 준비가 된 명령 큐(1502)의 명령에 대한 듀레이션과 Dd와 비교기 매트릭스(1506)에서 비교된다. 듀레이션은 Dd+1에 대해 " 0" (1 클럭 사이클을 나타냄)이나 " 1" (2 클럭 사이클을 나타냄)을 도출하도록, 가산기(1656)로 Db 비트를 데이터 발생 시간 Dd와 더함으로서, 1 클럭 사이클인지 2 클럭 사이클인지가 표시된다. 이 비교가 데이터 버스에 충돌이 없음을 보이면, 명령은 명령 큐로부터 처리된다.

데이터 큐 제어기(1658)는 데이터 큐(1504)의 작동을 제어한다. 명령 큐 제어기(1604)와 함께 자유 위치 표시기(1660)는 데이터 큐 요소(1652)로 새 데이터 큐 요소 정보의 삽입을 용이하게 한다. 상응하는 메모리 접근이 완료될 때 자유 위치 표시기는 데이터 큐 요소(1652)로부터 정보 제거를 또한 용이하게 한다. 제로 비교기(1662)와 버스트 표시기(1664)는, 데이터 큐 요소(1652)에 대한 Dd가 언제 0인지, 데이터 전송이 데이터 버스를 언제 더 이상 점유하지 않는 지, 그래서 상응하는 정보가 언제 데이터 큐로부터 제거될 수 있는 지를 결정하기 위해 사용된다.

발명의 또다른 특정 실시예에 따라, 충돌 감지는 비교기와 멀티플렉서의 2차원 어레이를 이용하여 보다 복잡해진다. 이 접근법은 앞서 기술한 1차원식 접근법보다 집약적이며, 처리될 준비가 된 명령에 대해 오직 한 개의 요소보다는 명령 큐의 모든 요소를 본다. 이는 이전에 처리된 명령에 대한 명령만을 스케쥴링하는 것이 아니라, 데이터 버스 상의 데이터 패킷의 순서에 대해서도 스케쥴링을 한다.

새 명령을 삽입하기 위해, 명령 파이프의 처리예정 부의 두 일련의 단계의 각각의 세트는 새 명령이 둘 사이에 삽입될 수 있는 지를 확인하기 위해 반드시 비교되어야 한다. 비교는 명령이 삽입될 수 있는 범위를 실제로 결정한다. 그 범위는 다음과 같다.

$CLEN_x$ = 명령 길이

$T_{cstart} = t_{cA} + CLEN_A$ ... (1)

$T_{cend} = t_{cB}$ ... (2)

t_{cA} 와 t_{cB} 가 같은 경우는 일련의 파이프라인 요소 A와 B에 대한 처리 시간이다. 파이프라인 요소 A가 파이프라인 요소 B의 앞에 있고, 따라서 처리 시간이 둘 중 낮다. 삽입이 있어야 하면, 물론 A와 B 요소 사이에 한 개 이상의 열린 슬롯이 존재하여야 한다. 따라서,

$N = T_{cend} - T_{start} + 1$ ... (3)

(N = 요소 A와 B 사이의 처리 슬롯의 수)

LEN < = tcb - $_t$ ca - $_C$ LENA.. . (4)

하드웨어에서, 다음의 조건을 간단히 구현하기 쉽다.

$(t_{cB} - CLEN_A) - (t_{cA} + CLEN_A) => 0$

이 범위의 시점과 종점은 관련된 데이터 슬롯의 가능한 범위를 또한 구체화한다 이 범위는 중복이 있는 지, 그리고 새 범위는 어떤 지를 확인하기 위해 데이터 파이프의 일련의 요소의 각각의 세트와 비교되어야 한다. 이 비교에서 다섯가지의 경우가 존재한다.

케이스 0케이스 0

데이터 슬롯 t_{dA} 와 t_{dB} 에 의해 기술되는 범위가 두 일련의 요소 M과 N의 범위의 바깥쪽에 있을 때,

$t_{dA} + CLEN_A => tdN...$ (6)

또는, $DLEN_X$ = 데이터 길이일 때,

$t_{dB} < = tdM+ _D LENM...$ (7)

쌍 M과 N 사이에 어떤 가능한 데이터 슬롯도 존재하지 않는다.

케이스 1케이스 1

데이터 슬롯 t_{dA} 와 t_{dB} 에 의해 기술되는 범위가 두 일련의 요소 M과 N의 범위의 내측에 있을 때,

$t_{dA} + CLEN_A => tdM+ _D LENM...$ (8)

$t_{dB} - CLEN + DLEN < = tdM...$ (9) (이때, CLEN은 새 명령의 길이, DLEN은 슬롯으로 표시한 새 데이터 길이)

이 경우에 가장 이른 가능한 데이터 슬롯 시간은 $t_{dA} + LEN_A$ 이고, 상응하는 명령 처리 시간은 $t_{cA} + CLEN_A$ 이다.

케이스 2케이스 2

이 경우에, 데이터 슬롯 t_{dA} 와 t_{dB} 에 의해 표현되는 범위는 요소 M의 양끝을 연결한다.

$t_{dA} + CLEN_A < tdM+ _D LENM...$ (10)

$t_{dB} - CLEN + DLEN > tdM+ _D LENM$그리고 $tdB - _{CL} EN + DLEN < tdM...$ (11)

이 경우에, 가장 이른 가능한 데이터 슬롯 시간은 $t_{dM} + DLEN_M + 1$이고, 이때 상응하는 명령 처리 시간은 $t_{dM} + CLEN_M$ - DATA_OFFSET이며, DATA_OFFSET은 명령 처리 시간과 데이터 점유 사이의 시간이다.

케이스 3케이스 3

이 경우에, 데이터 슬롯 t_{dA} 와 t_{dB} 에 의해 표현되는 범위는 요소 N의 양끝 사이이다. 이 경우에,

$t_{dA} + CLEN_A > tdM+ _D LENM...$ (12)

$t_{dA} + CLEN_A + DLEN < tdN...$ (13)

이 경우에, 가장 이른 가능한 데이터 슬롯 시간은 $t_{dA} + CLEN_M$ 이고, 이때 상응하는 명령 처리 시간은 $t_{dA} + CLEN_A + 1$이다. 케이스 1은 이 경우에 또한 포함될 수 있다.

케이스 4케이스 4

이 경우에, 데이터 슬롯 t_{dA} 와 t_{dB} 에 의해 표현되는 범위는 요소 M과 N에 의해 정의되는 범위를 포함한다. 이 경우에,

$t_{dA} + CLEN_A < tdM+ _D LENM...$ (14)

$t_{dB} - LEN > CtdN...$ (15)

따라서, 이 경우에, 가장 이른 가능한 데이터 슬롯 시간은 $t_{dA} + CLEN_M$ 이고, 이때 상응하는 명령 처리 시간은 $t_{dM} + CLEN_A + DATA_OFFSET$이고, $DATA\_OFFSET = t_{dA} - t_{cA}$ 이다.

가장 이른 가능한 슬롯이 항상 취해짐에 따라 스케쥴링 용도로 케이스 1과 케이스 3은 동일하다는 것이 명백하다. 따라서 조합된 케이스는 케이스 3이다. 마찬가지로, 원하는 결과가 $t_{dM} + LEN_M$ 임에 따라 케이스 2와 케이스 4 역시 동일하다. 이 경우에, t_{dA} 와 t_{dB} 에 의해 주어지는 범위에 의해 t_{dM} 이 걸쳐진다고는 것이 반드시 간단하게 나타나야 한다. 부가적으로, 유입 명령에 대한 가장 이른 가능한 처리 시간 t_c와 데이터 슬롯 t_d가 반드시 고려되어야 한다. 각 명령 파이프 쌍에 대한 각각의 데이터 파이프 쌍에서 반드시 이루어져야 하는 비교는 아래의 전산 프로그램과 같다.

```
if(((tcB – CLEN) => (tcA +CLENA)) && (tc <= (tcA + CLENA))){
        if(((tdA + CLENA) <= (tdM + DLENM)) && ((tdB – DLEN – (tdM + DLENM)) >=
0)){
                td = tdM + DLENM;
                tc = tcA – tdA + tdM + DLENM;
        }
        else if(((tdN - (tdA + CLENA + DLENA)) >= 0) && (tdA + CLENA) >= (tdM +
        DLENM)){
                td = tdA + CLENA;
                tc = tcA + CLENA;
        }
        else {
                td = IMPOSSIBLE;
                tc = IMPOSSIBLE;
        }
}
else if(((tcB – CLEN) => tc) && (tc > (tcA + CLENA))){
        if((td < (tdM + DLENM)) && ((tdB – DLEN – (tdM + DLENM)) >= 0)){
                td = tdM + DLENM;
                tc = tc – td + tdM + DLENM;
        }
        else if(((tdN - (td + DLEN)) >= 0) && td >= (tdM + DLENM)){
                td = td;
                tc = tc;
        }
        else {
                td = IMPOSSIBLE;
                tc = IMPOSSIBLE;
        }
}
else {

        td = IMPOSSIBLE;
        tc = IMPOSSIBLE;
}
```

따라서, 명령 파이프에 대해 만족한 명령은 다음과 같다

$t_{cB} – CLEN => t_{cA} + CLEN_A$
$t_{cB} – CLEN => t_c$
$t_c + CLEN <= t_{cB}$
$t_c > t_{cA} + CLEN_A$
$t_c <= t_{cA} + CLEN_A$

데이터 파이프에 대해 만족한 명령은 다음과 같다

$t_{dA} + CLEN_A <= t_{dM} + DLEN_M$
$t_{dA} + CLEN_A >= t_{dM} + DLEN_M$
$t_{dB} – DLEN >= t_{dM} + DLEN_M$
$t_{dN} >= t_{dA} + CLEN_A + DLEN_A$
$t_d < t_{dM} + DLEN_M$
$t_{dN} >= t_d + DLEN$
$t_d >= t_{dM} + DLEN_M$

따라서, 결정 논리 장치는 앞서 정의한 바와 같이 비교기 셀의 매트릭스로 구성된다. 최적의 선택은 가장 이른 명령 처리 시간이고, 이는 간단한 우선순위 인코더에 의해 결정된다.

재정렬 파이프 제어 논리 장치는 명령 및 데이터 파이프의 각 요소에 어떤 작동이 행하여질 지를 동적으로 결정하여야 한다.

현재의 명령 파이프에서, 각각의 파이프 요소는 4개의 가능한 작동을 가진다. 즉, 이전 요소로부터의 판독(파이프 진보), 전류 내용 유지(파이프 유지), 다음 요소로부터 판독(파이프 백업), 그리고 유입 명령 버스로부터 판독의 네가지를 포함한다. 앞서 네 경우에 의해 규정되는 바와 같이 파이프의 여러 점에서 여러 세트의 조건이 존재할 수 있다. SLiMAC에 처리될 요소는 요소 0으로 규정되고, 처리로부터 가장 먼 요소는 M으로 규정된다. 요소 N으로의 삽입이 이루어지는 시기는 재정렬 결정 논리 장치가 현 파이프라인의 최적의 삽입 점이 N - 1과 N 사이라는 것을 알 때이다.

케이스 1 - 유지(HOLD)케이스 1 - 유지(HOLD)

SLiMAC에 어떤 처리도 없거나, 새 명령의 삽입이 없을 경우에 파이프가 유지(hold)된다.

케이스 2 - 유지(HOLD) 및 삽입(INSERT)케이스 2 - 유지(HOLD) 및 삽입(INSERT)

이 경우에, SLiMAC에는 어떤 처리도 없지만, 파이프에는 새 명령이 삽입된다. 삽입이 요소 N에서 발생하면, 파이프는 요소 0으로부터 요소 N-1까지 유지될 것이고, 요소 N에서 삽입되어, 요소 N+1로부터 요소 M까지 백업될 것이다.

케이스 3 - 처리(ISSUE)케이스 3 - 처리(ISSUE)

이 경우에, 요소 0으로부터는 SLiMAC에 한 개의 처리가 존재하고, 나머지 파이프는 진행하여, 요소 0이 요소 1의 내용을 포함할 것이고, 요소 1은 요소 2의 내용을 포함할 것이며, 이와 같이 하여 궁극적으로는, 요소 M-1이 요소 M의 내용을 포함한다.

케이스 4 - 처리(ISSUE) 및 삽입(INSERT)케이스 4 - 처리(ISSUE) 및 삽입(INSERT)

이 경우에, 요소 0으로부터 SLiMAC로 처리가 존재하고, 요소 N에서 삽입이 존재한다. 이 경우에, 요소 0에서 N-2까지는 주어진 진행 작동이고, 요소 N-1은 주어진 삽입 작동이며, 요소 N-M은 유지될 것이다. 진행이 요소로부터 데이터를 저장하는 요소로 주어진다면, 요소 N에서의 삽입은 삽입 요소가 비갱신 파이프의 위치 N-1에서 종료될 것이라는 것을 실제로 의미한다.

도 20은 도 15에 도시되는 충돌 감지 시스템(1500)의 또다른 구현인 충돌 감지 시스템(2000)을 도시한다. 본 실시예에서, 충돌 감지 시스템(2000)은 타겟 응답 제한을 바탕으로 최적의 명령 순서를 달성하도록 명령을 재정렬하고, 내부 제어기와 타겟 서브시스템 사이의 데이터 전송에 대한 최적의 슬롯을 결정한다. 명령의 재정렬이 데이터 버스 상의 다른 데이터 패킷의 충돌을 유발할 수 없기 때문에, 특정 명령에 관한 명령 데이터 전송이 데이터 충돌을 유발할 경우 이러한 특정 명령의 처리를 금지하는 충돌 감지기(2002)가 필요할 것이다. 기술되는 실시예에서, 충돌 감지 시스템(2000)은 명령 큐(2004)에 연결된 충돌 감지기(2002)를 포함한다.

기술되는 실시예에서, 충돌 감지기(2002)는 " 처리될" 명령(명령 큐(2004)에 저장됨)과, " 이미 처리된" 명령(데이터 큐(2006)에 저장됨) 사이의 모든 가능한 데이터 충돌을 감지한다. 기술되는 실시예에서, 멀티플렉서(2008)에 각각 연결되는 N개의 명령 큐(2004)가 존재한다. N개의 명령 큐(2004) 각각은 처리된 상기 명령들, 명령이 타겟 장치로 처리된 후 범용 제어기(104)와 타겟 장치(108) 간에서 데이터 버스 상에 데이터 전송이 나타날 때를 표시하는 시간 인자 " d_time_{ND} " , 데이터 버스트 전송을 나타내는 버스트 비트(b_{ND}), 그리고 판독/기록 비트(rw_{ND})를 저장하도록 배열된다. 기술되는 실시예에서, 데이터 큐(2006)는 이미 타겟 장치로 처리된 요청에 대해 타겟 장치(108)와 제어기(104) 사이에 데이터 버스 상에서 데이터 전송이 나타날 때를 표시하는 시간 인자 " d_time_{D} " 를 저장한다. 명령 큐(2006)는 버스트-비트(b_{ND})와 판독/기록 비트(rw_{ND})를 또한 저장한다.

선호되는 실시예에서, 충돌 감지 시스템(2000)은 처리될 상기 명령을 저장하고 재정렬하도록 큐 및 링크 제어기 유닛(2008)을 포함한다. 큐 및 제어기 유닛(2008)은 데이터가 데이터 버스 상에 나타나는 시간과 명령의 새 처리 시간을 또한 계산한다. 큐 및 제어기 유닛(2008)은 명령 큐로부터 데이터 큐로 처리 요소를 전송하고, 명령이 처리된 후 명령 큐로부터 처리 요소를 제거한다. 큐 및 제어기 유닛(2008)은 메모리로의 접속이 완료된 후 데이터 큐로부터 데이터 요소를 또한 제거한다.

도 21에서, 타겟 장치로의 모든 판독/기록 명령은 데이터 패킷 전송과 관련된다. 타겟 장치로의 명령 처리 이전에, 새

데이터 패킷 ND(새 데이터)는 데이터 큐에 충돌없이 삽입될 수 있는 지를 알아보기 위해 타이밍 정보에 따라 확인된다. 도 21에 도시되는 본 예에서, 처리 데이터 패킷 D는 데이터 큐에 이미 위치하고, 새 데이터 패킷 ND는 처리 데이터 패킷 D와 비교된다. 처리 데이터 패킷 D와 새 데이터 패킷 ND는 버스트 접근을 나타낸다. 따라서 본 예에서, 새 데이터 패킷 ND가 데이터 충돌없이 처리 데이터 패킷 D에 대해 위치할 수 있는 방법에는 두가지 가능성이 있다. 새 데이터 패킷 ND는 처리 데이터 패킷 D의 좌측이나 우측에 위치할 수 있다.

이 특정 예는 비-버스트 및 버스트 데이터 전송을 지원하는 메모리 제어기의 충돌 감지를 도시한다. 데이터 버스의 양방향 특성으로 인해, 1 클럭 사이클이 일련의 판독-기록이나 기록-판독 전송 사이에 삽입되어야 한다.

가능한 결과에는 여러 가지가 있으며, 그 중 일부가 아래에 나열된다.

1) ND가 D 뒤나 앞에 위치할 경우 D와 ND 사이에 충돌이 없다.

2) 일련의 판독-기록이나 기록-판독 데이터 전송 사이에 1 클럭-사이클이 삽입되어야 한다. 명령 및 데이터 큐의 모든 요소는 작동이 " 판독 데이터" (rw=0)인 지, " 기록 데이터" (rw=1)인 지를 나타내는 " rw" 비트를 저장한다.

3) 데이터 패킷은 한 개의 데이터 스트림(비-버스트 전송)이나 네 개의 스트림(버스트 전송)으로 이루어진다. 명령 및 데이터 큐의 모든 요소는 작동이 " 버스트 전송" (버스트=1)인 지, " 비-버스트 전송" (버스트=0)인 지를 나타내는 " 버스트" 비트를 저장한다.

처리될 각각의 명령에 대하여 처리될 데이터 패킷과 처리된 데이터 패킷 쌍에서 반드시 이루어져야할 비교는 아래의 전산 프로그램과 같다.

```
// 변수 초기화
collision = NO;

// 명령 큐로부터 새 패킷의 종료가 버스트 비트에 따라 결정됨

if    (burst_ND = 1) then d_time_end_ND = d_time_ND + 3
else                      d_time_end_ND = d_time_ND

 for i=1 to last_element_from_Data_Queue
 begin
    // 데이터 큐로부터 패킷의 종료가 버스트 비트에 따라 결정됨
if    (burst_D[i]  = 1) then d_time_end_D[i]    = d_time_D[i] + 3
else                         d_time_end_D[i]    = d_time_D[i]

// 두 연속의 판독/기록이나 기록/판독 사이에서 1 클럭이 구현되어야 함

if    (rw_D[i] = rw_ND) then
begin
      d_time_end_D[i]    = d_time_end_D[i]    + 1
      d_time_end_ND      = d_time_end_ND      + 1
end

// 충돌 감지
      if    NOT((d_time_ND > d_time_end_D[i]) or (d_time_D[i] >
d_time_end_ND))  collision = YES;
       end.
```

발명의 또하나의 실시예에서, 두 일련의 메모리 접근 사이의 시간을 예측하는 장치 및 방법이 공개되어, 새 명령에 대한 가장 이른 " 명령 처리 시간" 을 신속하게 계산할 수 있다. 도 22에서, 특정 페이지에 마지막으로 처리된 명령과 메모리로의 예측된 다음 접근 사이의 시간을 저장하는 N 페이지 타이머(2202)를 가지는 예측기 시스템(2200)이 도시된다. 동일한 페이지에 대한 다음 접근은 " 닫힘" , " 열림" , " 기록" , 또는 " 판독" 중 하나일 수 있다. 유입되는 새 명령(가령, 판독)은 처리 전에 특정 페이지 접근이 기다려야하는 시간을 표시하는 한 개의 특정 페이지 타이머를 선택한다. 동일한 새 명령은 이 명령(판독)과, 동일한 페이지로의 다음 가능한 접근(닫힘, 열림, 기록, 그리고 판독) 사이에 삽입되어야할 타이밍 탐색표(2204)의 적절한 내용을 선택한다. 타이머의 해상도는 1 클럭 사이클이다. 타이밍 탐색표 - 데이터는 명령 처리 후 데이터 버스 상의 데이터가 몇 사이클동안 유효한 지를 표시하는 시간을 저장한다. 새 명령이 작동하지 않으면, 그 값이 0에 이를 때까지 모든 페이지 타이머의 값은 모든 사이클이다.

