(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) Int. Cl.[7]
G06F 13/00
G06F 9/34

(45) 공고일자 2001년01월15일
(11) 등록번호 10-0278136
(24) 등록일자 2000년10월17일

(21) 출원번호 10-1993-0018923
(22) 출원일자 1993년09월18일
(65) 공개번호 특1994-0007692
(43) 공개일자 1994년04월27일

(30) 우선권주장 92-276665 1992년09월21일 일본(JP)

(73) 특허권자 가부시끼가이샤 히다치 세이사꾸쇼 가나이 쓰도무
일본국 도쿄도 지요다구 간다 스루가다이 4-6
(72) 발명자 나까하라시게루
일본국 도꾜도 오우메시 신마찌769 세이와료
(74) 대리인 백남기

심사관 : 오홍수

(54) 데이타처리장치 및 데이타처리방법

요약

연산코드의 증대를 억제하면서 고기능을 처리를 실현할 수 있고, 또 고기능인 처리를 적은 스텝수로써 실현할 수 있는 데이타처리장치 및 데이타처리방법에 관한 것으로써, 연산코드의 비트수나 명령제어부의 논리규모의 증대를 억제하면서 기능을 다양하게 확장할 수 있게 하기 위해, 레지스터에 대한 데이타의 라이트 또는 리드시에 그 데이타에 대해서 특정한 처리를 실시하는 기능을 미리 레지스터(기능을 갖는 레지스터) 에 할당해서 기능을 갖는 레지스터를 명령 중에서 지정하는 것에 의해 데이타에 대해서 명령의 연산코드에 의해서 규정되는 명령의 처리와 기능을 갖는 레지스터가 갖는 특정 기능을 조합한 다양한 데이타처리를 실현한다.

대표도

도1

명세서

[발명의 명칭]

데이타처리장치 및 데이타처리방법

[도면의 간단한 설명]

제1도는 본 발명에 관한 데이타처리장치의 기능을 갖는 레지스터의 정의와 기능의 1 예의 설명도.

제2도는 기능을 갖는 레지스터의 1예인 낫레지스터의 C언어적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제3도는 기능을 갖는 레지스터의 1예인 인크리먼트 레지스터의 C언어적인 적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제4도는 기능을 갖는 레지스터의 1예인 수정 레지스터의 C언어적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제5도는 기능을 갖는 레지스터의 1예인 수정 인덱스레지스터 및 얼라인먼트 레지스터의 C언어적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제6도는 상기 얼라인먼트 레지스터를 사용한 다른 처리예를 도시한 설명도.

제7도는 기능을 갖는 레지스터의 1예인 자리올림/빌림수 레지스터의 C언어적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제8도는 기능을 갖는 레지스터의 1예인 시프트 레지스터의 C언어적인 정의 및 그것을 사용한 처리예를 개념적으로 도시한 설명도.

제9도는 제2도~제8도에서 설명한 기능을 갖는 레지스터를 사용한 데이타처리장치의 1실시예의 블럭도.

제10도는 기능을 갖는 레지스터 및 범용레지스터의 지정방식을 도시한 1예의 설명도.

제11도는 기능을 갖는 레지스터의 오퍼레이션을 제9도의 구성에 따라서 실행할때의 제어방식을 파이프라인형식으로 도시한 1예의 설명도.

제12도는 제11도와는 다른 기능을 갖는 레지스터의 오퍼레이션을 제9도의 구성에 따라서 실행할때의 제어

방식을 파이프라인형식으로 도시한 1예의 설명도.

제13도는 마스크회로의 1예의 회로도.

제14도는 제9도에 도시되는 데이타처리장치의 칩평면도.

제15도는 기능을 갖는 레지스터를 갖지 않는 데이타처리장치와 본 발명에 관한 데이타처리장치의 파이프라인 스테이지에 의해 구분된 1예의 동작 흐름도.

제16도는 명령의 포맷과 하드웨어의 관계를 도시한 설명도.

[발명의 상세한 설명]

본 발명은 연산코드의 증대를 억제하면서 고기능인 처리를 실현가능하게 하며, 또 고기능인 처리를 적은 스텝수로써 실현가능한 데이타처리장치 및 데이타처리방법에 관한 것으로써, 예를들면 마이크로컴퓨터와 같은 논리집적회로장치에 적용해서 유효한 기술에 관한 것이다.