도 23의 발명의 또하나의 실시예에서, 발명의 한 실시예에 따라 장치 접근 우선순위화 장치(2302)를 가지는 장치 제어기(2300)가 도시된다. 기술되는 실시예에서, 우선순위화 장치(2302)는 요청 제어기 유닛(2304)에 연결되는 다수의 장치 요청을 수신하고 저장하도록 요청 큐(2303)를 포함한다. 상기 요청 제어기 유닛(2304)은 요청 큐(2303)의 어떤 위치로부터도 특정 응답을 페치하기 위해 사용되고, 다수의 공유 장치(108) 중 적절한 하나에 페치 응답을 전송하도록 사용된다. 기술되는 실시예에서, 우선순위화 장치(2302)는 요청 장치(102)에 운반될 특정 저장 응답에 사용되는 응답 제어기 유닛(2308)에 연결되는 공유 장치(108)로부터 응답을 수신하고 저장하도록 배열되는 응답 큐(2306)를 또한 포함한다.

선호되는 실시예에서, 각각의 응답 및 요청은 도 1E에 도시되는 ID 번호(150)와 연관된다. 그래서, 각각의 요청 및 그 응답은 동일한 ID 번호(150)를 가진다. 앞서 기술한 바와 같이, ID 번호(150)는 5개의 데이터 비트를 포함한다. 제 1, 2 데이터 비트는 특정 응답/요청이 속하는 요청 장치의 그룹을 식별하는 그룹 선택기 필드(152)이다. 또한 앞서 기술한 바와 같이, 요청 번호 필드(RN)(153)는 그룹 선택기 필드(152)에 의해 식별되는 요청 장치 그룹과 연계된 요청/응답의 수를 나타낸다. 그래서, 동일한 요청 장치로부터의 일련의 요청이 일련의 요청 번호 필드(153)를 가진다.

작동 중에, 요청 및 응답 제어기(각각, 2304, 2308)는 그룹 우선순위 선택기 레지스터(154), 지속 로크 카운터 레지스터(156), 재정렬 선택기(2312)를 통합시킨다. 그룹 우선순위 선택기 레지스터(154)는 RN(152)에 의해 식별되는 특정 요청/응답 그룹에 대한 우선순위 정보를 포함한다. 이때 한 실시예에서, " 3" 은 가장 높은 우선순위를 나타내고, " 0" 은 가장 낮은 우선순위를 나타내어, 높은 우선순위의 요청이 낮은 우선순위의 요청을 피해갈 수 있다.

지속 로크(livelock) 상황을 방지하기 위해, 지속 로크 카운터 레지스터(156)는 높은 우선순위를 가지는 일련의 요청(또는 응답)이 얼마나 많이 낮은 우선순위의 요청(또는 응답)을 피해갈 수 있는 지에 관한 정보를 포함한다. 지속 로크 카운터 레지스터(156)는 높은 우선순위의 요청이 낮은 우선순위의 요청을 피해가는 상황에서만 작동한다. 실제로 적절한 큐에 낮은 우선순위의 요청(또는 응답)이 없을 경우, 지속 로크 카운터 레지스터(156)는 작동하지 않는다.

(57) 청구의 범위

청구항 1.

메모리 제어기에 의해 멀티뱅크형 메모리의 메모리 뱅크에 포함되는 메모리 페이지의 메모리 접근을 속도향상시키는 방법으로서, 이때 메모리 제어기는 다수의 페이지 레지스터를 포함하고, 각각의 페이지 레지스터는 다수의 메모리 뱅크 중 하나와 연계되며, 특정 뱅크와 연계된 페이지 레지스터는 특정 뱅크 번호, 특정 뱅크 번호 내에 위치하는 열린 페이지 어드레스, 열린 페이지 상태를 저장하도록 배열되고, 메모리 제어기는 다수의 페이지 레지스터 각각에 연결되는 조절가능한 비교기 유신을 추가로 포함하며, 상기 방법은,

- 유입 시스템 어드레스 요청을 수신하고, 이때 유입 시스템 어드레스 요청은 요청된 뱅크 번호와 요청된 페이지 번호를 포함하며,

- 비교기에 의해 요청된 뱅크 번호에 상응하는 페이지 레지스터를 위치시키고,

- 위치한 페이지 레지스터에 포함된 저장된 페이지 어드레스를 요청된 페이지 어드레스와 비교하며,

- 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치할 때, 요청된 뱅크 번호에 상응하는 메모리 뱅크의 요청 페이지에 접근하는, 이상의 단계를 포함하는 것을 특징으로 하는 방법.

청구항 2.

제 1 항에 있어서, 상기 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치하지 않을 때, 상기 방법은,

- 저장된 페이지 어드레스에 상응하는 메모리 페이지를 닫고,

- 요청된 페이지 어드레스에 상응하는 메모리 페이지를 열며,

- 요청된 메모리 페이지에 접근하는, 이상의 단계를 포함하는 것을 특징으로 하는 방법.

청구항 3.

제 2 항에 있어서, 메모리 장치는 SDRAM인 것을 특징으로 하는 방법.

청구항 4.

제 3 항에 있어서, 비교기는 조절가능한 비교기인 것을 특징으로 하는 방법.

청구항 5.

제 4 항에 있어서, 조절가능한 비교기는 멀티뱅크 메모리 장치의 작동 특성에 따르는 구조를 가지는 것을 특징으로 하는 방법.

청구항 6.

제 5 항에 있어서, 작동 특성은 메모리 제어기에 연결된 어드레스 공간 제어기 유닛에 저장되는 것을 특징으로 하는 방법.

청구항 7.

제 6 항에 있어서, 특정 뱅크 번호가 결정될 때, 특정 뱅크의 작동 특성은 조절가능한 비교기를 배열하는 데 사용되는 것을 특징으로 하는 방법.

청구항 8.

메모리 제어기에 의해 N 메모리 뱅크의 양을 가지는 멀티뱅크형 메모리의 메모리 뱅크에 포함되는 메모리 페이지로의 접근을 속도향상시키는 방법으로서, 이때 메모리 제어기는 M 페이지 레지스터의 양을 포함하고, M 페이지 레지스터의 양은 N 메모리 뱅크의 양보다 작아서, N 메모리 뱅크가 M 페이지 레지스터로 전부 표현될 수는 없으며, M 메모리 뱅크 중 선택된 뱅크에 상응하는 선택된 뱅크 번호, 선택된 뱅크 내에 위치하는 열린 페이지 어드레스, 그리고 열린 페이지 상태를 저장하도록 한 개의 페이지 레지스터가 배열되고, 또한 메모리 제어기는 다수의 페이지 레지스터 각각에 연결되는 조절가능한 비교기 유닛을 추가로 포함하며, 상기 메모리 제어기는 1과 M을 포함한 1과 M 사이의 정수를 임의로 발생시키도록 배열되는 임의 페이지 레지스터 번호 발생기를 추가로 포함하고, 상기 방법은,

- 유입 시스템 어드레스 요청을 수신하고, 이때 유입 시스템 어드레스 요청은 요청된 뱅크 번호와 요청된 페이지 번호를 포함하며,

- 비교기에 의해 요청된 뱅크 번호에 상응하는 페이지 레지스터를 위치시키며,

- 위치한 페이지 레지스터에 포함되는 저장된 페이지 어드레스를 요청된 페이지 어드레스와 비교하고,

- 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치할 때, 요청된 뱅크 번호에 상응하는 메모리 뱅크의 요청 페이지에 접근하며,

- 요청된 뱅크 및 저장된 페이지 어드레스가 일치하지 않을 경우, 임의 페이지 레지스터 번호 발생기에 의해 번호가 부여되는 뱅크의 페이지 레지스터로부터 페이지 어드레스를 이용하여 저장된 페이지를 닫고,

- 유입 시스템 어드레스로부터 페이지 어드레스를 이용하여 새 페이지를 열며,

- 요청된 뱅크에 접근하는, 이상의 단계를 포함하는 것을 특징으로 하는 방법.

청구항 9.

제 8 항에 있어서, 메모리 장치는 SDRAM인 것을 특징으로 하는 방법.

청구항 10.

제 9 항에 있어서, 비교기는 조절가능한 비교기인 것을 특징으로 하는 방법.

청구항 11.

제 10 항에 있어서, 조절가능한 비교기는 멀티뱅크 메모리 장치의 작동 특성에 따르는 구조를 가지는 것을 특징으로 하는 방법.

청구항 12.

제 11 항에 있어서, 상기 작동 특성은 메모리 제어기에 연결되는 어드레스 제어기 유닛에 저장되는 것을 특징으로 하는 방법.

청구항 13.

제 12 항에 있어서, 특정 뱅크 번호가 결정될 때, 특정 뱅크의 작동 특성은 조절가능한 비교기 배열에 이용되는 것을 특징으로 하는 방법.

청구항 14.

메모리 제어기에 의해 N 메모리 뱅크의 양을 가지는 멀티뱅크형 메모리의 메모리 뱅크에 포함되는 메모리 페이지로의 메모리 접근을 속도향상시키는 방법으로서, 이때 메모리 제어기는 M 페이지 레지스터의 양을 포함하고, M 페이지 레지스터의 양은 N 메모리 뱅크의 양보다 작아서, N 메모리 뱅크가 M 페이지 레지스터로 전부 표현될 수는 없으며, M 메모리 뱅크 중 선택된 뱅크에 상응하는 선택된 뱅크 번호, 선택된 뱅크 내에 위치하는 열린 페이지 어드레스, 그리고 열린 페이지 상태를 저장하도록 한 개의 페이지 레지스터가 배열되고, 또한 메모리 제어기는 다수의 페이지 레지스터 각각에 연결되는 조절가능한 비교기 유닛을 추가로 포함하며, 상기 메모리 제어기는 가장 오래된 사용되지 않은 페이지 레지스터의 번호를 임의로 발생시키도록 최근에 거의 사용되지 않은 발생기를 추가로 포함하고, 상기 방법은,

- 유입 시스템 어드레스 요청을 수신하고, 이때 유입 시스템 어드레스 요청은 요청된 뱅크 번호와 요청된 페이지 번호를 포함하며,

- 비교기에 의해 요청된 뱅크 번호에 상응하는 페이지 레지스터를 위치시키며,

- 위치한 페이지 레지스터에 포함되는 저장된 페이지 어드레스를 요청된 페이지 어드레스와 비교하고,

- 저장된 페이지 어드레스가 요청된 메모리 뱅크에 대한 요청된 페이지 어드레스와 일치할 때, 요청된 뱅크 번호에 상응하는 메모리 뱅크의 요청 페이지에 접근하며,

- 요청된 뱅크 및 저장된 페이지 어드레스가 일치하지 않을 경우, 최근에 거의 사용되지 않는 페이지 레지스터 번호 발생기에 의해 번호가 부여되는 뱅크의 페이지 레지스터로부터 페이지 어드레스를 이용하여 저장된 페이지를 닫고,

- 유입 시스템 어드레스로부터 페이지 어드레스를 이용하여 새 페이지를 열며,

- 요청된 뱅크에 접근하는, 이상의 단계를 포함하는 것을 특징으로 하는 방법.

청구항 15.

제 14 항에 있어서, 메모리 장치는 SDRAM인 것을 특징으로 하는 방법.

청구항 16.

제 15 항에 있어서, 비교기는 조절가능한 비교기인 것을 특징으로 하는 방법.

청구항 17.

제 16 항에 있어서, 조절가능한 비교기는 멀티뱅크 메모리 장치의 작동 특성에 따르는 구조를 가지는 것을 특징으로 하는 방법.

청구항 18.

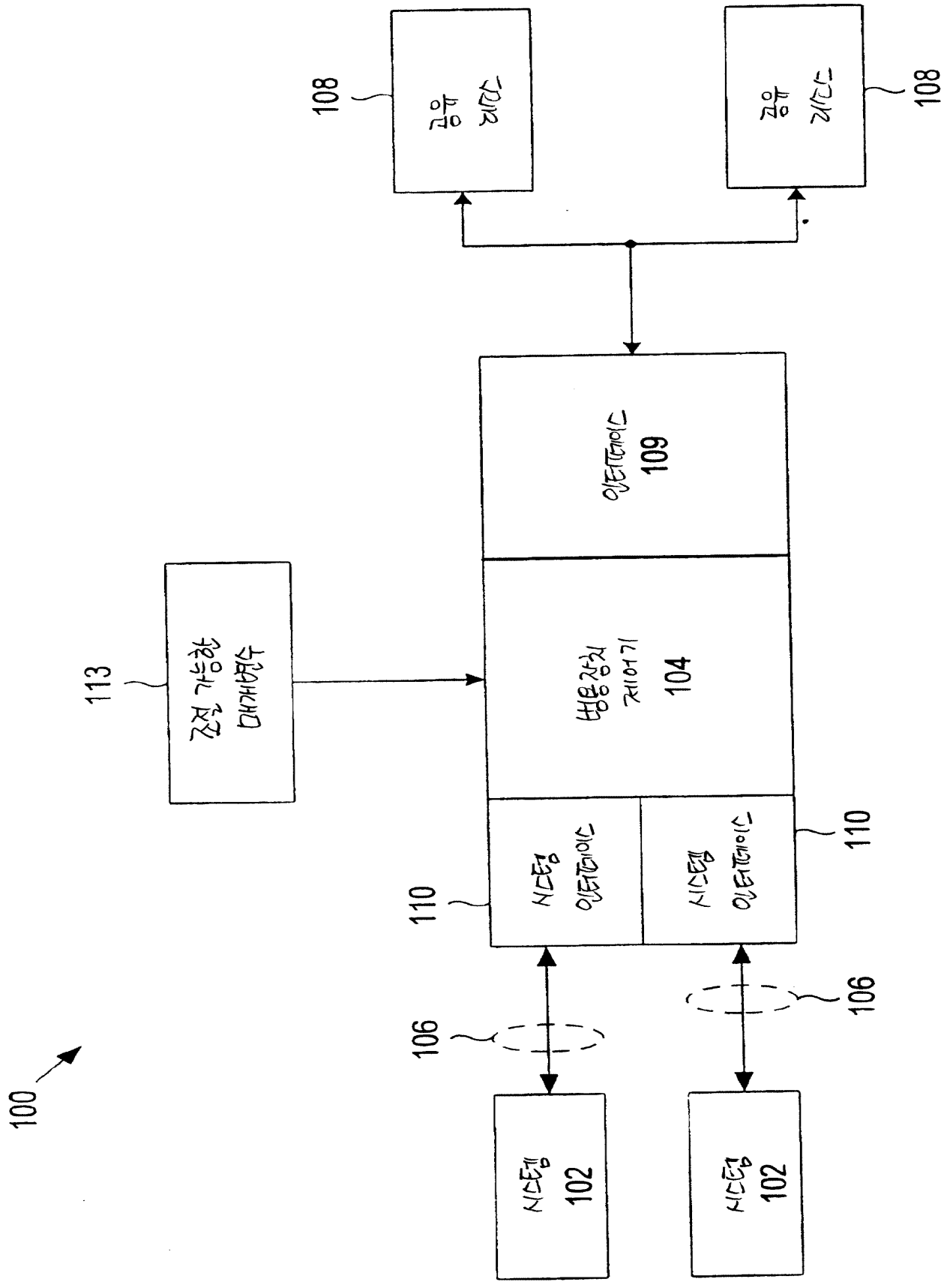
제 17 항에 있어서, 상기 작동 특성은 메모리 제이기에 연결되는 어드레스 공간 제어기 유닛에 저장되는 것을 특징으로 하는 방법.

청구항 19.

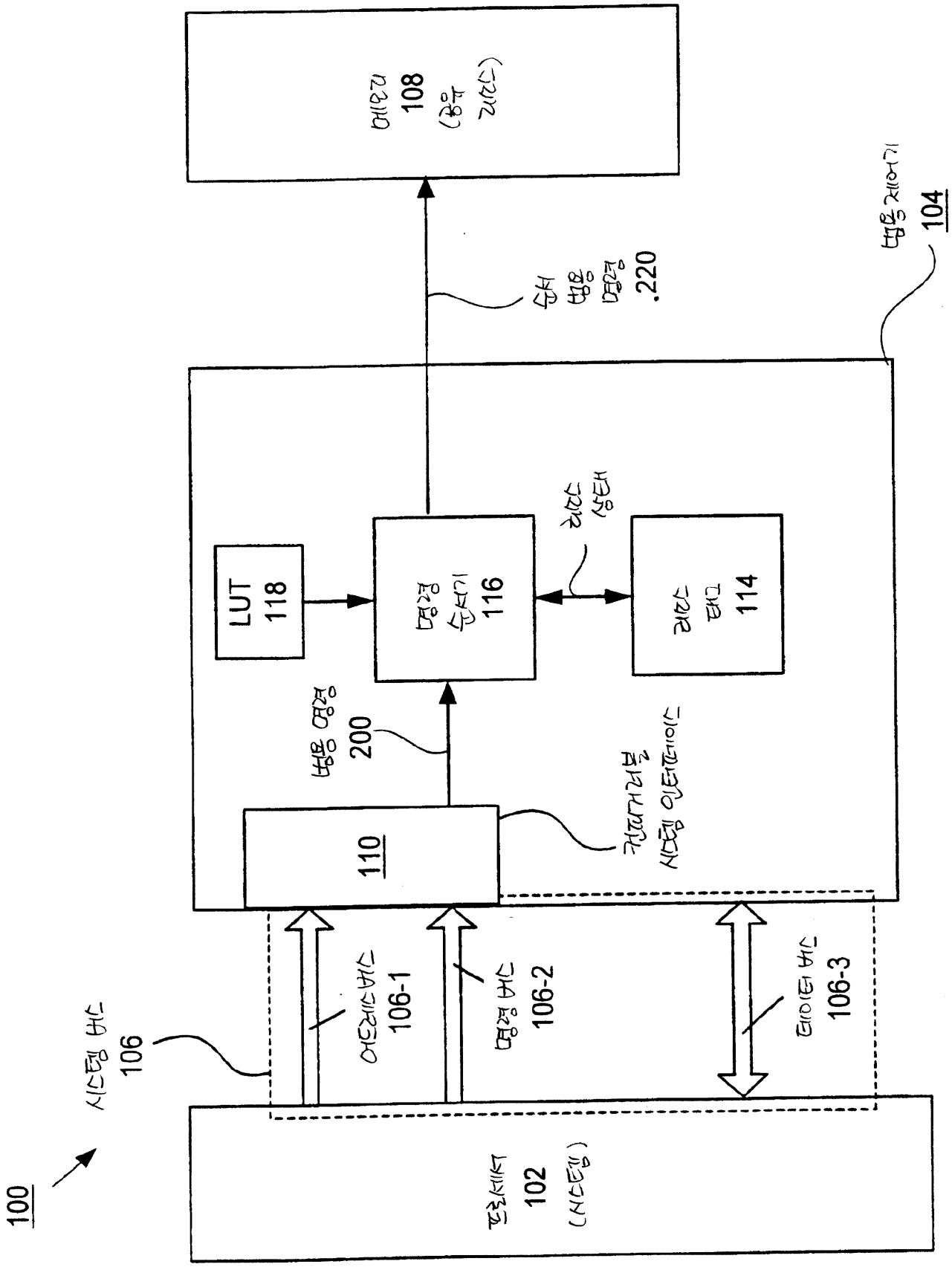
제 18 항에 있어서, 특정 뱅크 번호가 결정될 때, 특정 뱅크의 작동 특성은 조절가능한 비교기의 배열에 사용되는 것을 특징으로 하는 방법.