펀노이만형 등의 데이타처리장치의 기본적인 구성은 거의 같고, 예를들면 캐시메모리 등에 기억한 일련의 명령은 순번으로 명령제어부로 리드하고, 그 명령이 무엇인가를 나타내고 있는 연산코드라 불리우는 비트영역을 디코드해서 데이타에 실시해야할 처리를 결정하고 또, 오퍼랜드라 불리우는 피연산데이타를 지정하는 영역을 디코드해서 데이타를 페치하여 상술한 처리를 그 데이타에 대해서 실행하는 것이다. 이와 같은 데이타처리장치의 1예로써는 Mips사의 R4000칩 (닛케이일렉트로닉스, 1991년10월14일 닛케이PB사발행, No. 538호, pp. 171~185), Intel사의 i960칩 (닛케이일렉트로닉스, 1990년1월8일 닛케이PB사발행, No. 490호, pp. 177~186) 및 Hewlett Packard사의 PA-RISC칩 (닛케이마이크로디바이스, 닛케이PB사 발행의 1999년9월호, PP. 50~55) 등을 열거할 수 있다.

또, 데이타처리장치의 아커댁쳐는 비교적 간단한 명령세트로써 처리의 고속화 및 하드웨어의 간소화를 도모하고자 하는 RISC (Reduced Instruction Set Computer) 적 아커택처와 오브젝트지향의 아커택쳐와 같은 비교적 복잡한 명령세트로써 고기능화를 지향하는 CISC (Complex Instruction Set Computer) 적 아커택처로 크게 구분할 수 있다.

현실적으로 제공되어 있는 각종 데이타처리장치의 아커택처가 RISC인지 CISC중 어느것인가 라는 식별은 실제로는 어렵지만 다양한 어드레싱모드나 1개의 명령으로 복수의 연산을 실행하는 복합명령등을 CISC적인 명령이라 생각할 수 있다.

상기 RISC적인 데이타처리장치에 있어서는 CISC적인 데이타 처리장치가 1명령으로 제공하는 고기능인 처리를 기본적으로 간단한 처리를 실현하는 명령을 여러개 조합하는 것에 의해 대처하고 있었다. 이 때문에 RISC적인 데이타처리장치에 있어서는 출현빈도가 적을지도 모르지만 그와 같은 처리를 능률적으로 실행할 수 없었다. 이때, 다양한 어드레싱모드나 1개의 명령으로 복수연산을 실행하는 복합명령 등 CISC적인 명령을 기본적인 명령세트에 추가하고자 하면 하드웨어가 서포트하는 명령수가 증가해버린다. 그 결과 명령제어부의 디코더등의 랜덤논리의 논리규모가 크게 되어 임계경로의 지연시간의 증대, 즉 LSI칩의 성능저하나 칩면적의 증가, 개발일정의 증대화 등이 초래된다.

본 발명의 목적은 연산코드의 증대를 억제하면서 기능의 다양화를 실현할 수 있는 데이타처리장치 및 데이타처리방법을 제공하는 것이다.

또, 본 발명의 다른 목적은 고기능인 처리를 적은 스텝수로써 실현할 수 있는 데이타처리장치 및 데이타처리방법을 제공하는 것이다.

본 발명의 또 다른 목적은 하드웨어의 증대를 극력 억제해서 다양한 기능을 실현할 수 있는 데이타처리장치를 제공하는 것이다.

본 발명의 상기 및 그외의 목적과 새로운 특징은 본 명세서의 기술 및 첨부도면으로 명확하게 될 것이다.

본원에서 개시되는 발명중 대표적인 것의 개요를 설명하면 다음과 같다.

즉, 레지스터에 대한 데이타의 라이트 또는 레지스터에서의 데이타의 리드시에 그 데이타에 대해서 특정한 처리를 실시하는 기능을 미리 소정의 레지스터(기능을 갖는 레지스터)에 할당해서 기능을 갖는 레지스터를 명령중에서 지정하는 것에 의해 기능을 갖는 레지스터에서 리드되거나 또는 라이트되는 데이타에 대해서 명령의 연산 코드에 의해서 규정되는 명령의 처리와 지정된 기능을 갖는 레지스터가 갖는 특정기능을 조합한 다양한 데이타처리를 실현하는 것이다.