도면

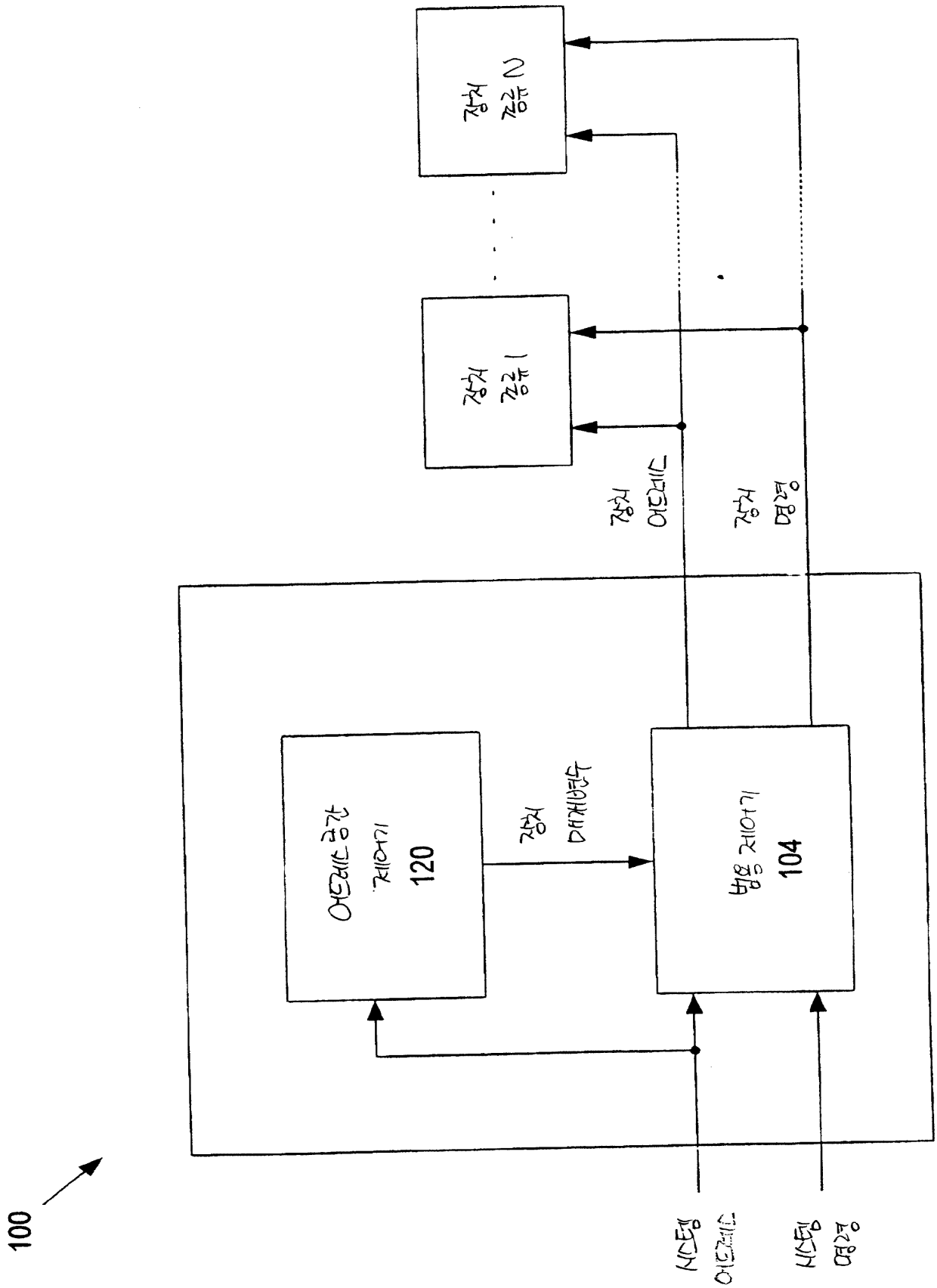
도면 1a


100
113
조정 가능한 대체부하
108
108
인터페이스 109
범용장치 제어기 104
110
110
106
106
시스템 102
시스템 102

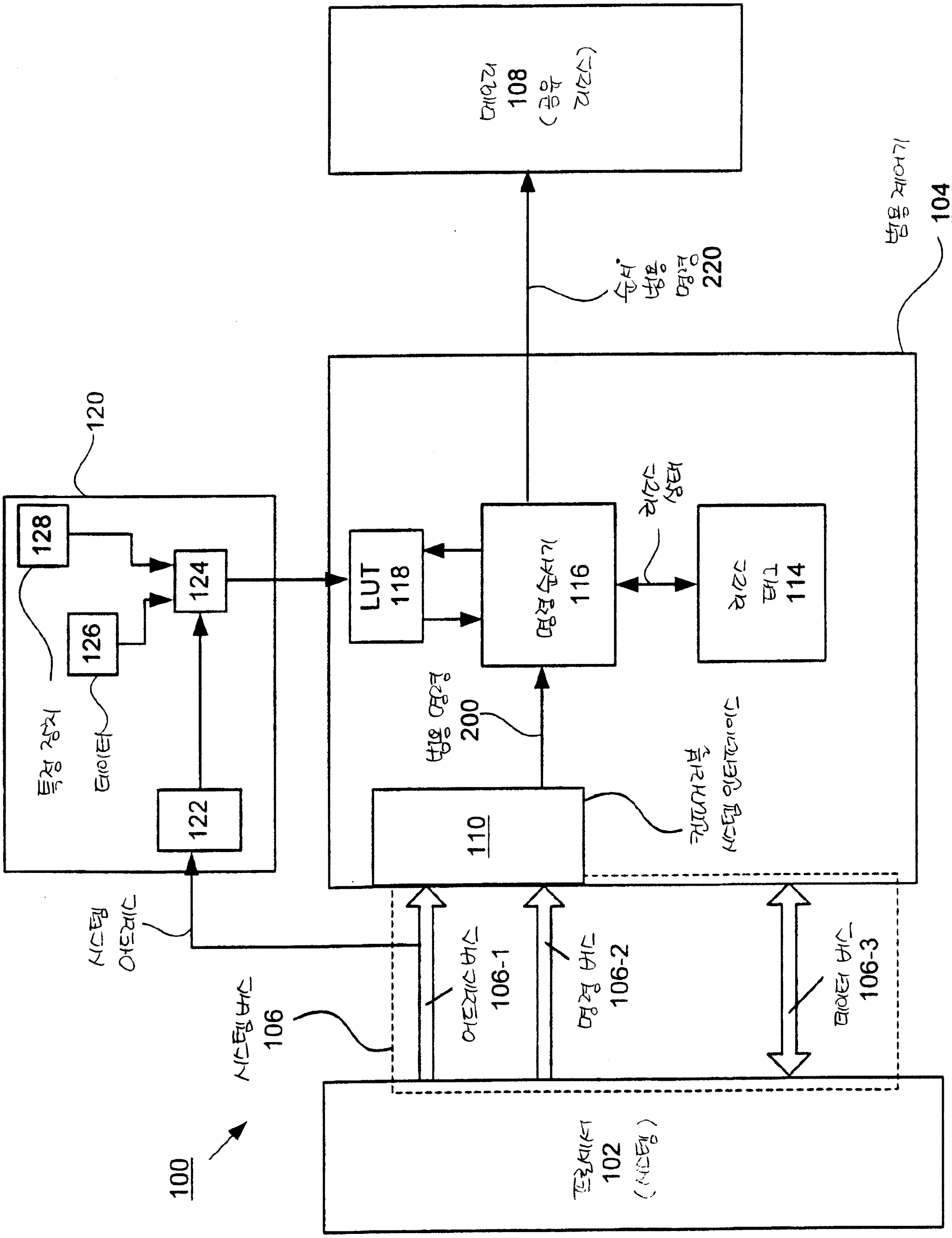
도면 1b


100
시스템 버스
106
프로세서
102
(시스템)
어드레스 버스
106-1
명령 버스
106-2
데이터 버스
106-3
110
컨트롤러 및 시스템 인터페이스
변용 명령
200
LUT
118
명령 순서기
116
리소스 상태
리소스 태그
114
순서 변용 명령
220
메모리
108
(동적 RAM)
변용 제어기
104

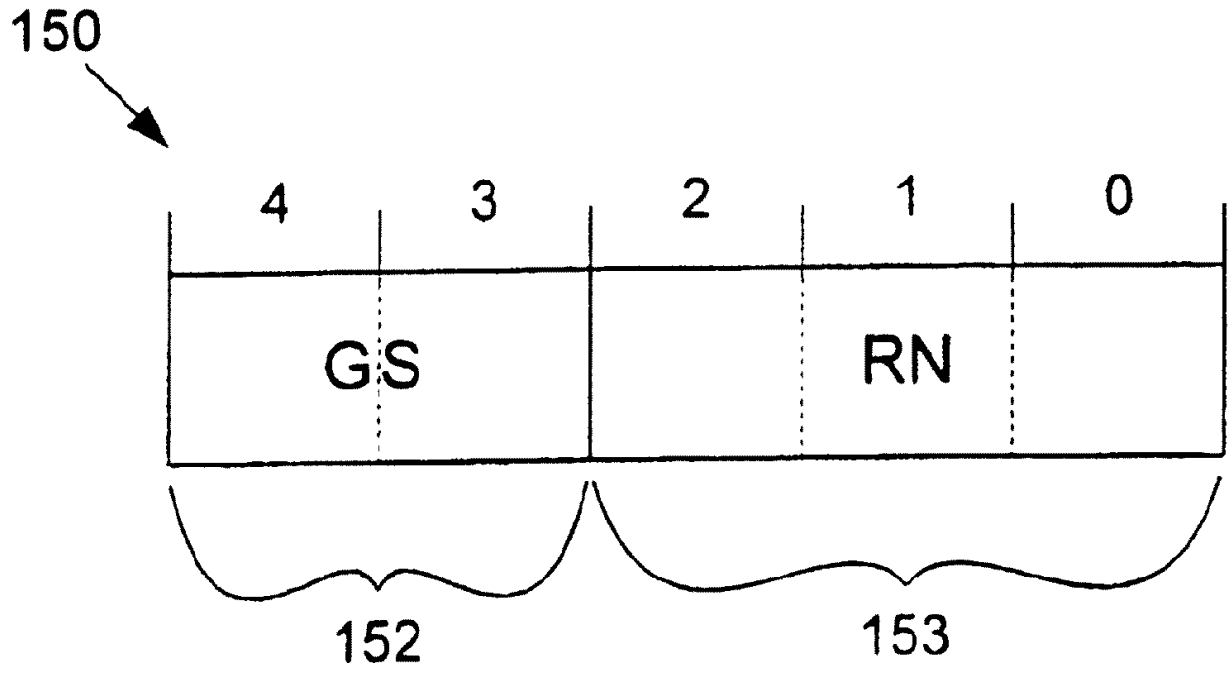
도면 1c


100
장치 갯수 N
장치 갯수 1
장치 어드레스
장치 명령
어드레스 공간 제어기 120
장치 데이터
범용 제어기 104
시스템 어드레스
시스템 명령

도면 1d

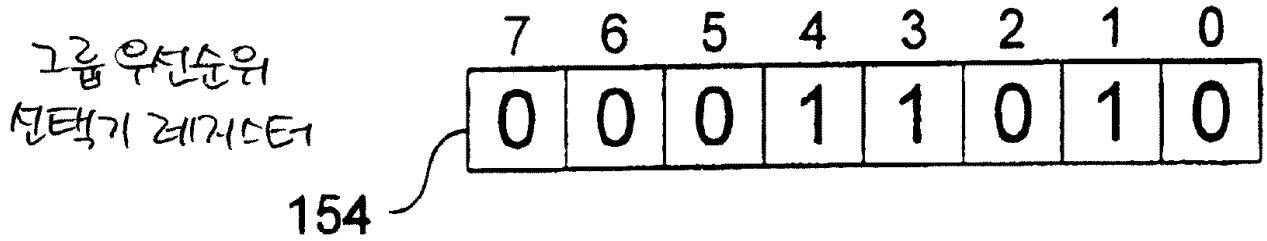

메모리
108
(공유 자원)
범용 제어기
104
순서 범용 명령
220
120
128
124
126
122
특정 장치
데이터
LUT
118
명령 순서기
116
리소스 상태
리소스 태그
114
범용 명령
200
110
컨피겨러블 시스템 인터페이스
시스템 어드레스
시스템 버스
106
어드레스 버스
106-1
명령 버스
106-2
데이터 버스
106-3
프로세서
102
(시스템)
100