이것에 의해, 명령의 연산코드에 의해서 규정되는 명령의 처리와 기능을 가진 레지스터가 갖는 특정 기능을 조합해서 다양한 데이타 처리를 실현할 수 있다. 이것은 명령세트의 명령수를 증가하는 일 없이, 즉 연산코드의 비트수를 증대시키는 일없이 데이타처리장치에 의한 처리를 다양화 또는 다기능화하도록 작용하고, 연산코드의 디코드논리로 대표되는 명령제어부의 하드웨어의 증대를 극력 억제해서 다양한 기능을 실현할 수 있다.

상기에 의해서 실현되는 다양한 기능은 1개의 명령중의 오퍼랜드 지정영역 등에서의 기능을 갖는 레지스터의 지정에 의해 달성되며, 또한 그 지정된 기능을 갖는 레지스터에 할당되는 고유의 기능은 상기 레지스터에 대한 데이타의 라이트 또는 리드시에 실행되므로, 기능확장에 따라서 실행해야할 명령수가 증가하지 않고, 이것이 고기능인 처리를 적은 스텝수로써 실현하도록 작용한다.

제1도에는 본 발명에 관한 데이타처리장치의 기능을 갖는 레지스터 (이하, 오퍼레이팅 레지스터라고도 한다) 의 정의와 기능의 1 예가 도시된다. 제1도에 있어서의 오퍼레이팅 레지스터의 일반적인 정의는 C언어적인 형태로 표시되고, 오퍼레이팅 레지스터는 (1) 에 표시되는 Operating Read Data Register:Reg[R]과 (2) 에 표시되는 Operating Write Data Register:Reg[W]로 크게 구별된다. 오퍼레이팅 리드 데이타 레지스터Reg[R]은 미리 정의되어 있는 Reg[R]고유의 오퍼레이션Fr(i)를 i 번째의 명령의 Reg[R]로 규정되는

특정의 필드의 값을 파라미터로 해서 Reg[R]의 내용에 대해서 실행하고, 그 결과를 리드데이타 (read data)로 하는지 재차 상기 Reg[R]로 되돌려 라이트하는지의 어느것인가 한쪽이던가 또는 그 양쪽을 실행한다.

오퍼레이팅 라이트 데이타 레지스터Reg[W]는 미리 정의되어 있는 Reg[W]고유의 오퍼레이션Fw(i) 를 i번째의 명령의 Reg[W]로 규정되는 특정의 필드의 값을 파라미터로 해서 라이트 데이타 (write data)에 대해서 실행하고 상기 Reg[W]로 라이트한다.

상기 오퍼레이팅 레지스터Reg[R], Reg[W]는 제1도에 도시되는 포맷의 명령중에서 지정할 수 있다. 도면에 도시되는 명령은 연산코드 지정필드와 오퍼랜드 및 그밖의 정보의 지정필드를 갖고, 연산코드 지정필드에는 데이타처리장치의 아커택쳐에 따라서 준비되어 있는 연산코드 (LOAD나 ADD라는 명령코드)OP가 기술되고, 오퍼랜드 및 그밖의 정보의 지정필드에는 상기 연산코드에 따라서 명려을 실행하기 위해 필요한 오퍼랜드 (연산대상으로 되는 것)그 자체나 그 소재, 더나아가서는 그밖의 제어정보가 기술된다. 상기 오퍼레이팅 레지스터Reg[R], Reg[W]는 명령포멧중에서 상기 오퍼랜드 및 그밖의 정보의 지정필드에서 지정된다. 그 지정은 예를들면 오퍼레이팅 레지스터 Reg[R], Reg[W]에 할당된 고유의 레지스터번호로 지정할 수 있다.

이와 같이 고유의 기능이 할당된 오퍼레이팅 리드 레지스터Reg[R]을 명령중에서 지정했을때 상기 명령의 실행에 있어서는 그명령의 연산코드로 규정되는 처리 (오퍼레이션) 와 오퍼레이팅 리드 데이타 레지스터 Reg[R]고유의 처리가 상기 오퍼레이팅 리드 데이타 레지스터Reg[R]의 유지정보를 이용해서 실행된다. 예를들면 그 명령중에서 지정된 오퍼레이팅 레지스터Reg[R]에 미리 할당되어 있는 고유의 오퍼레이션을 상기 레지스터의 유지값에 대해서 실행하고, 그 결과를 상기 오퍼레이팅 레지스터Reg[R] 에서의 리드데이타로써 연산코드로 지정된 처리로 인도하는 처리를 실행한다. 상기 연산코드로 지정된 처리로 인도한후 그 인도한 리드데이타를 상기 오퍼레이팅 레지스터Reg[R]로 되돌려 라이트하는 처리를 필요에 따라서 또 실행할 수 있다.