도면 1e1

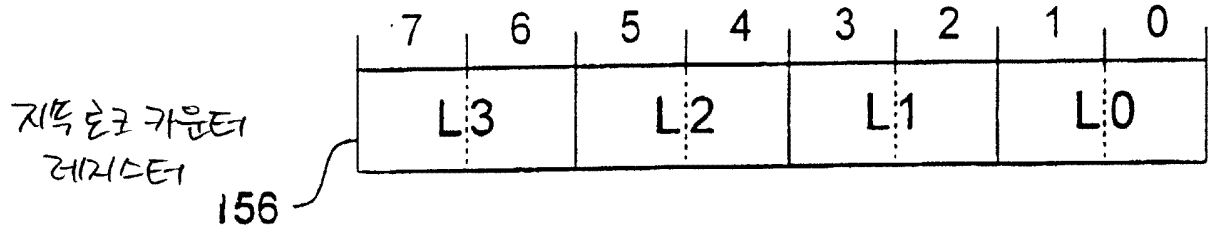


RN - 요청 번호
GS - 그룹 선택기

도면 1e2

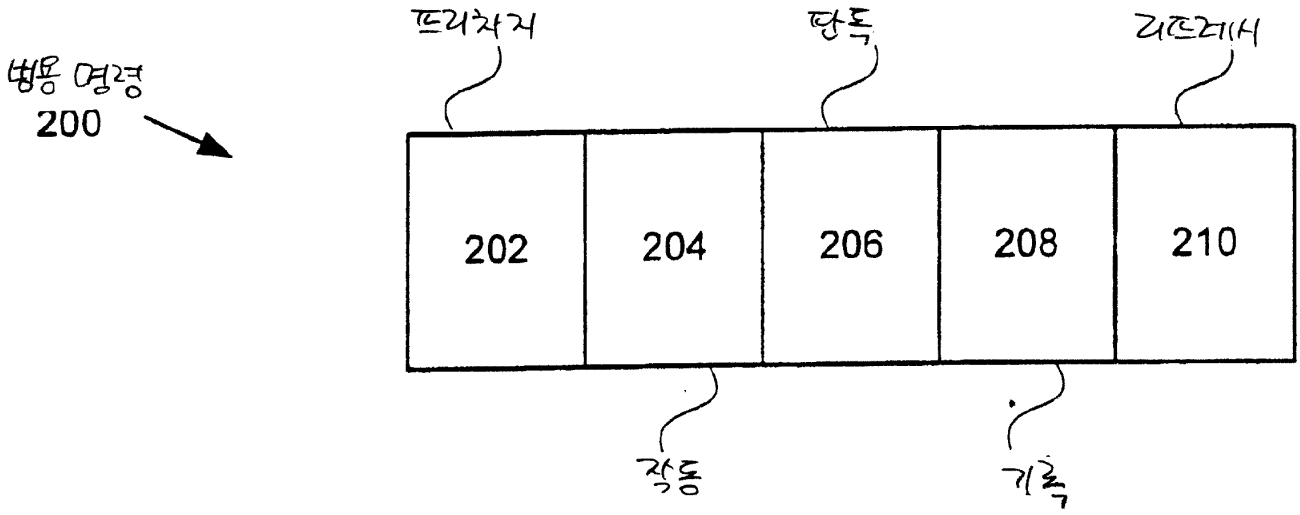


도면 1e3

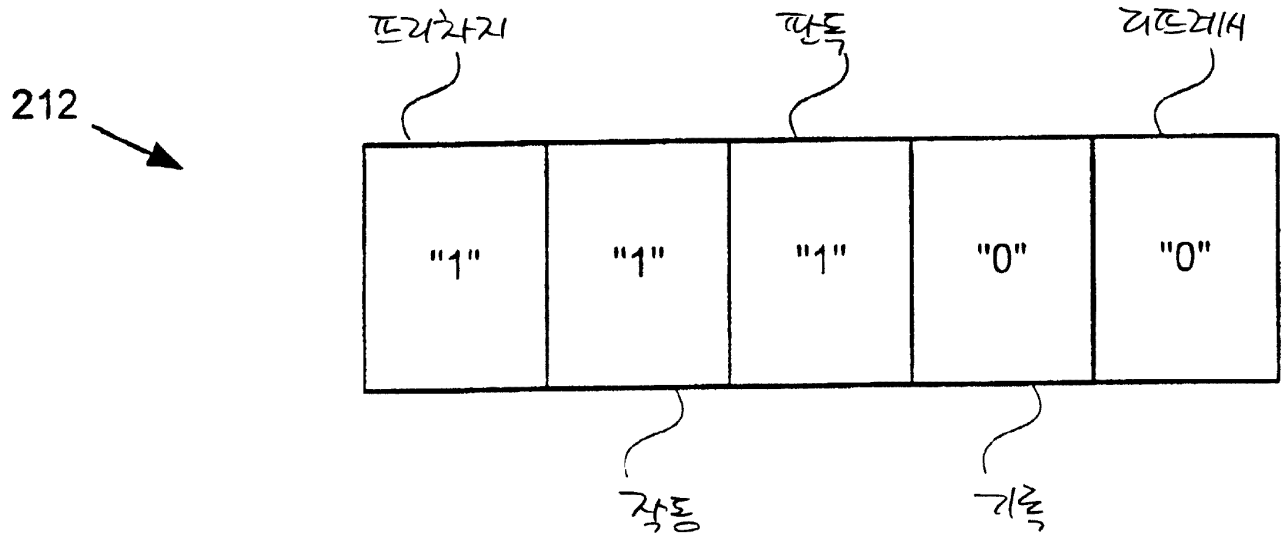


L3...L0 - 우선순위 "00"에서 "11"까지의 지속 토크

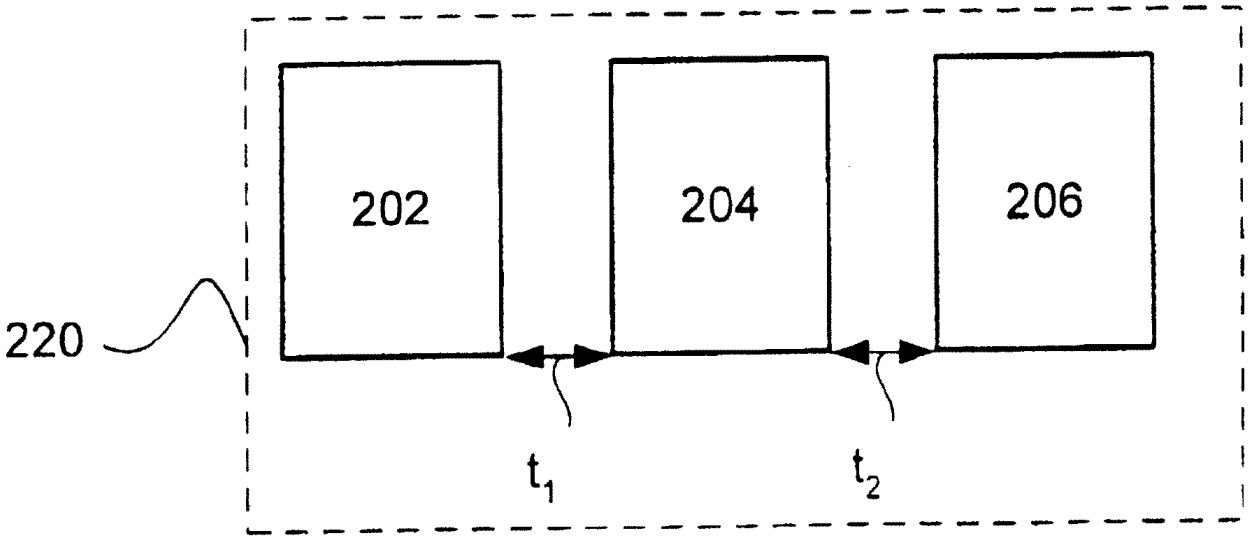
도면 2a


범용 명령
200
프리차지
판독
리프레시
202
204
206
208
210
작동
기록

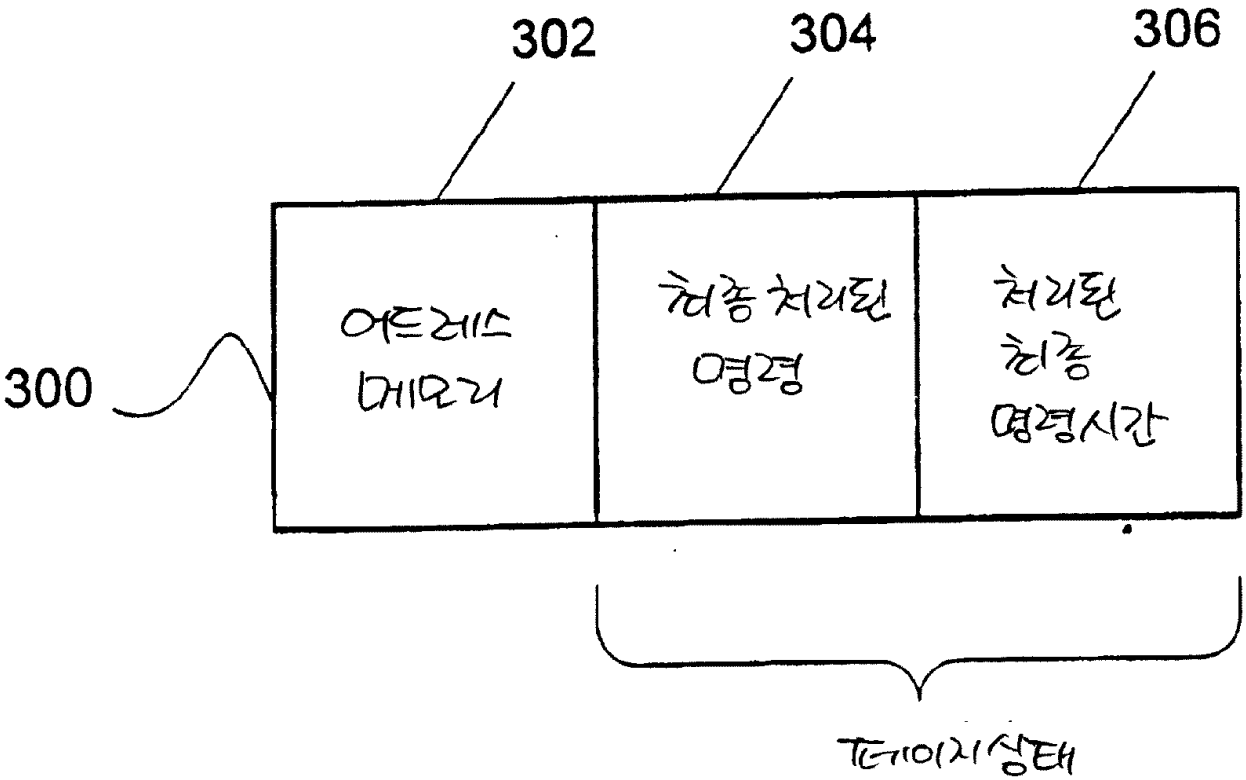
도면 2b


212
프리차지
판독
리프레시
"1"
"1"
"1"
"0"
"0"
작동
기록

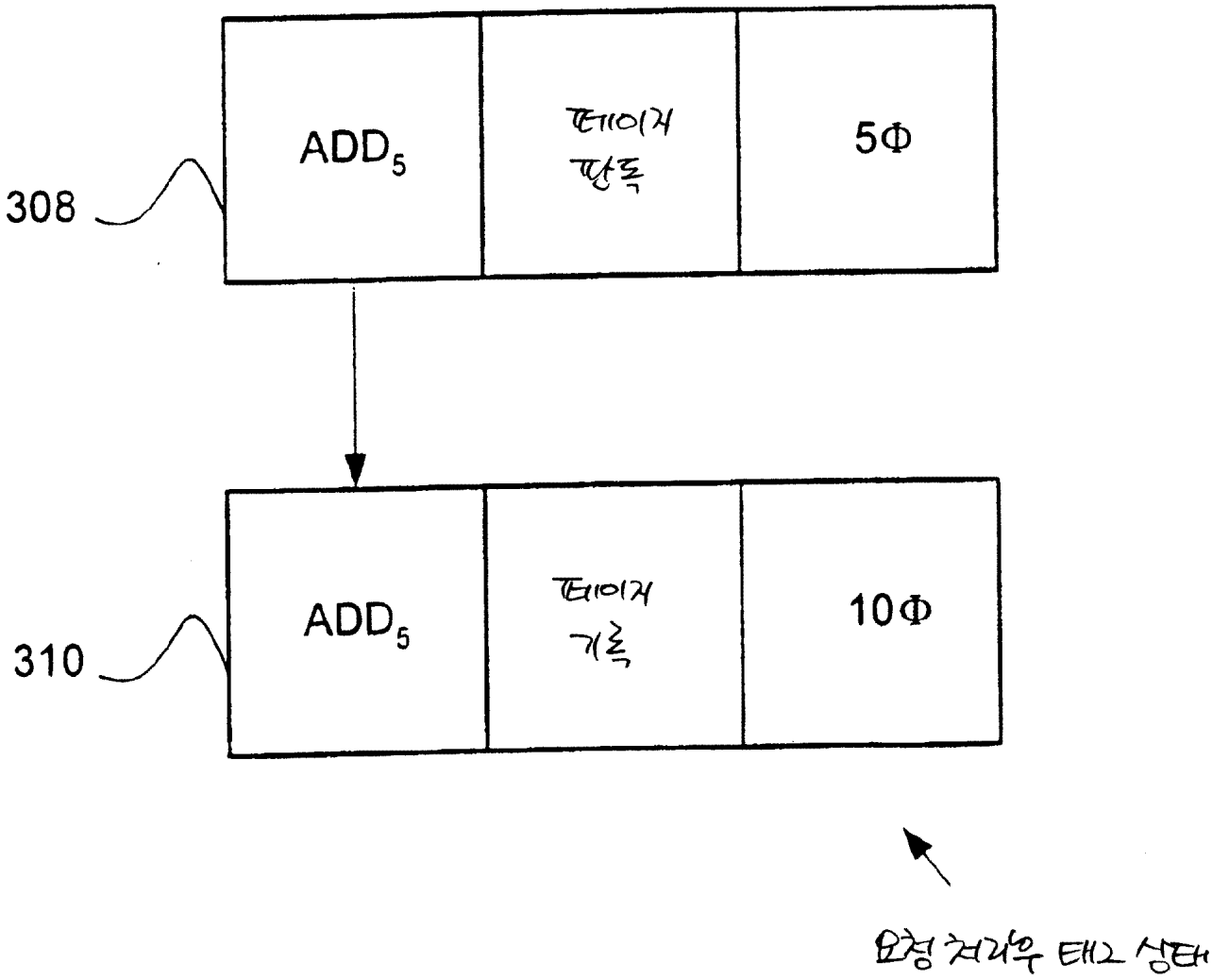
도면 2c


202
204
206
220
t_1
t_2

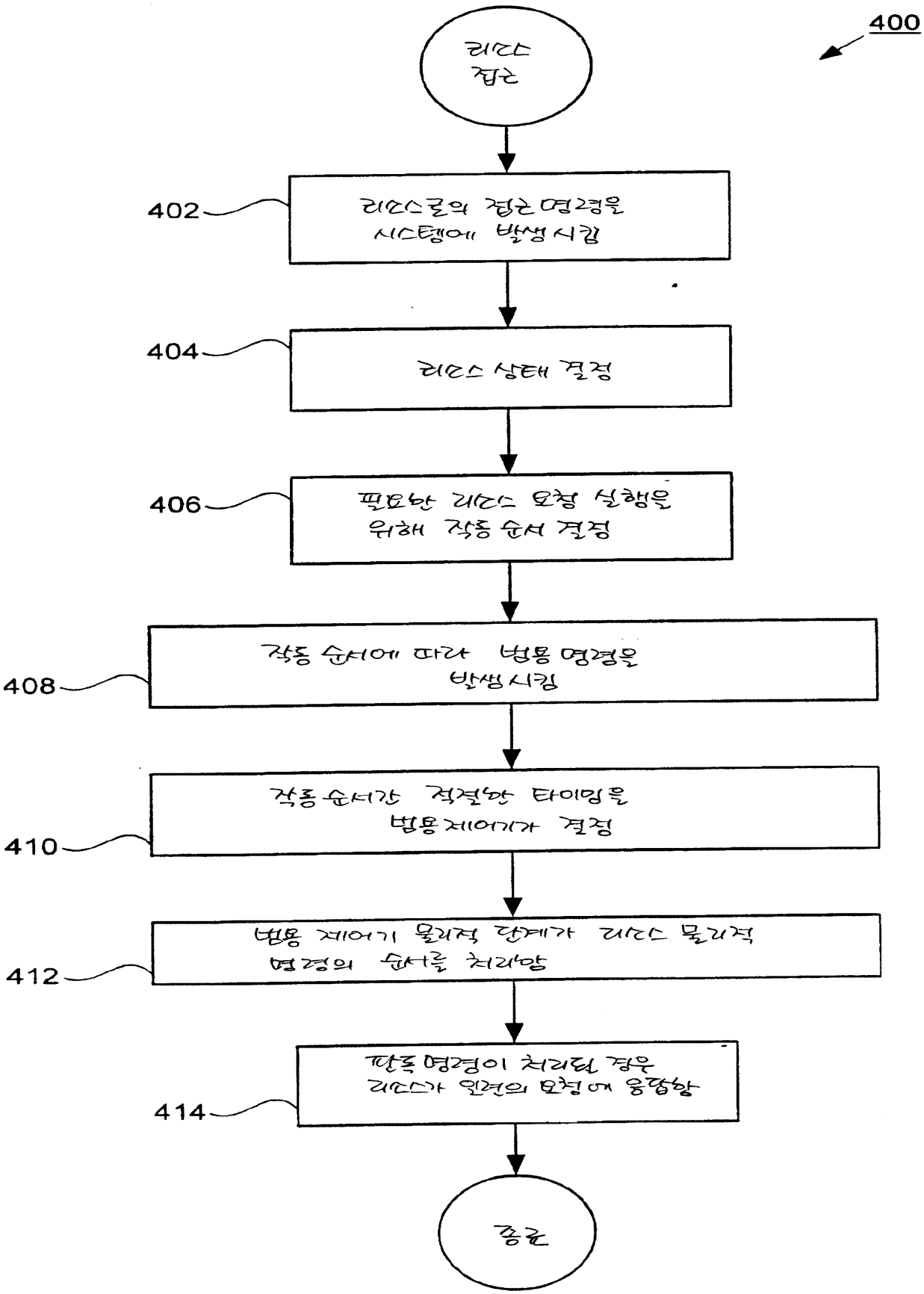
도면 3a


302
304
306
300
어드레스 레지스터
처음 처리된 명령
처리된 최종 명령시간
데이지상태

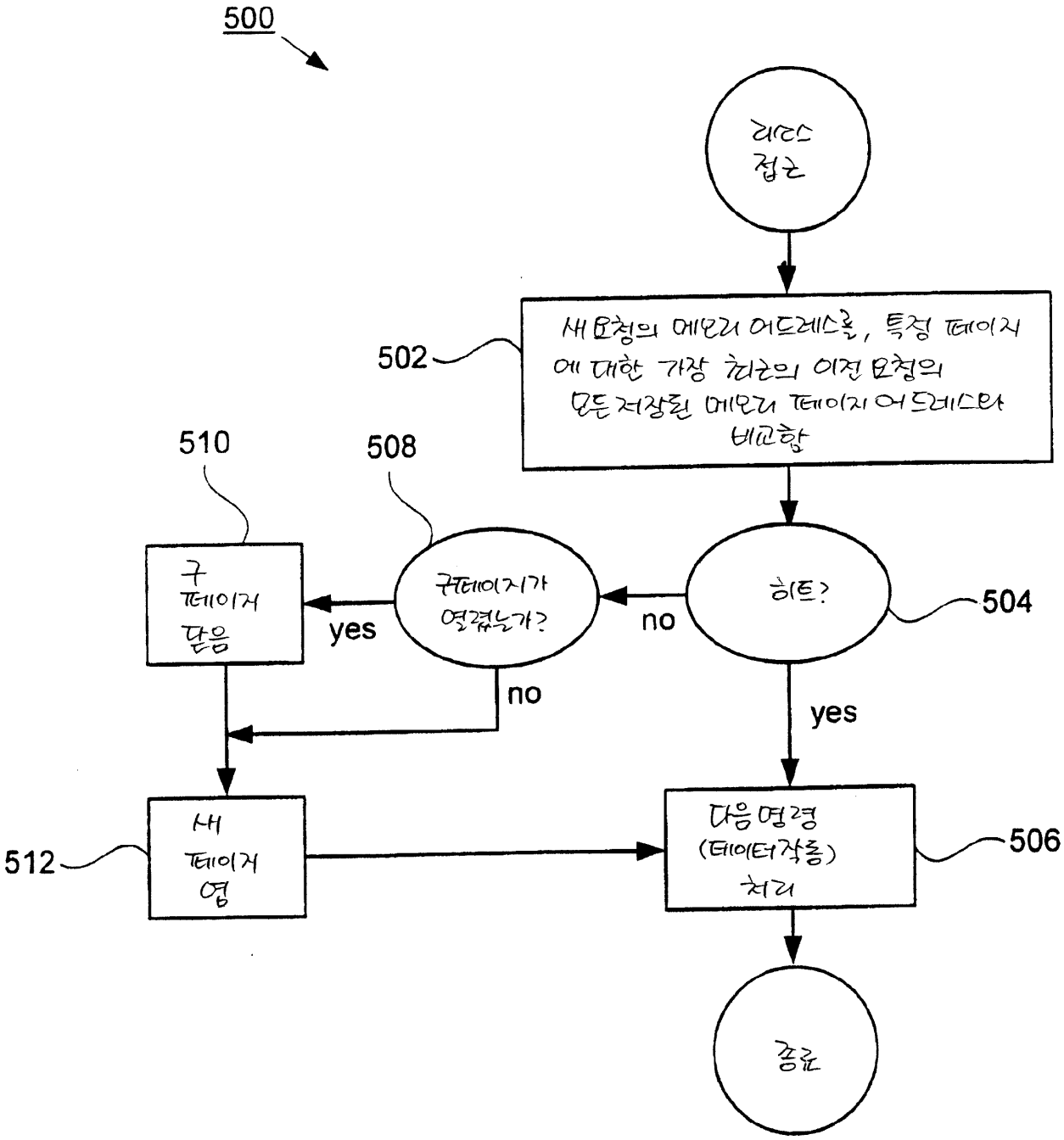
도면 3b



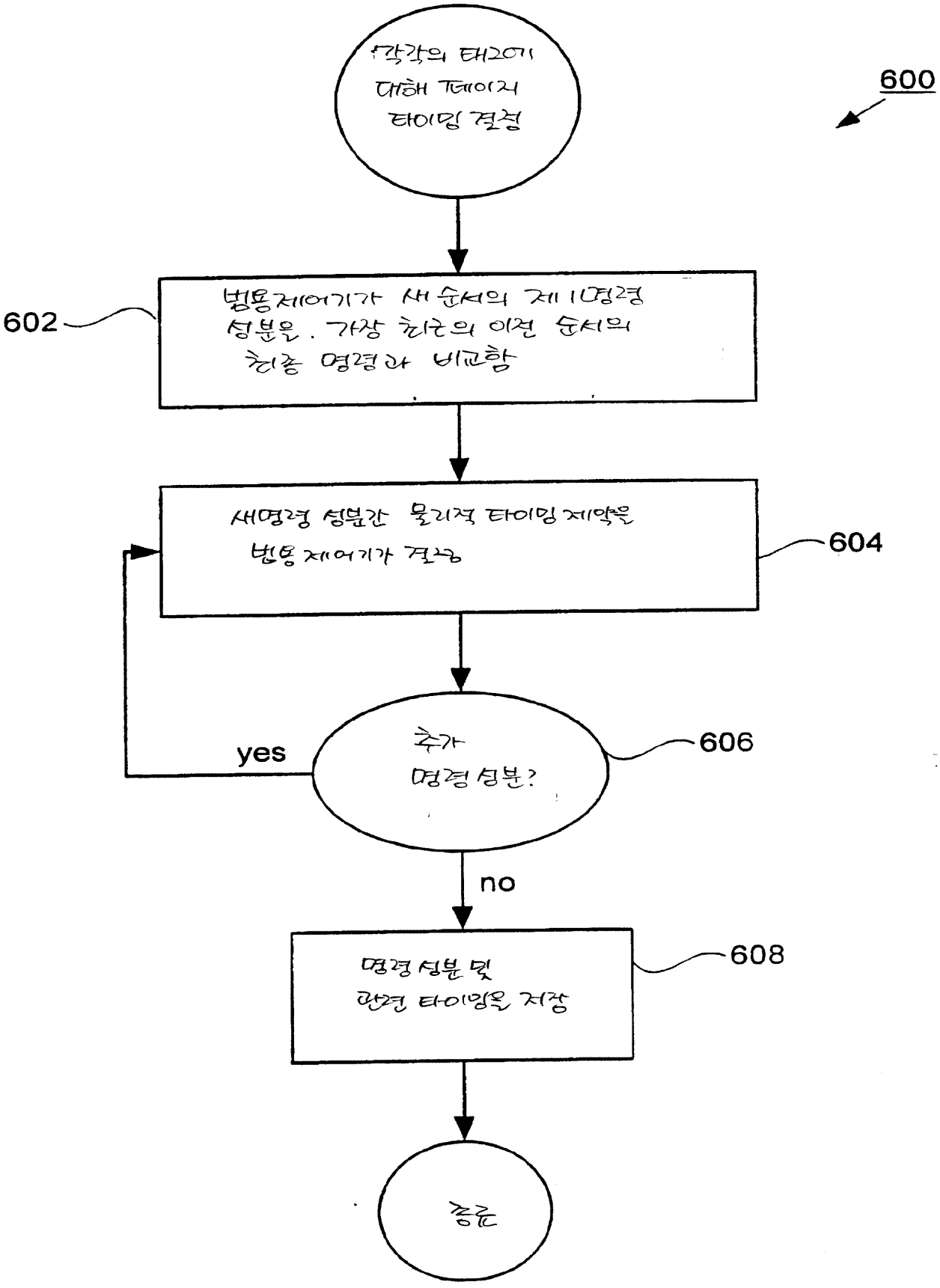
도면 4


400
리소스 접근
402
리소스로의 접근 명령을 시스템에 발생시킴
404
리소스 상태 결정
406
필요한 리소스 요청 실행을 위해 작동 순서 결정
408
작동 순서에 따라 범용 명령을 발생시킴
410
작동 순서간 적절한 타이밍을 범용 제어기가 결정
412
범용 제어기 물리적 단계가 리소스 물리적 명령의 순서를 처리함
414
판독 명령이 처리된 경우 리소스가 외부의 요청에 응답함
종료

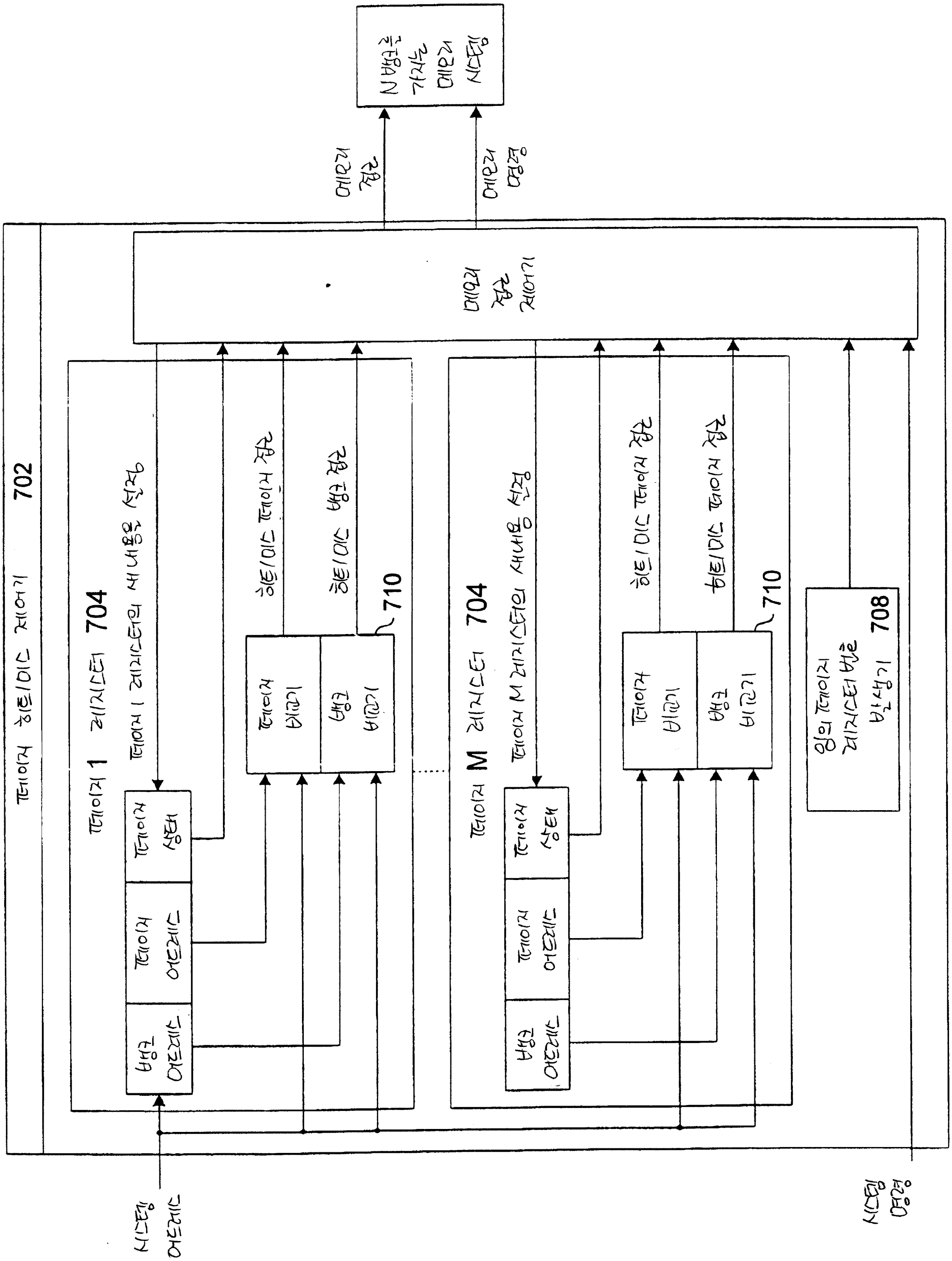
도면 5


500
리턴 접근
502
새 요청의 메모리 어드레스를, 특정 페이지에 대한 가장 최근의 이전 요청의 모든 저장된 메모리 페이지 어드레스와 비교함
히트?
504
no
yes
구 페이지가 열렸는가?
508
yes
no
510
구 페이지 닫음
512
새 페이지 열
다음 명령 (데이터 작동) 처리
506
종료

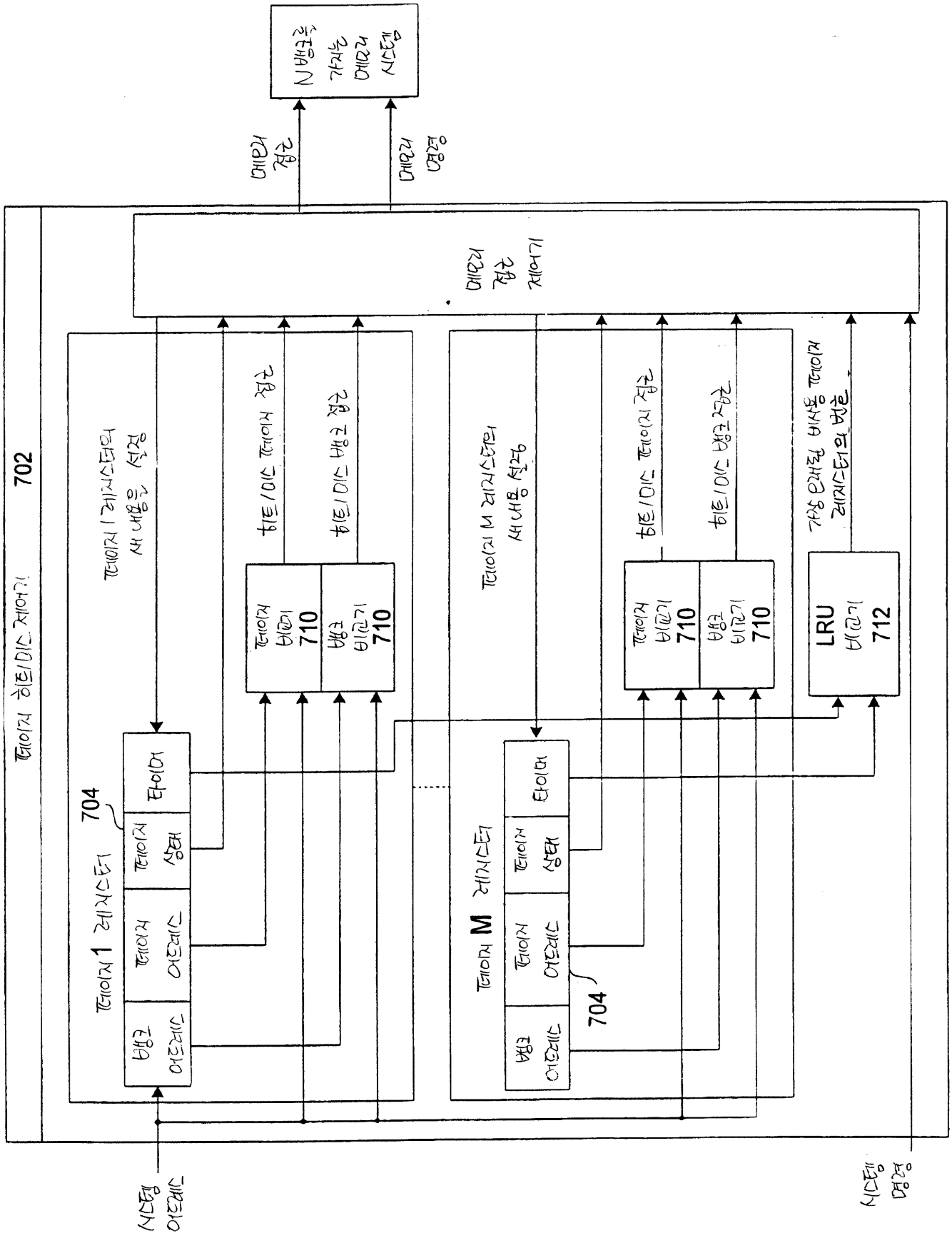
도면 6


각각의 태그에 대해 페이지 타이밍 결정
600
범용제어기가 새 순서의 제 1명령 성분을, 가장 최근의 이전 순서의 최종 명령과 비교함
602
새명령 성분간 물리적 타이밍 제약은 범용 제어기가 결정
604
추가 명령 성분?
606
yes
no
명령 성분 및 관련 타이밍을 저장
608
종료

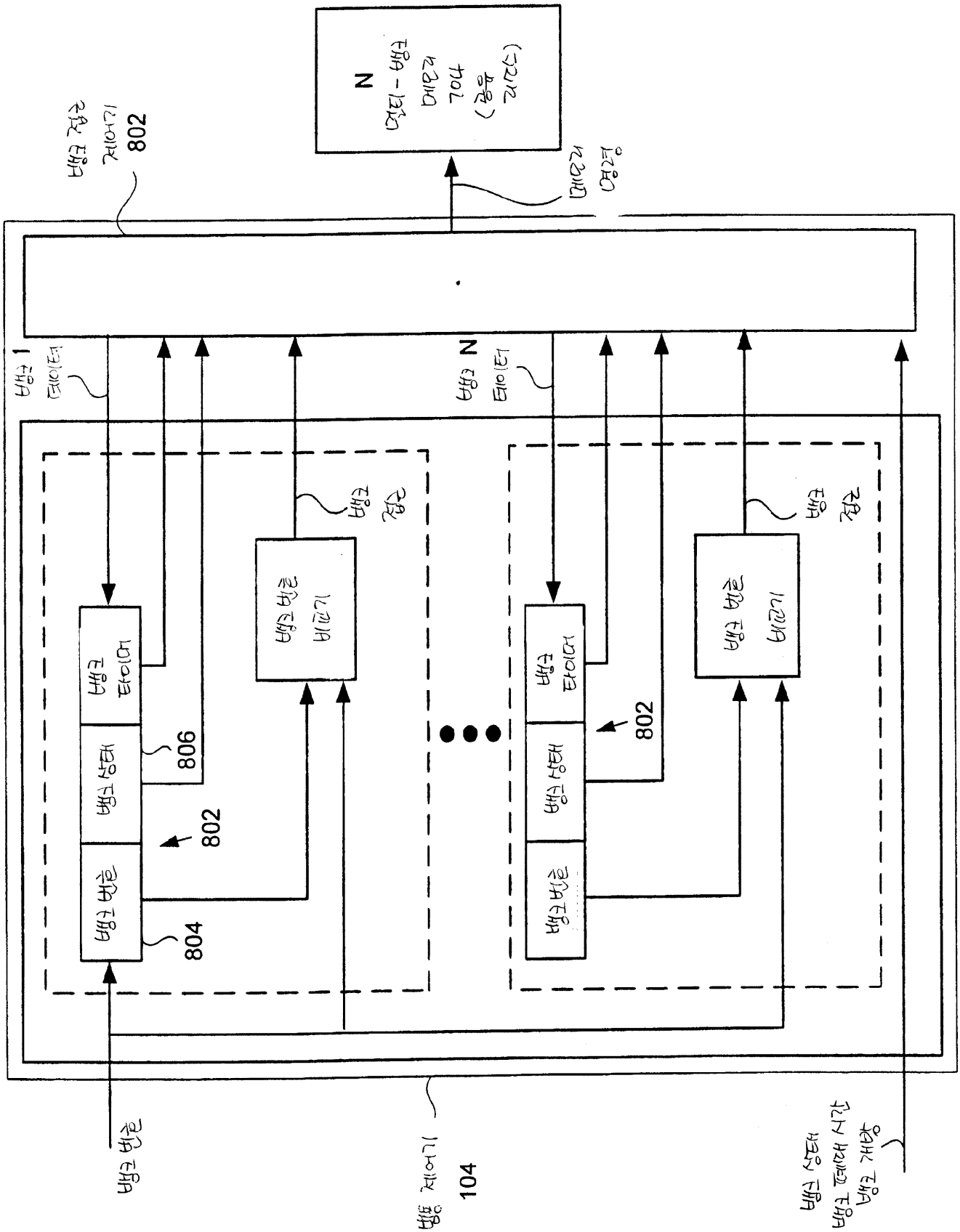
도면 7a


N 뱅크 가지는 메모리 시스템
메모리 상태
메모리 명령
메모리 상태 제어기
페이지 히트/미스 제어기 702
페이지 1 레지스터 704
페이지 1 레지스터의 새내용 설정
히트/미스 페이지 설정
히트/미스 뱅크 설정
710
페이지 비교기
뱅크 비교기
페이지 상태
페이지 어드레스
뱅크 어드레스
페이지 M 레지스터 704
페이지 M 레지스터의 새내용 설정
히트/미스 페이지 설정
히트/미스 페이지 설정
710
페이지 비교기
뱅크 비교기
페이지 상태
페이지 어드레스
뱅크 어드레스
임의 페이지 레지스터 번호 발생기 708
시스템 어드레스
시스템 명령

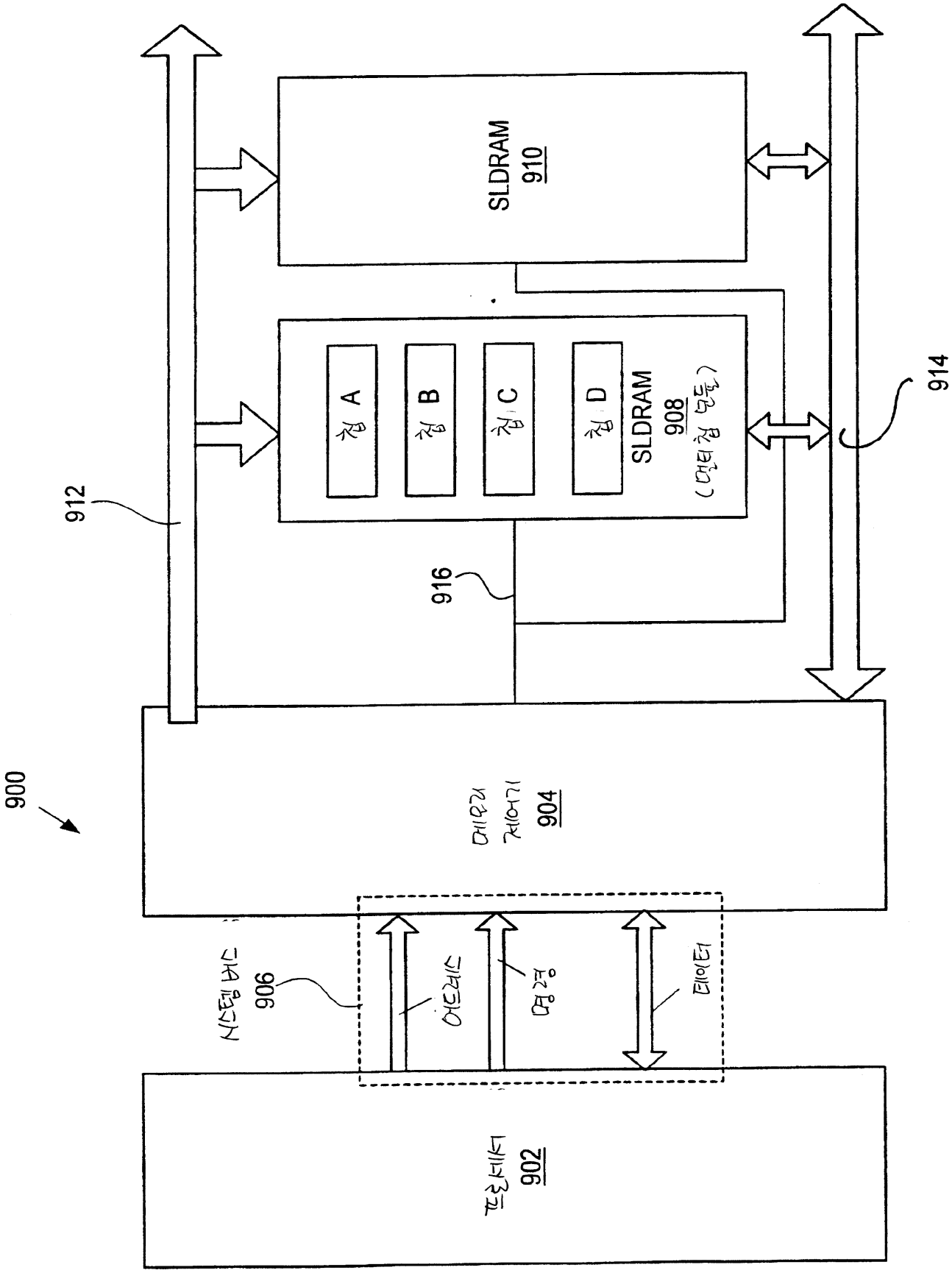
도면 7b


702
페이지 히트/미스 제어기
페이지 1 레지스터
704
페이지 M 레지스터
704
타이머
페이지 상태
페이지 어드레스
변경 어드레스
페이지 레지스터 새 내용을 선정
히트/미스 페이지 검출
히트/미스 변경 검출
페이지 비교기 710
변경 비교기 710
LRU 비교기 712
가장 오래된 비사용 페이지 레지스터의 번호
메모리 접근 제어기
메모리 접근
메모리 응답
시스템 메모리 가지기
시스템 어드레스
응답

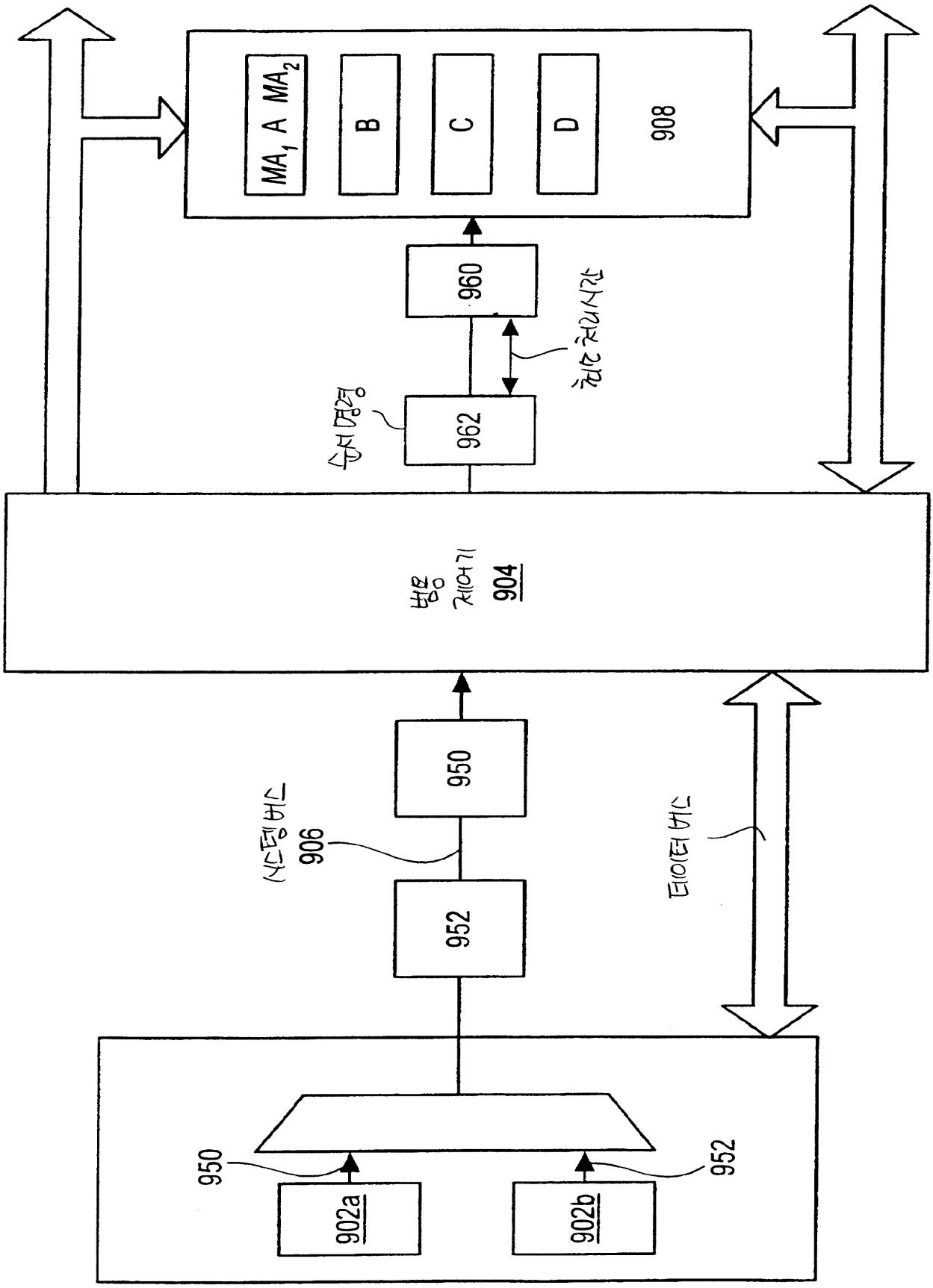
도면 8



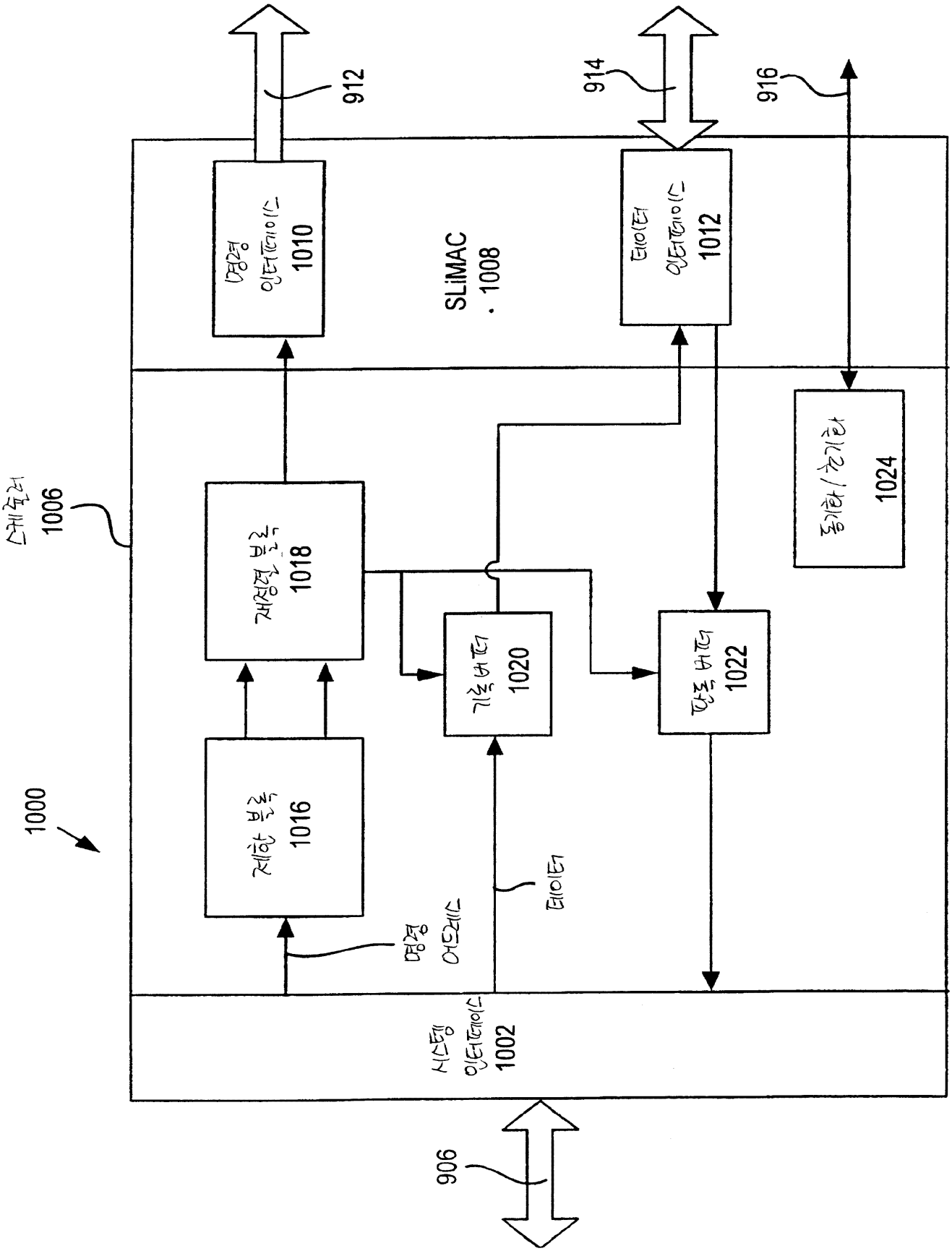
도면 9a


900
912
914
916
SLDRAM
910
SLDRAM
908
(멀티칩 모듈)
칩 A
칩 B
칩 C
칩 D
메모리 제어기
904
인터페이스
906
어드레스
명령
데이터
프로세서
902