또, 고유의 기능이 할당된 오퍼레이팅 라이트 데이타 레지스터Reg[W]를 명령중에서 지정했을때 상기 명령의 실행에 있어서는 그 명령의 연산코드로 규정되는 처리와 오퍼레이팅 라이트 데이타 레지스터Reg[W]고유의 처리가 실행되고, 그 결과가 상기 오퍼레이팅 라이트 데이타 레지스터Reg[W]에 저장된다. 예를들면 그 명령중에서 지정된 오퍼레이팅 레지스터Reg[W]에 미리 할당되어 있는 고유의 오퍼레이션을 그 명령의 연산코드에 의해서 상기 오퍼레이팅 레지스터Reg[W]로의 라이트가 지정된 데이타에 대해서 실행하고, 그것을 상기 오퍼레이팅 레지스터Reg[W]로 라이트하는 처리를 실행한다.

오퍼레이팅 리드 데이타 레지스터Reg[R]과 오퍼레이팅 라이트 데이타 레지스터Reg[W]에 할당된 각각 고유의 처리는 상기 레지스터에 대한 데이타의 리드 또는 라이트시에 실행된다.

이 원리적인 설명에서 명확한 바와 같이 기존의 소정의 연산코드로 오퍼레이팅 레지스터Reg[R], Reg[W]의 지정을 따르는 것에 의해 연산코드의 종류를 증가시키는 일없이 동일 연산코드, 즉 1종류의 명령에 의해서 실현할 수 있는 처리를 다양하게 또는 풍부하게 할 수 있어 명령수 및 명령제어부의 논리규모를 증가시키는 일없이 고기능적인 명령과 동일한 처리를 서포트할 수 있게 된다. 관점을 바꾸면 원래 고기능적인 명령을 전용의 연산코드로 준비하고 있는 데이타처리장치에 있어서는 그 기능을 삭감하는 일없이 명령수 및 명령제어부의 논리규모를 감소할 수 있다.

다음에 제2도~제8도를 참조하면서 상기 오퍼레이팅 레지스터의 구체적인 예 및 그것을 이용한 처리의 1예를 메모리와 레지스터간의 데이타의 흐름에 착안해서 개념적으로 설명한다.

제2도에는 낫레지스터의 C언어적인 정의 및 그것을 사용한 처리예가 개념적으로 도시된다. 낫레지스터N은 상기 오퍼레이팅 라이트 데이타 레지스터 Reg [W]의 범주에 포함되고, 낫레지스터N(Reg N])으로의 라이트 데이타의 각 비트값을 반전시켜서 낫레지스터N(Reg[N])으로 라이트하는 것이다. 제2도에는 응용예로써 디스티네이션에 낫레지스터N을 지정한 LOAD명령 「LOAD disp(b), N」이 도시되어 있다. 이 명령에 있어서는 디스티네이션에 낫레지스터N이 지정되어 있고, 이 명령이 실행되면 범용레지스터b {GR [b]}에 미리 설정된 값(기준 어드레스)에 디스플레이스먼트 (disp)를 가산하는 것에 의해 얻어진 메모리어드레스에 저장되어 있던 데이타가 낫레지스터N으로 라이트될때 상기 낫레지스터 고유의 기능으로써 그 데이타를 비트반전하는 기능이 부가된다. 따라서, 제2도에 도시되는 명령 「LOAD disp(b), N」과 같이 그 디스티네이션에 낫레지스터N을 지정하는 것에 의해 낫레지스터N으로의 라이트데이타를 낫레지스터N으로 라이트하는 처리와 함께 그 라이트데이타의 비트반전을 1명령 (LOAD명령) 으로 실행할 수 있다.