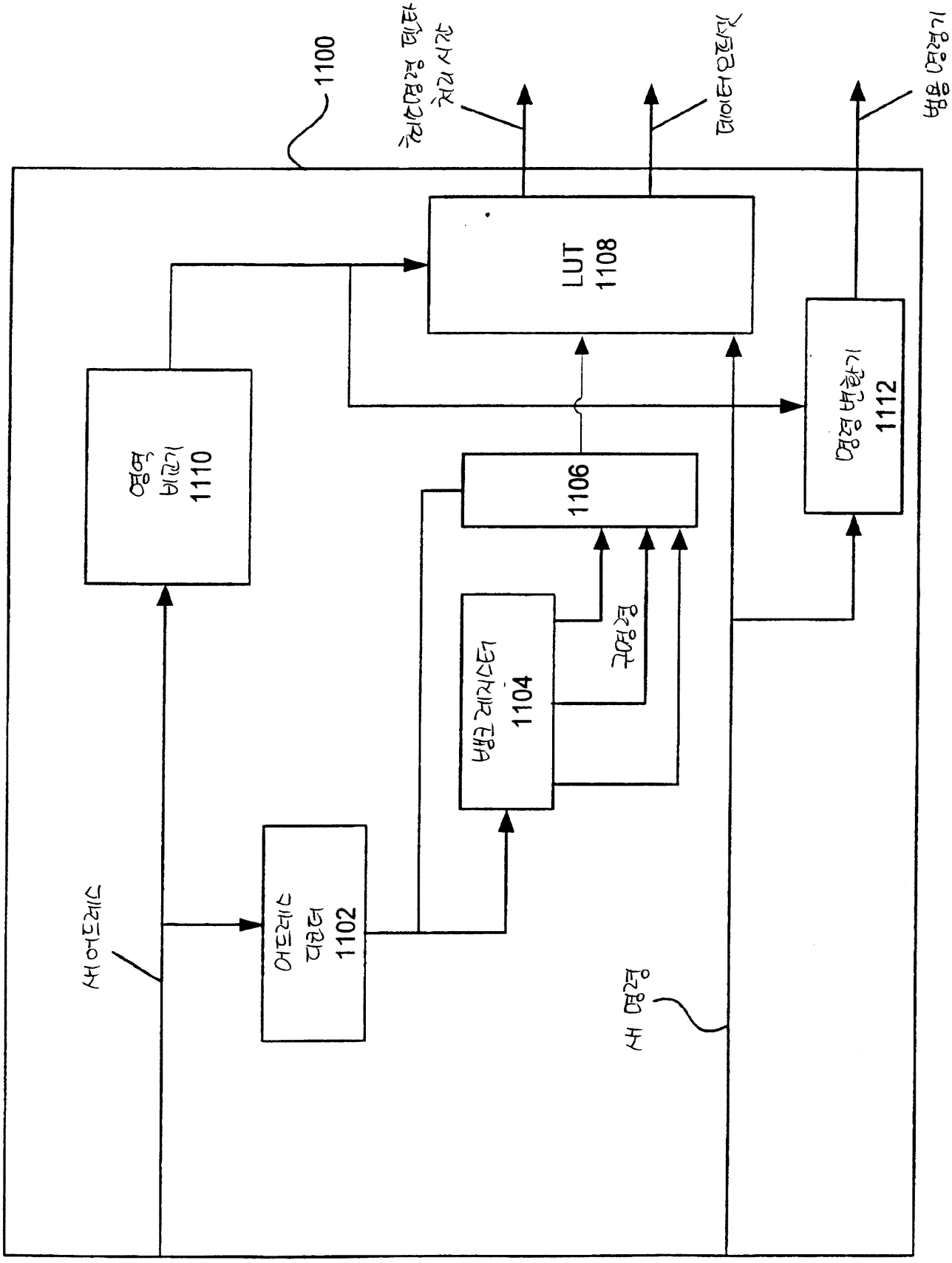
도면 9b


MA1 A MA2
B
C
D
908
960
처리시간
수신명령
962
응용 제어기 904
950
시스템 버스
906
952
데이터 버스
950
902a
902b
952

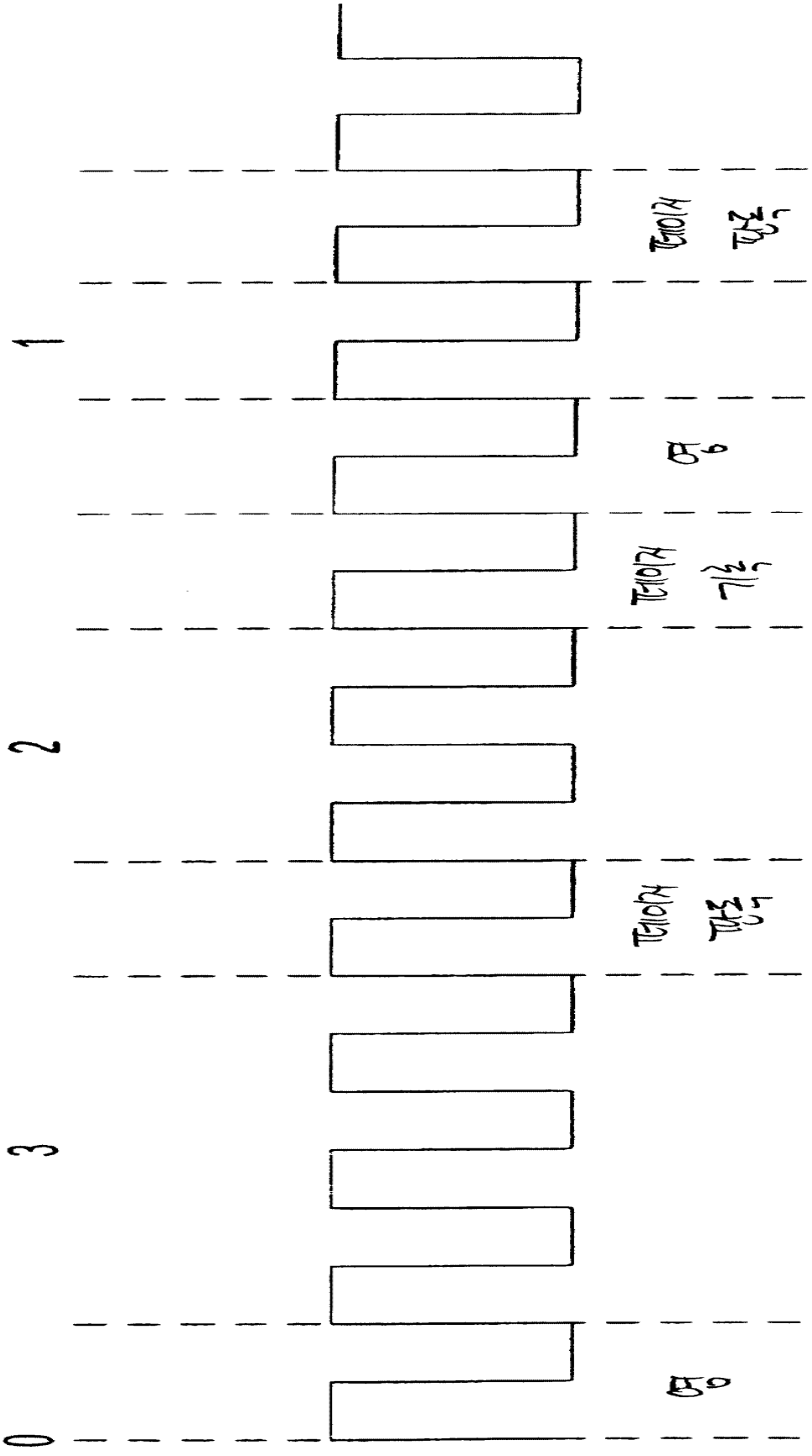
도면 10


912
914
916
명령 인터페이스 1010
SLIMAC 1008
데이터 인터페이스 1012
산술 논리 장치 1006
레지스터 뱅크 1018
기록 버퍼 1020
판독 버퍼 1022
타이머/계수기 1024
명령 캐시 1016
명령 어드레스
데이터
시스템 인터페이스 1002
906
1000