제3도에는 인크리먼트 레지스터의 C언어적인 정의 및 그것을 사용한 처리예가 개념적으로 도시된다. 인크리먼트 레지스터 In (Reg[In])은 상기 오퍼레이팅 리드 데이타 레지스터Reg[R]의 범주에 포함되고, 인크리먼트 레지스터 In (Reg [In]의 데이타의 리드가 실행될때 어떤 정수만큼 그 레지스터의 내용을 증가시켜서 재차 인크리먼트 레지스터 In (Reg[In])으로 되돌려 라이트하는 것이다. 제3도의 응용예에서는 인크리먼트 레지스터 In은 LOAD명령①,②의 기본어드레스 지정용의 레지스터로써 사용된다. 이와 같은 응용예에 있어서는 데이타를 메모리에서 레지스터로 라이트할때마다 인크리먼트 레지스터 In의 기본 어드레스가 자동적으로 정수씩 증가되므로 연속되는 메모리영역의 데이타의 리드처리를 통상의 LOAD명령을 사용해서 실행할 수 있다.

제4도에는 수정 레지스터의 C언어적인 정수 및 그것을 사용한 처리예가 개념적으로 도시되어 있다. 수정 레지스터M (Reg[M])는 상기 오퍼레이팅 리드 데이타 레지스터 Reg[R]의 범주에 포함되고, i번째의 명령의 디스플레이스먼트(disp)가 부라면 수정 레지스터M (Reg[M])이 유지하는 값을 리드데이타(read data)로 하고, 디스플레이스먼트(disp)가 0 또는 정이라면 수정 레지스터M(Reg[M])의 값에서 디스플레이스먼트를 뺀 것을 리드데이타 (read data)로 한다. 또, 리드가 실행될때마다 수정 레지스터M (Reg[M])의 내용에 디스플레이스먼트를 가산한 데이타를 재차 수정 레지스터M (Reg[M])으로 되돌려 라이트한다. 이 수정 레지스터M (Reg[M])을 LOAD명령에 있어서의 기본 어드레스지정용의 레지스터로 해서 사용한 응용예가 제4도에 도시된다. 이 예에 있어서는 디스플레이스먼트(disp)가 부인 경우이다. 상술한 수정 레지스터M의 정의에 따라 리드데이타는 수정 레지스터M의 값으로 된다. 그 때문에 이때의 메모리액세스어드레스는 수정 레지스M의 값에 디스플레이스먼트 (disp)를 가산한 값 (Reg[M] + disp)로 된다. 따라서, 이 명령의 실행에 있

어서는 메모리어드레스 (Reg[M] + disp)에서 데이타가 레지스터 t2로 로드된다. 응용예에 있어서의 디스플레이스먼트 (disp)가 정인 경우의 구체적인 예는 도시하고 있지 않지만 상술한 수정 레지스터M의 정의에 따라 리드데이타는 수정 레지스터M의 값에서 디스플레이스먼트 (disp)를 뺀 것으로 된다. 그 때문에 이때의 메모리액세스어드레스는 수정 레지스터M의 값과 같은 값 (Reg[M]-disp+disp)으로 된다. 이것에 위해 스택 또는 FIFO (First-in First-out)의 푸시/팝을 위한 포스트-인크리먼트 (Post-Increment)와 프리-디크리먼트 (Pre-Decrement)의 어드레싱을 통상의 LOAD명령에 의해 실현할 수 있게 된다. LOAD명령②의 경우에는 수정 레지스터M의 값이 전의 명령①의 실행의 결과로써 M+disp로 변경되어 있다. 그 때문에 LOAD명령②의 실행시에는 메모리액세스어드레스가 M+disp+disp로 된다.

제5도에는 수정 인덱스레지스터(4)의 C언어적인 정의가 도시되어 있다. 수정 인덱스레지스터Mi (Reg[Mi])는 제3도의 인크리먼트레지스터 In (Reg [In])의 오퍼레이션으로 정수(Const)대신에 i번째의 명령으로 지정된 레지스터Reg [r]의 값을 사용하는 것이다. 제3도의 응용예에 수정 인덱스레지스터Mi (Reg [Mi])를 사용하는 경우 미리 Reg [r]에 적당한 값을 넣어두는(예를 들면 다른 LOAD 명령으로 미리 적당한 값을 Reg [r]로 로드해두는) 것에 의해 임의의 값(Reg [r]로 설정한 값)으로 기본어드레스의 증가를 실행할 수 있다.