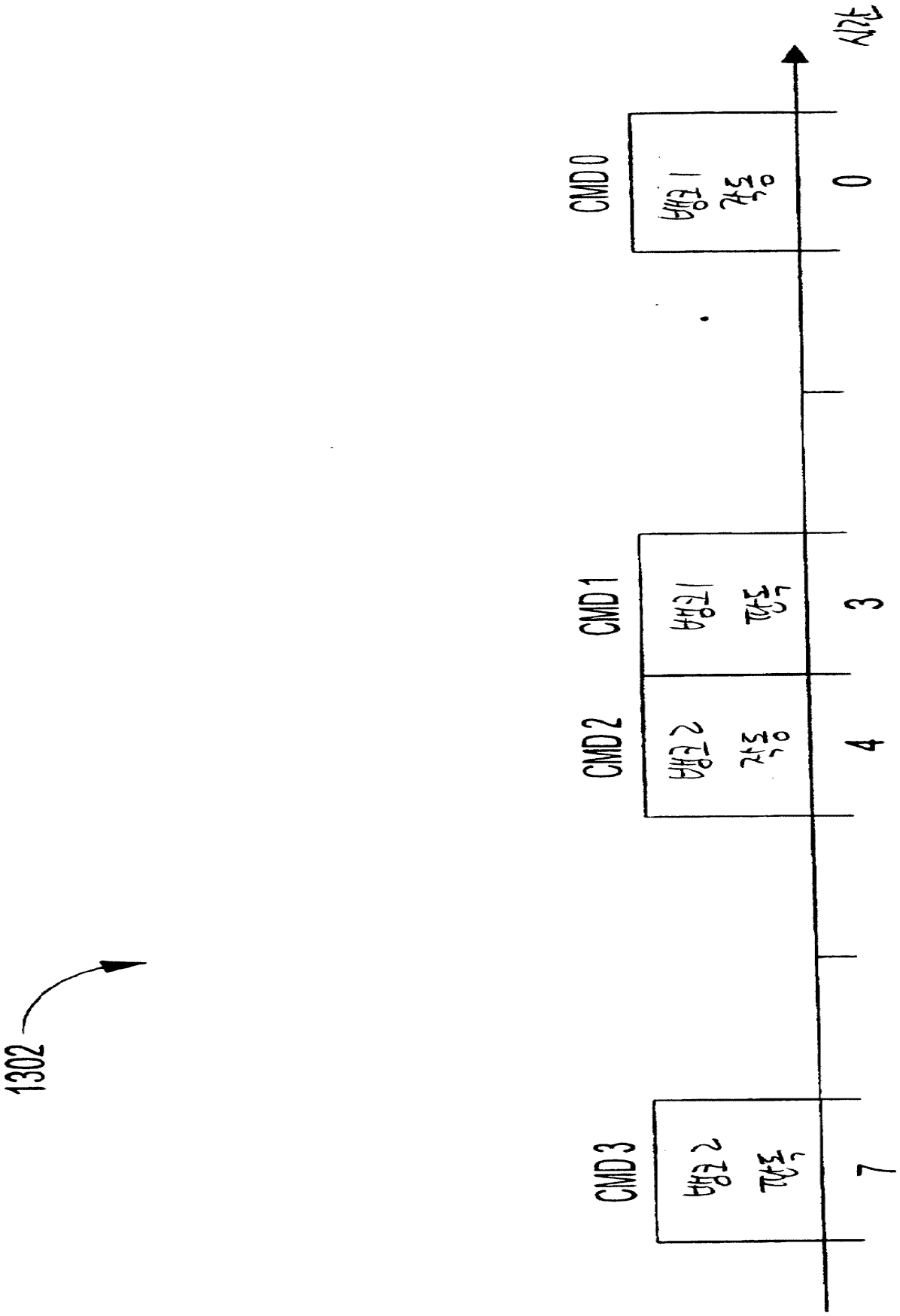
도면 11


1100
LUT
1108
명령
버퍼기
1110
1106
명령 변환기
1112
뱅크 레지스터
1104
어드레스
디코더
1102
새 어드레스
새 명령
데이터 인터페이스
범용 명령기

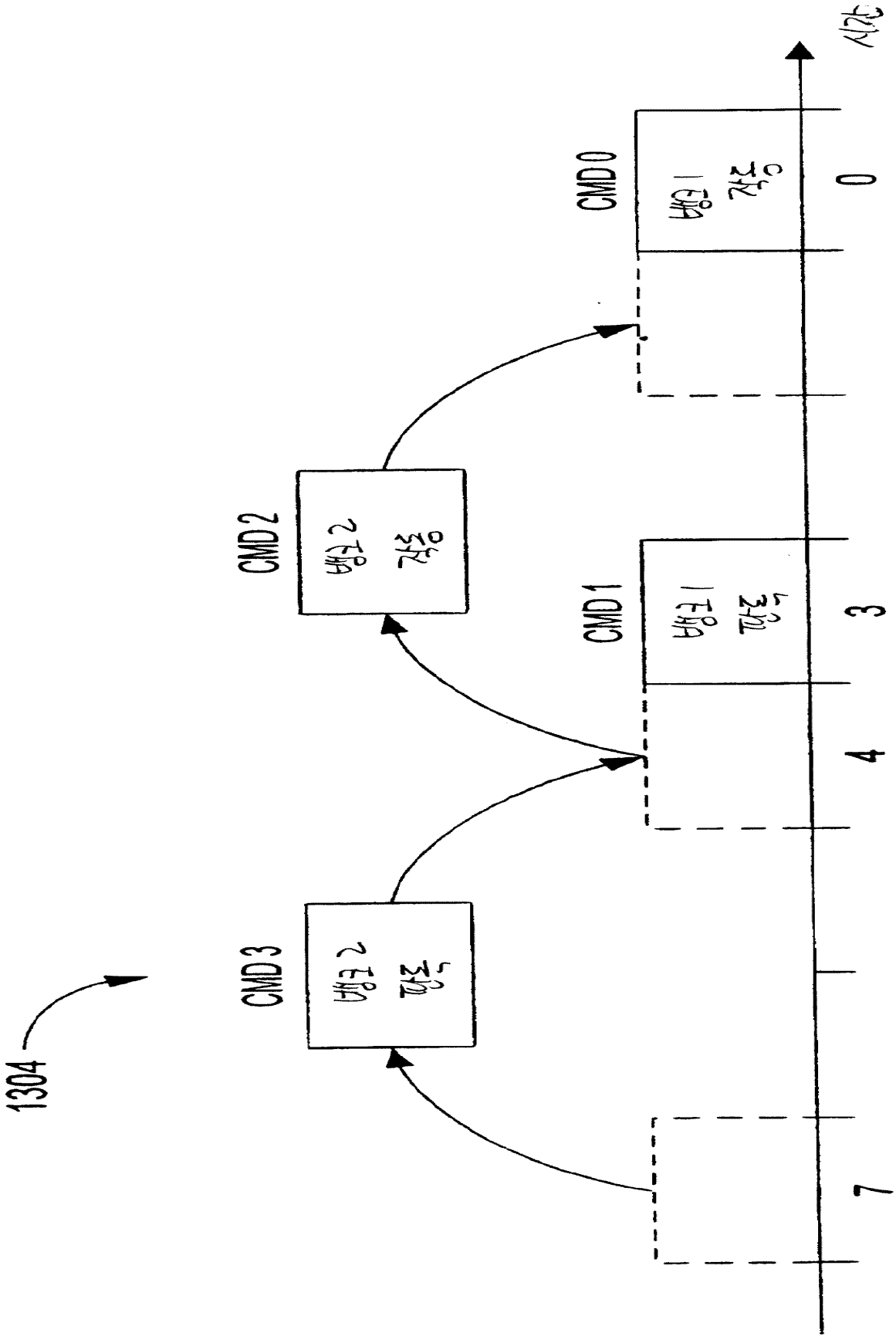
도면 12



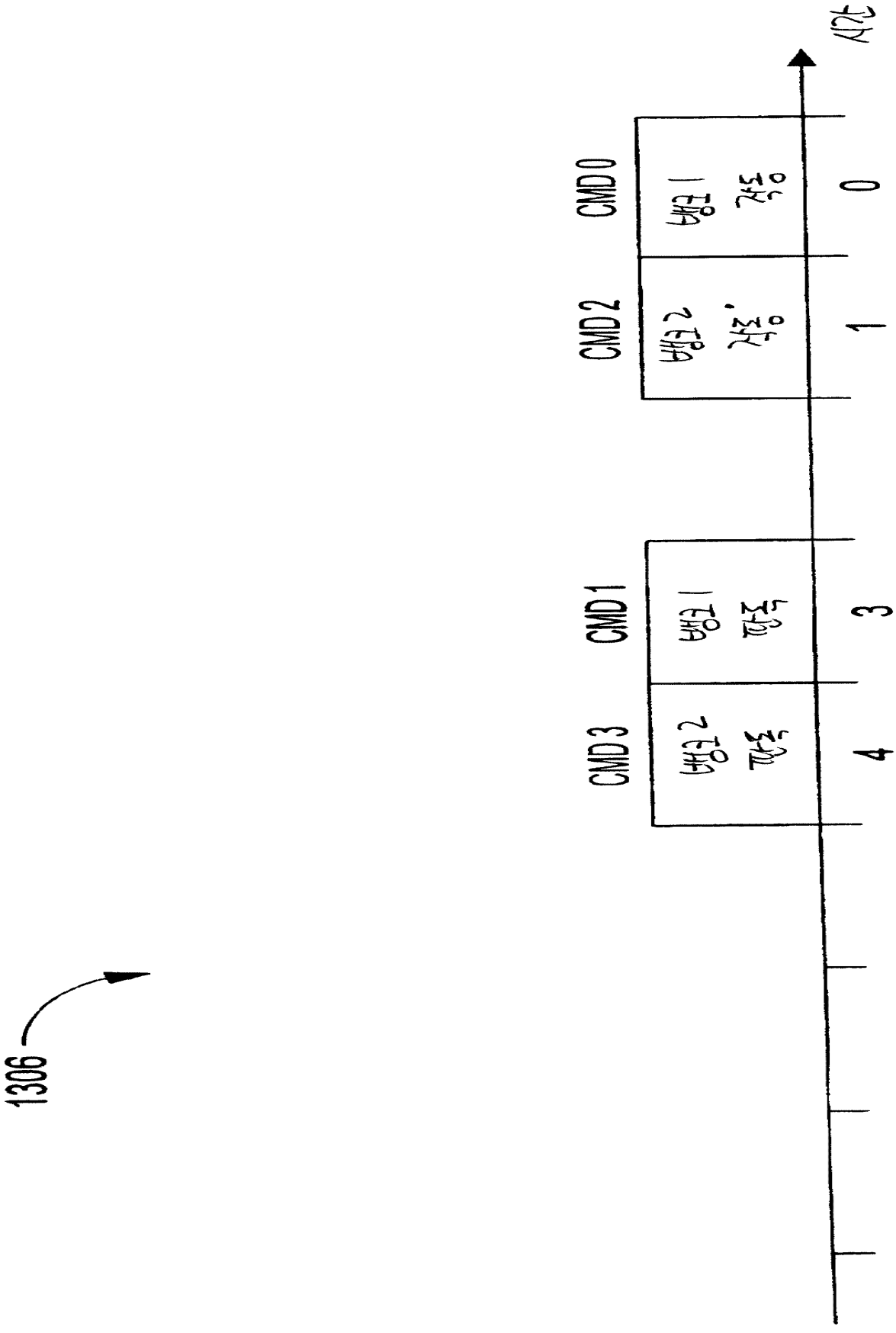
도면 13a



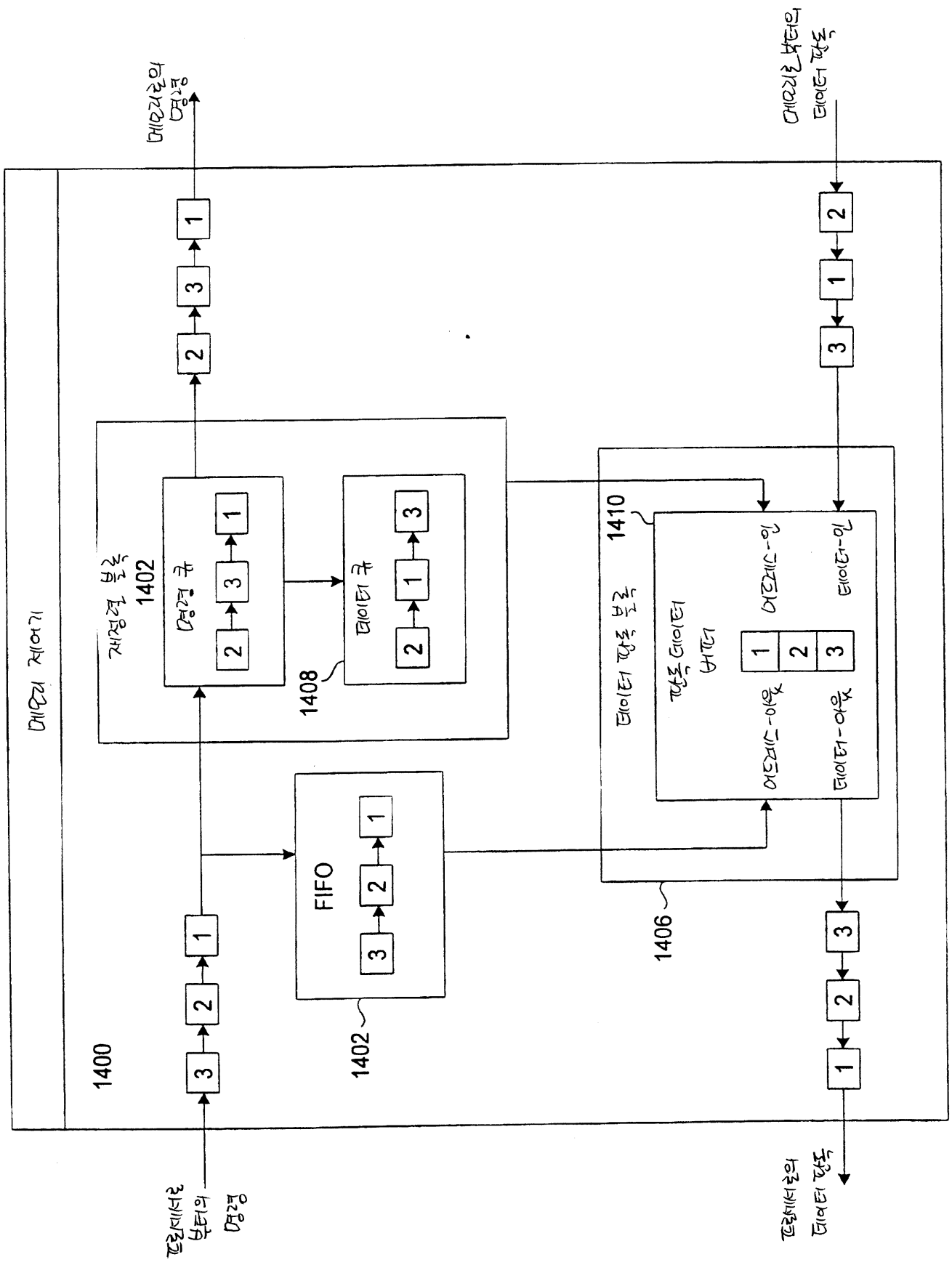
도면 13b



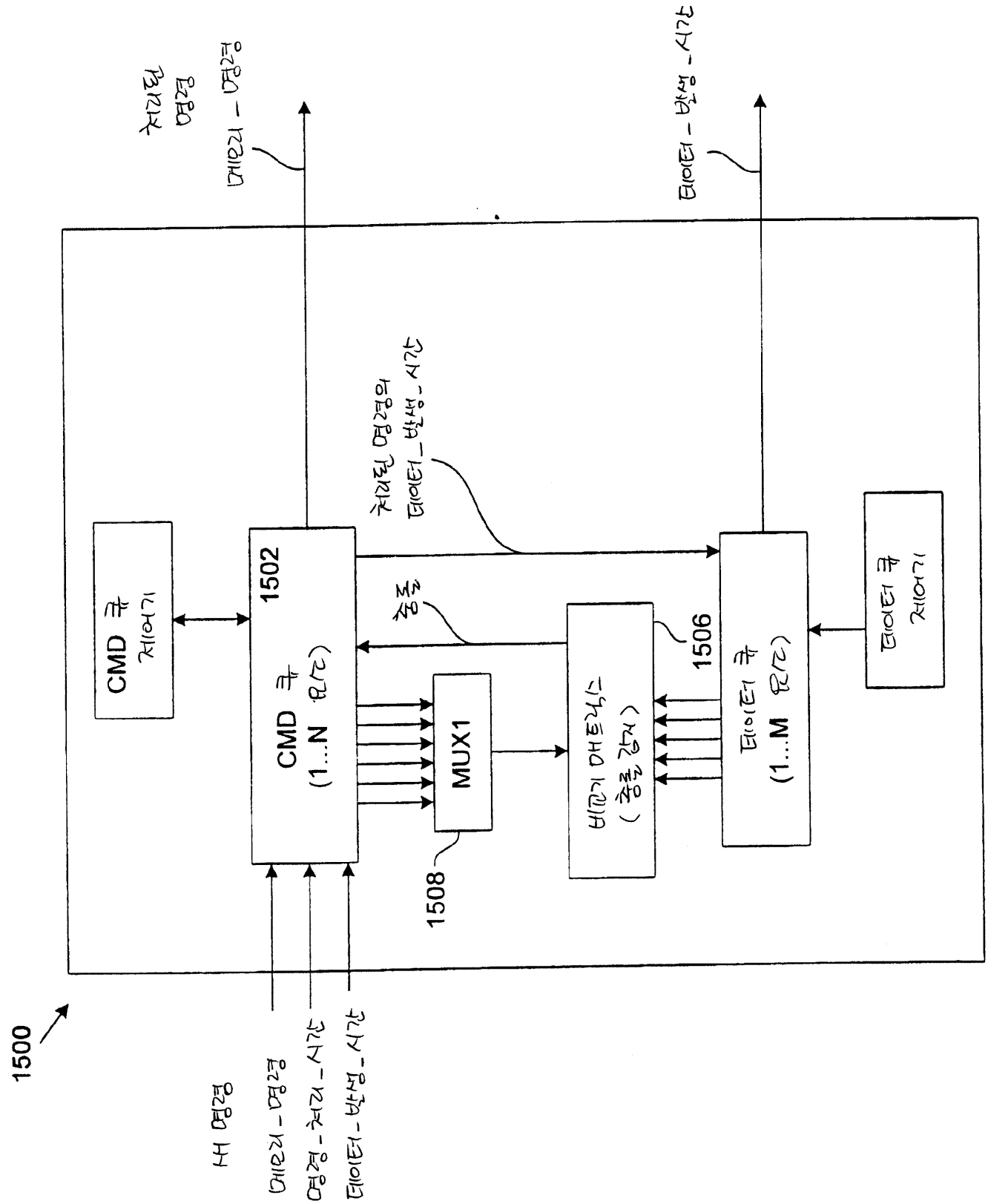
도면 13c



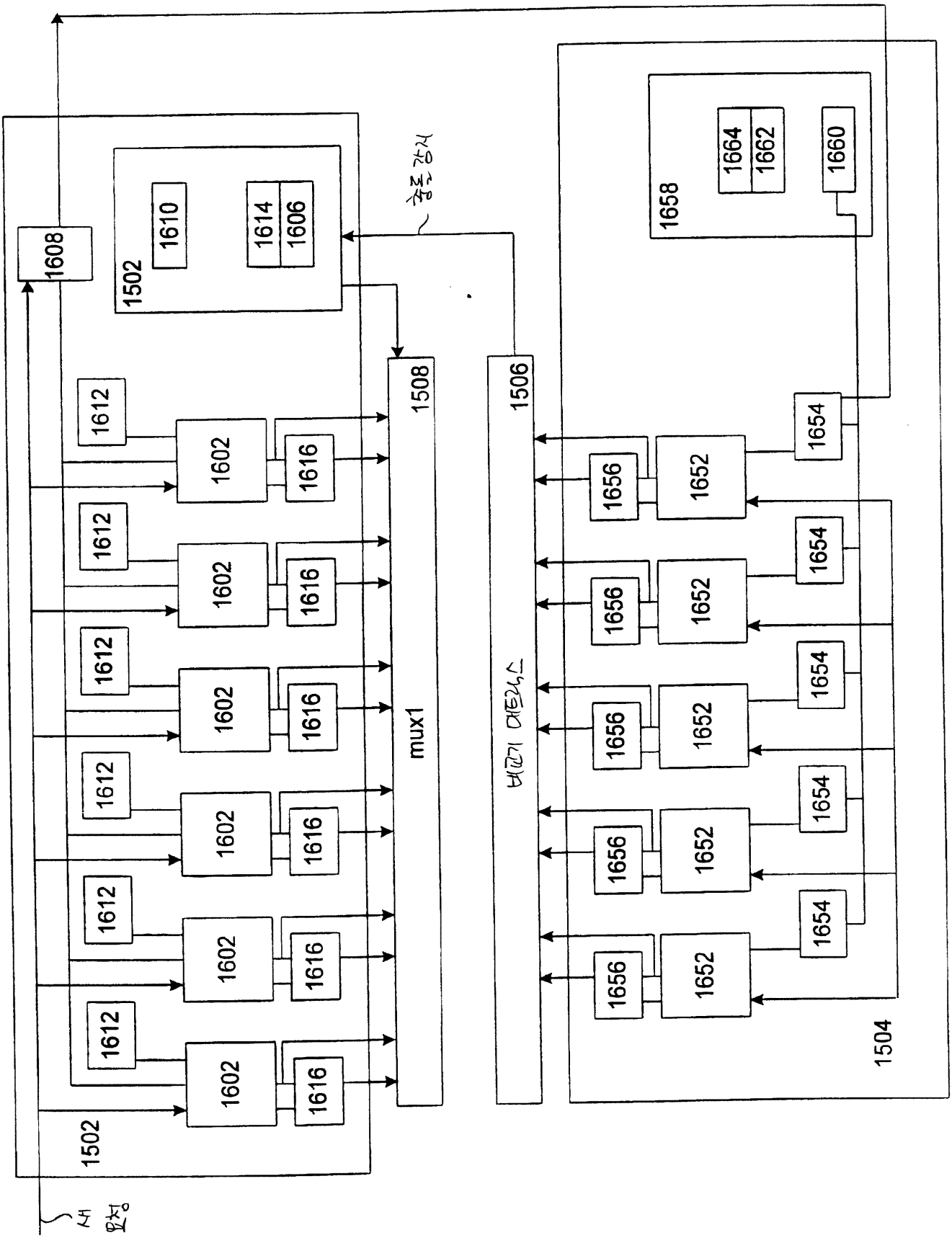
도면 14


메모리 제어기
1400
재정렬 블록
1402
명령 큐
데이터 큐
1408
FIFO
1402
데이터 패킹 블록
1410
패킹 데이터 버퍼
어드레스-인
데이터-인
어드레스-아웃
데이터-아웃
1406
메모리로의 명령
메모리로부터의 데이터 판독
프로세서로부터의 명령
프로세서로의 데이터 판독

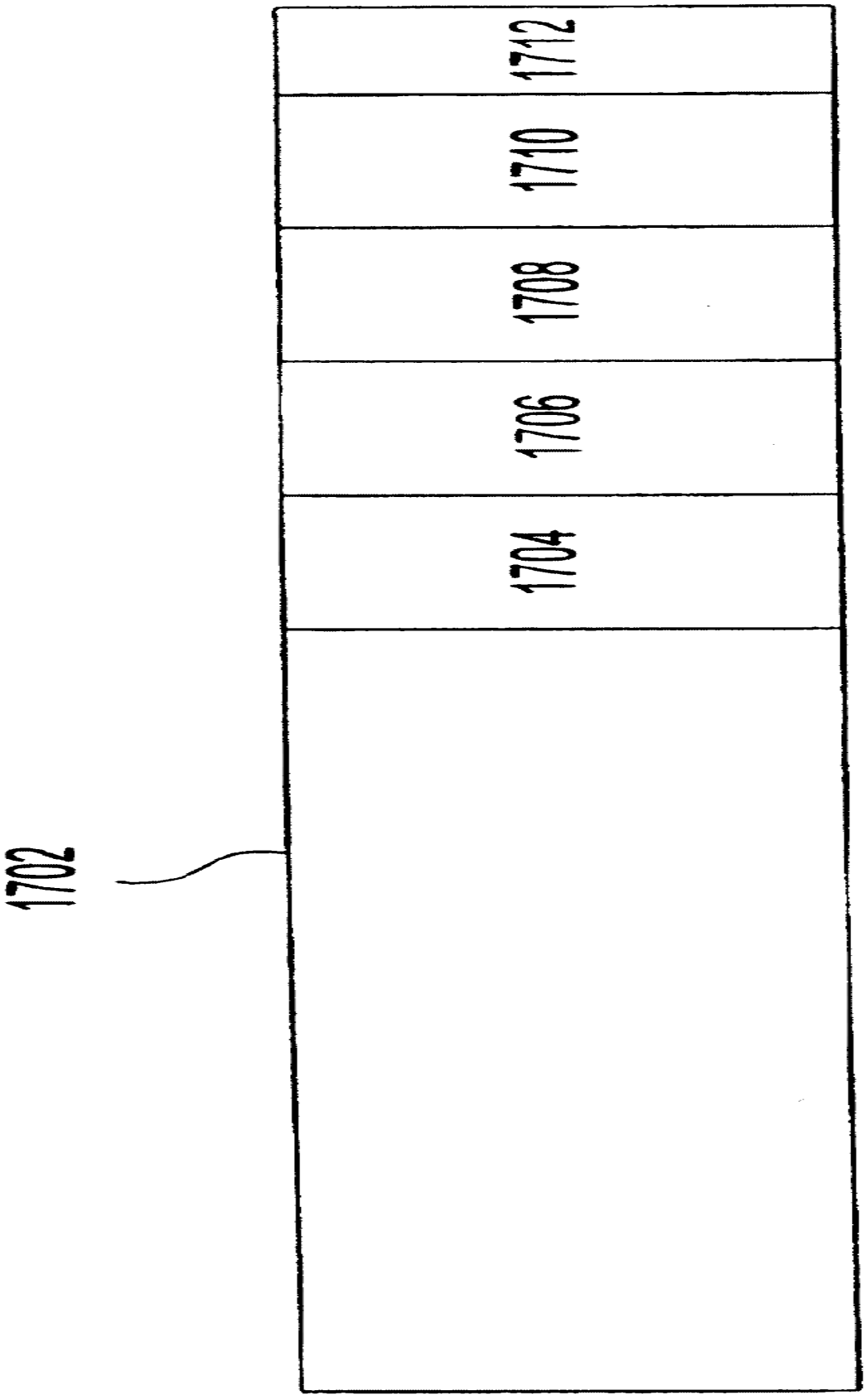
도면 15


1500
처리된 명령
메모리 - 명령
데이터 - 반생 - 시간
CMD 큐 제어기
1502
CMD 큐 (1...N 로우)
처리된 명령의 데이터 - 반생 - 시간
출구
MUX1
1508
비관기 매트릭스 (출구 감지)
1506
데이터 큐 (1...M 로우)
데이터 큐 제어기
새 명령
메모리 - 명령
명령 - 처리 - 시간
데이터 - 반생 - 시간

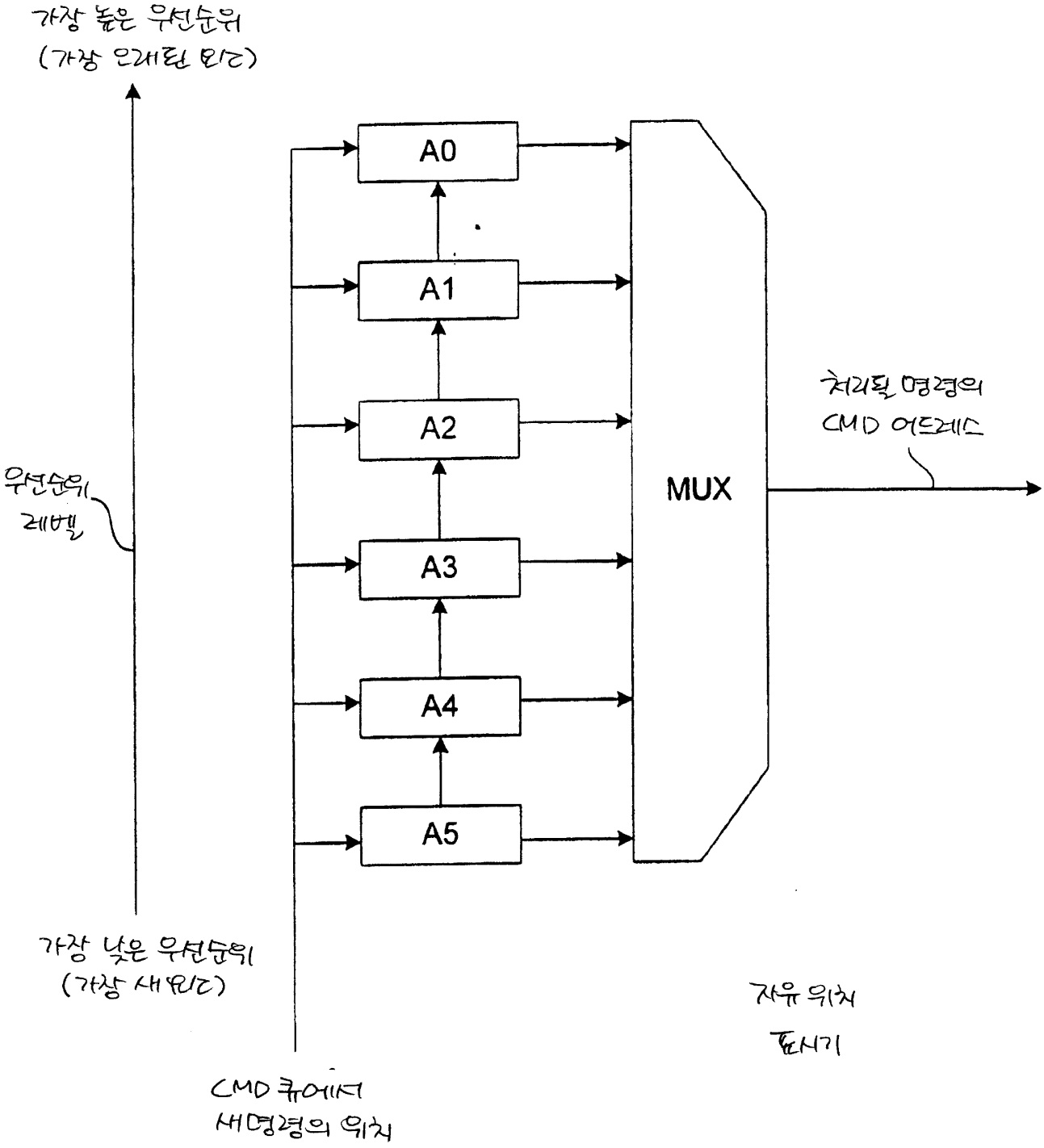
도면 16


1608
1502
1610
1614
1606
충돌 감지
1508
1506
1658
1664
1662
1660
1612
1602
1616
1654
1656
1652
mux1
버퍼기 메모리스
1504
1502
서버

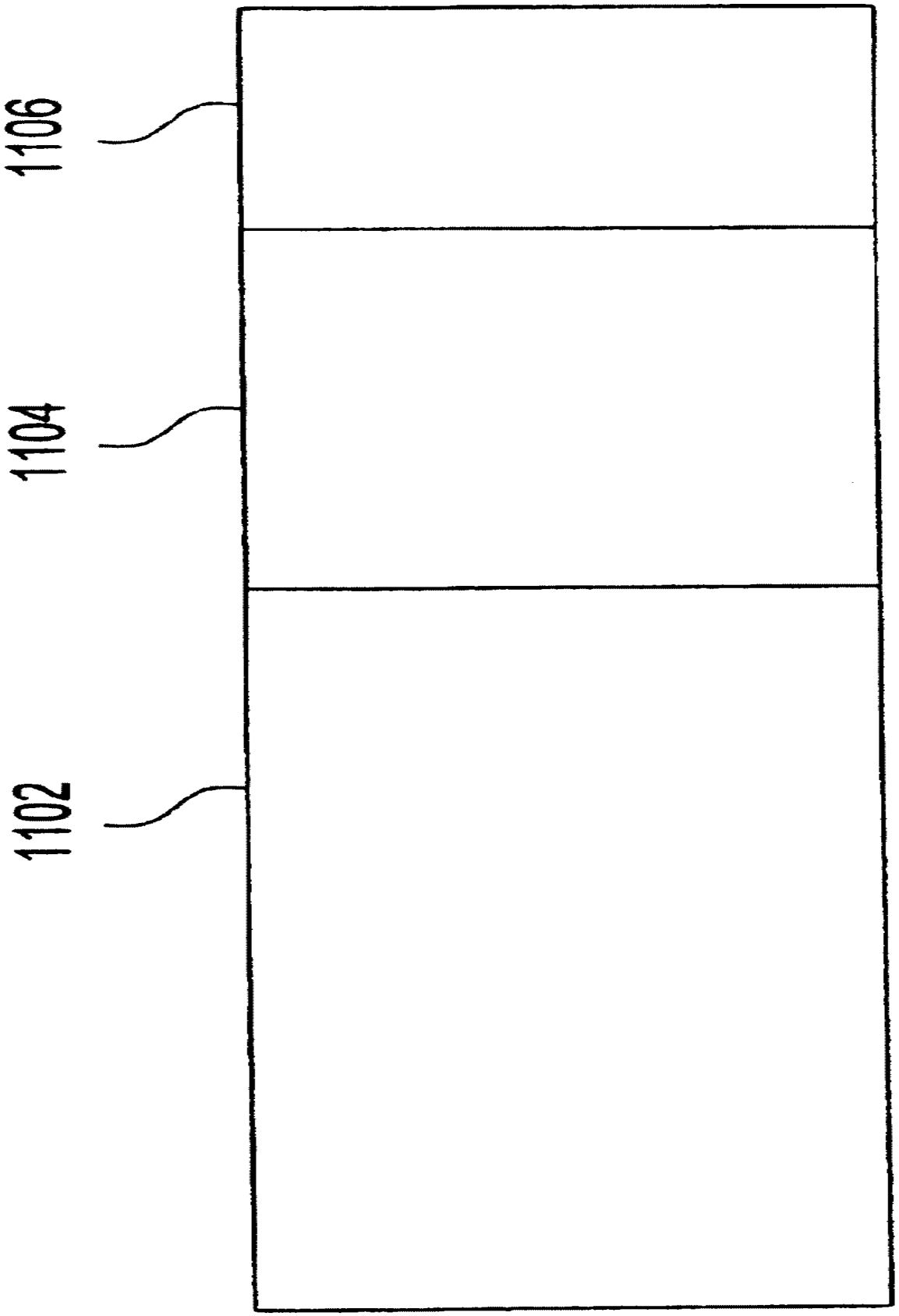
도면 17



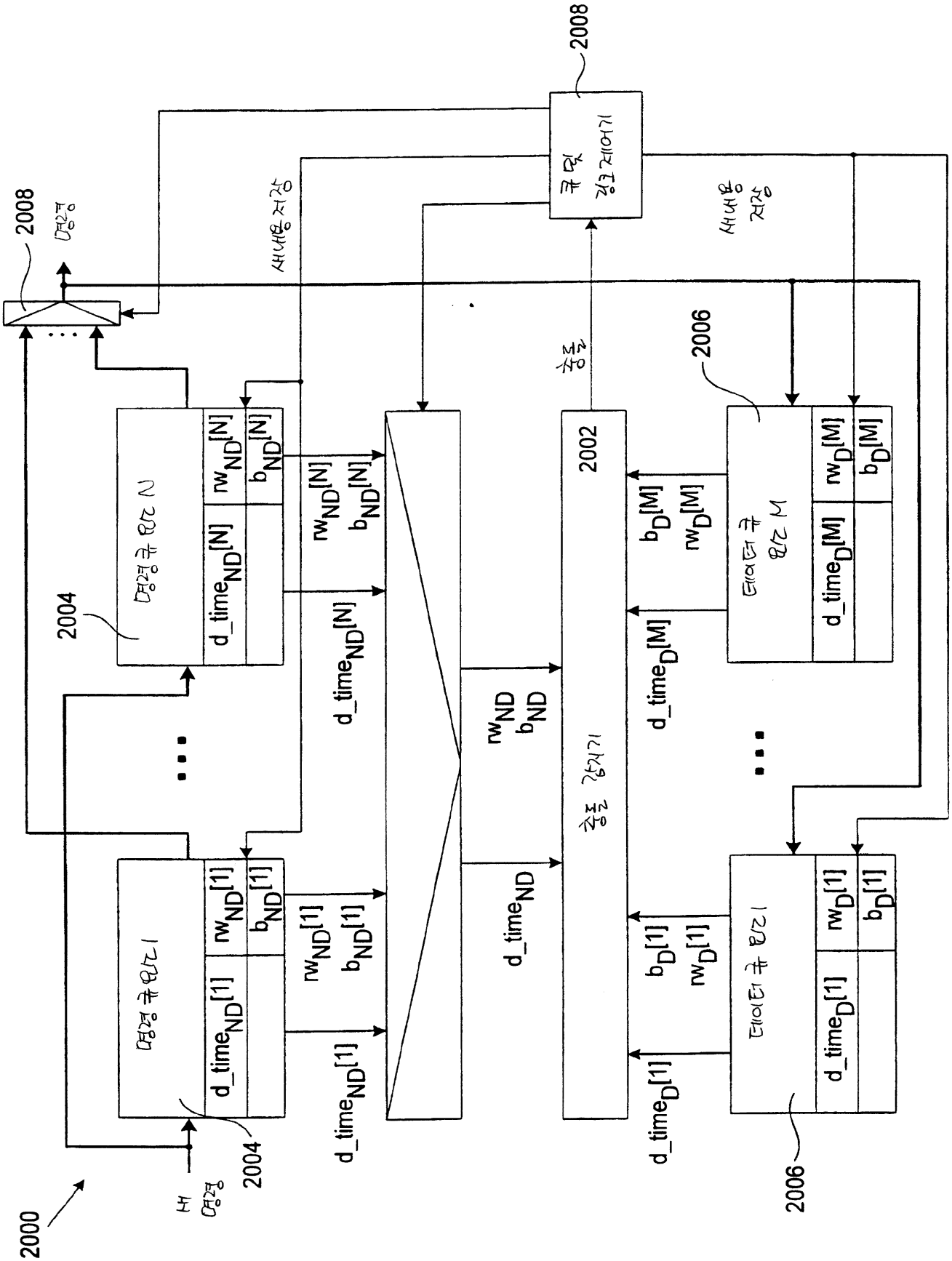
도면 18


가장 높은 우선순위
(가장 오래된 위치)
A0
A1
A2
A3
A4
A5
MUX
처리될 명령의
CMD 어드레스
우선순위
레벨
가장 낮은 우선순위
(가장 새로운 위치)
자유 위치
포인터
CMD 큐에서
새명령의 위치

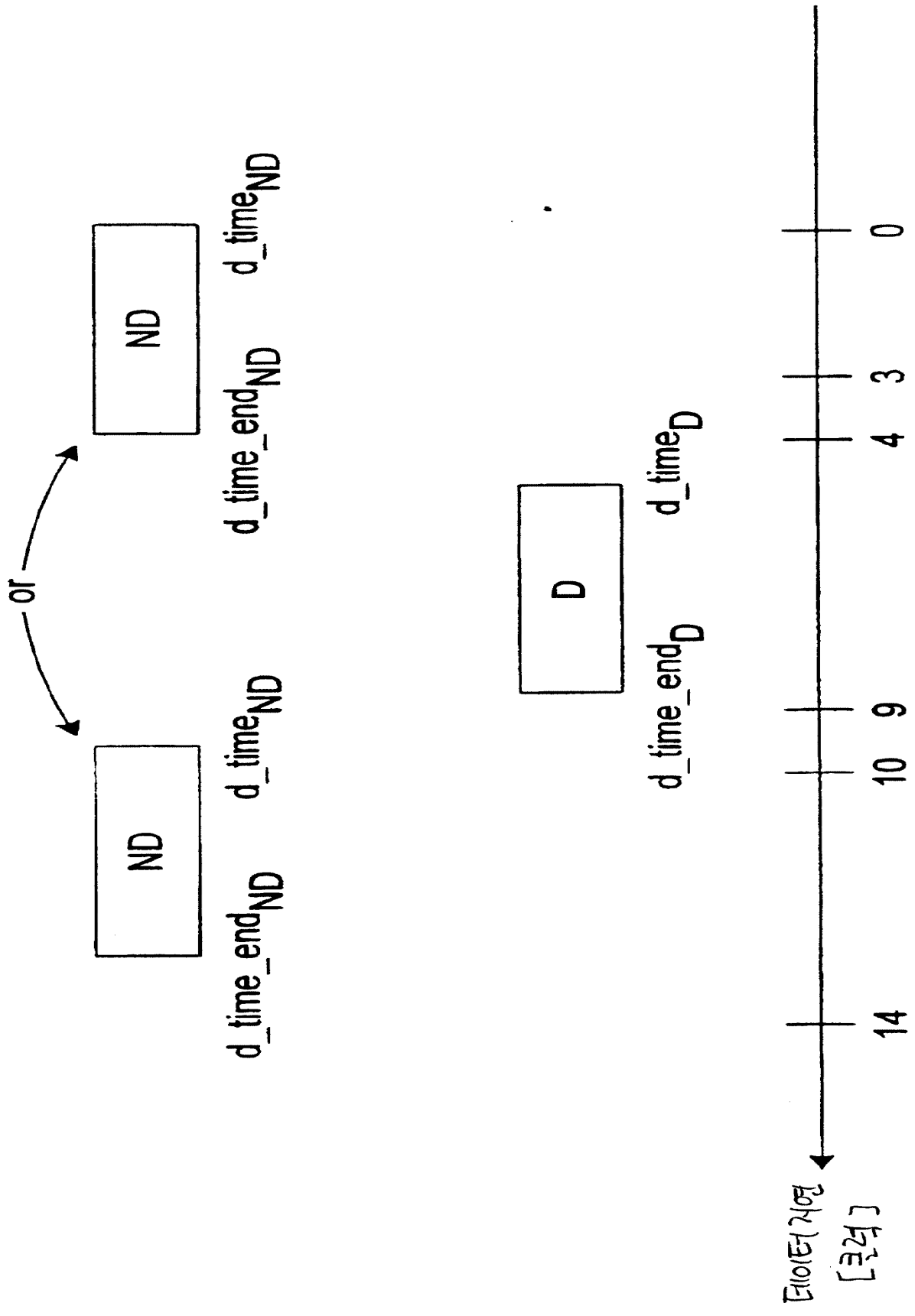
도면 19



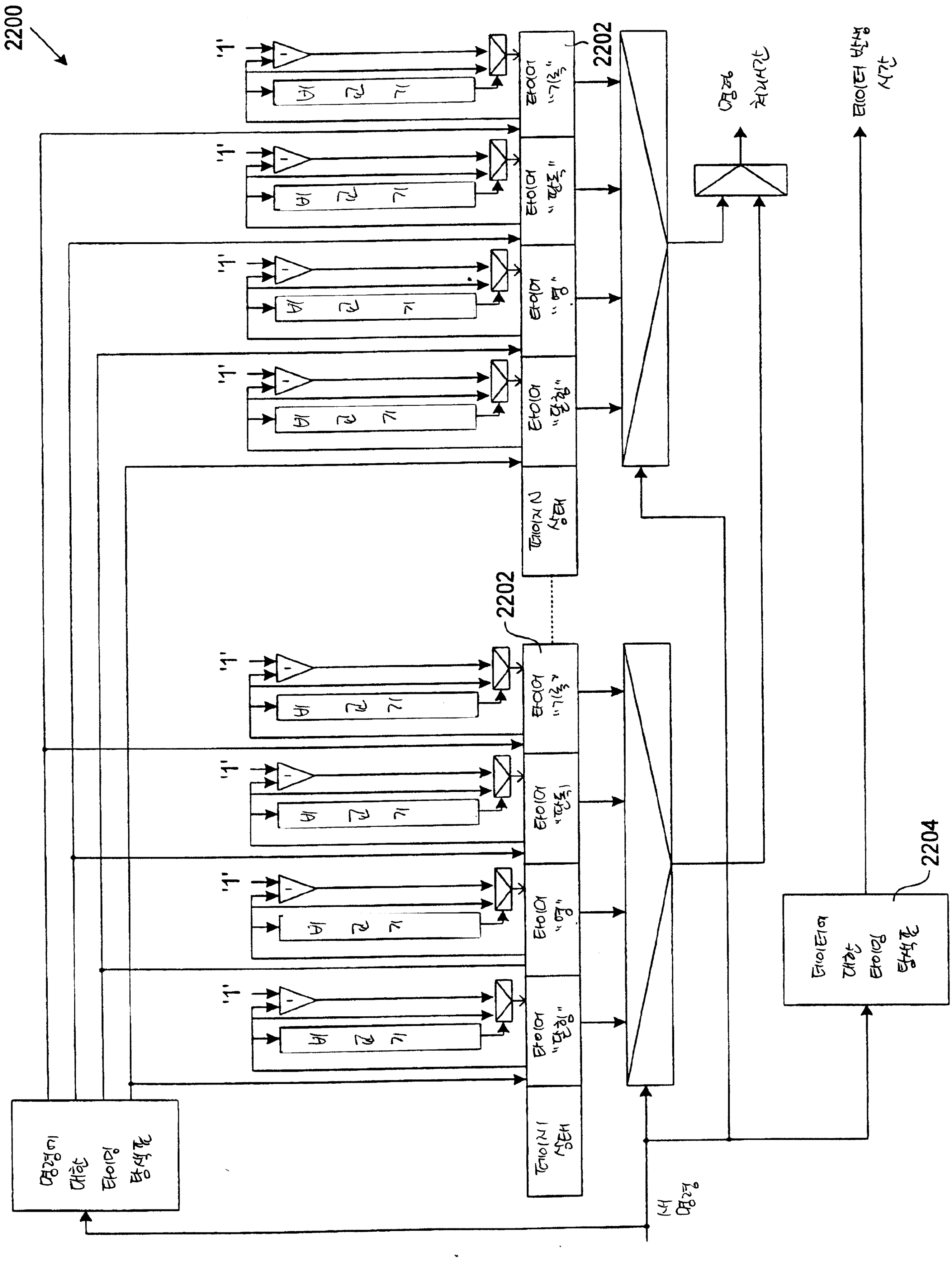
도면 20


2000
2002
2004
2006
2008
명령
새내용저장
출력
명령큐 0
명령큐 N
데이터큐 0
데이터큐 M
d_time_ND[1]
d_time_ND[N]
d_time_ND
d_time_D[1]
d_time_D[M]
w_ND[1]
w_ND[N]
w_ND
w_D[1]
w_D[M]
b_ND[1]
b_ND[N]
b_ND
b_D[1]
b_D[M]
출력 감지기

도면 21


ND
ND
or
d_time_end_ND
d_time_ND
d_time_end_ND
d_time_ND
D
d_time_end_D
d_time_D
0
3
4
9
10
14
데이터 거리
[크럭]

도면 22



도면 23

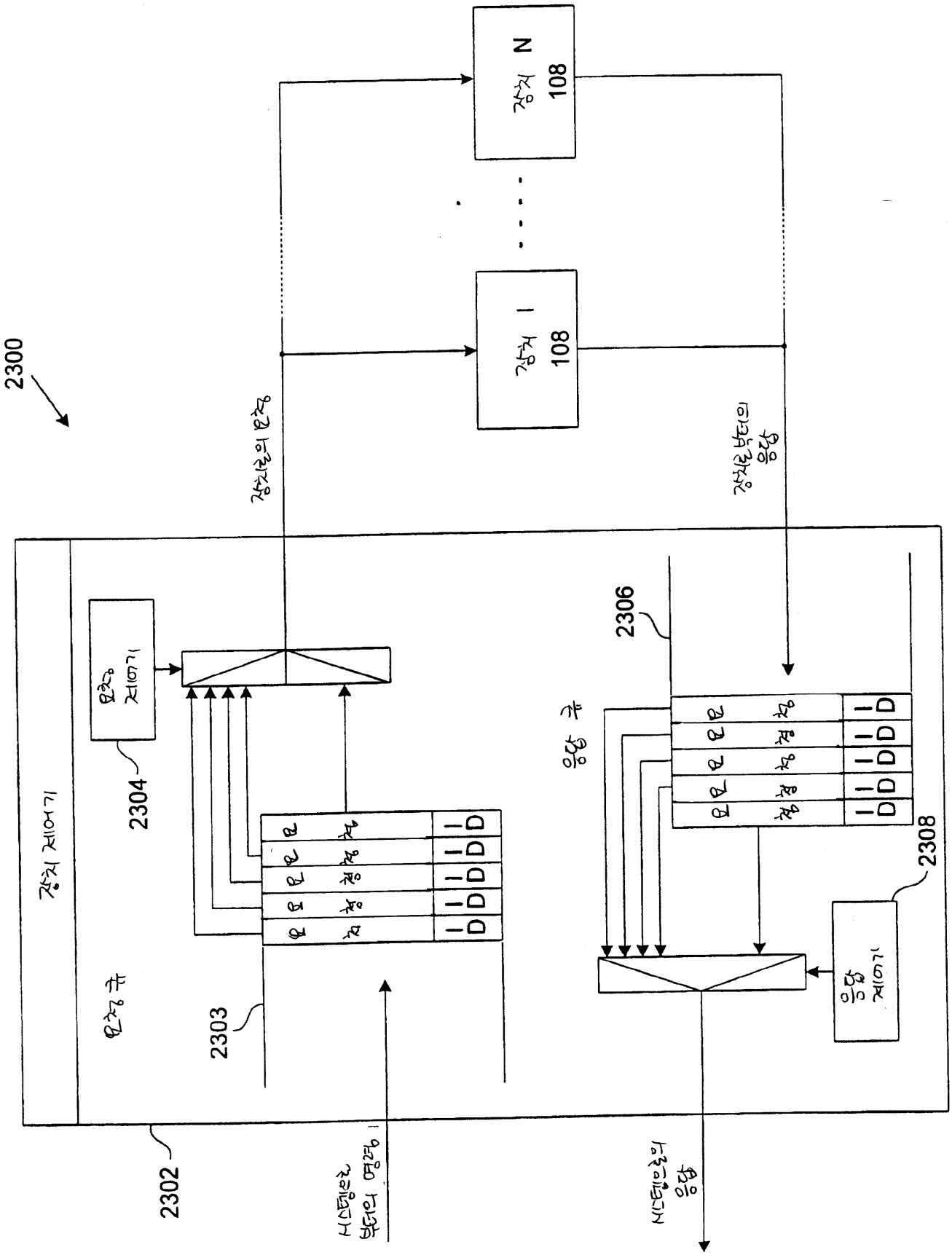

2300
장치 1
108
장치 N
108
장치의 요청
장치로의 응답
장치 제어기
요청 제어기
2304
2306
응답 큐
2308
응답 제어기
요청 큐
2303
2302
시스템으로 명령
시스템으로 응답

Table 4

in_command	명령		가상 랭크 0	1	2	3	대체	LUT 입력	ΔT
open @ 1000	100	400					0		0
page read @1000	100	400	100 open				1	open page read region 0	3ϕ
page write @1001	100	400	100 page read				2	page read page write region 0	2ϕ
open @ 1001	10	200	100 page write				3	open	0
page read @ 1001	100	400	100 page read			200 open	0	page read page write region 0	1

도면 24

공개특허 특2001-0086035