제5도에는 또 얼라인먼트 레지스터의 C언어적인 정의 및 그것을 사용한 처리예가 개념적으로 도시되어 있다. 얼라인먼트 레지스터AL (Reg[AL])은 상기 오퍼레이팅 라이트 데이타 레지스터 Reg[W]의 범주에 포함되고, 인덱스로 지정되는 마스크패턴과 얼라인먼트 레지스터AL (Reg[AL])로의 라이트데이타의 각 비트 마다의 논리곱을 채용하는 것이며, 그 논리곱의 처리를 한후의 데이타를 얼라인먼트 레지스터AL (Reg[AL])로 라이트한다. 예를들면 응용예 1 과 같이 마스크패턴 (AL0)이 상위 하프워드는 전부 0, 하위 하프워드는 전부 1로 정의되어 있는 것으로 하면 LOAD명령의 디스티네이션으로써 지정된 얼라인먼트 레지스터AL0에는 하위 하프워드에는 메모리의 대응하는 비트의 값이, 상위 하프워드에는 전체 비트 0 의 데이타가 라이트된다. 이 얼라인먼트 레지스터AL0은 제6도에 도시되는 응용예2와 같이 디스티네이션의 레지스터 t 로 데이타를 라이트할때의 중간버퍼로써 사용할 수도 있다. 제6도에 있어서, 그 중간버퍼는 Ag0, 마스크패턴은 (Ag0)으로써 표시되어 있다. 따라서, 응용예 1,2로 나타낸 바와 같이 얼라인먼트 레지스터AL을 라이트데이타의 디스티네이션(또는 중간버퍼)로써 지정하는 것에 의해 그 라이트데이타, 마스크패턴의 논리곱의 처리 및 얼라인먼트 레지스터AL (레지스터 t)로의 라이트처리가 1명령(LOAD명령)으로 실행할 수 있다. 특히 제한되지 않지만 마스크패턴은 얼라인먼트 레지스터에 대응하고 있다. 예를들면 상기 얼라인먼트 레지스터AL0에는 상기 마스크패턴(AL0)이 대응하고 있고, 상기 얼라인먼트 레지스터Ag0에는 상기 마스크패턴(Ag0)이 대응하고 있다.

또, 메모리액세스어드레스는 범용레지스터b {GR [b]}의 값과 디스플레이스먼트(disp)의 합에 의해 얻어진다.

제7도에는 자리올림/빌림수 (Carry/Borrow)레지스터의 C언어적인 정의 및 그것을 사용한 처리예가 개념적으로 도시되어 있다. 자리올림/빌림수레지스터C/B(Reg [C/B])는 상기 오퍼레이팅 라이트 데이타 레지스터 Reg[W]의 범주에 포함되고, 자리올림/빌림수레지스터 C/B (Reg[C/B])로의 라이트데이타(write data)에 프로그램 상태 워드PSW(Program Statas Word)이 자리올림이나 빌림수의 상태를 나타내는 비트, 즉PSW [C/B]의 값을 가산해서 자리올림/빌림수레지스터 C/B (Reg[C/B])로 라이트하는 것이다. 응용예와 같이 가감산명령 (ADD명령등)의 디스티네이션을 자리올림/빌림수레지스터 C/B/에 지정하는 것에 의해 1명령의 실행으로 2개의 레지스터 a,b의 내용의 가감산결과에 또 전회의 연산으로 발생한 자리올림 또는 빌림수의 값 PSW[C/B]를 가산할 있다.

제8도에는 시프트 레지스터Reg [ni]의 C언어적인 정의 및 그것을 사용한 처리예가 개념적으로 도시되어 있다. 시프트 레지스터 ni (Reg[ni])는 상기 오퍼레이팅 리드 데이타 레지스터Reg [R]의 범주에 포함되고, 시프트 레지스터 ni (Reg [ni])의 내용을 i비트만큼 좌측으로 시프트하고, 리드데이타 (read data)로써 출력한다. 제8도의 응용예에서는 ADD명령이 도시되어 있다. 이 명령을 실행하면 시프트 레지스터 n2에서 리드된 데이타는 소정의 비트, 소정의 방향으로 시프트(예를들면 2비트 왼쪽으로 시프트)된후에 레지스터 b의 값과 가산되어 그 가산 결과의 데이타가 디스티네이션의 레지스터 t로 라이트된다.

제9도에는 제2도~제8도에서 설명한 기능을 갖는 레지스터를 사용한 데이타처리장치의 1실시예의 블럭도가 도시되어 있다. 제9도에 도시되는 데이타처리장치는 공지의 반도체집적회로 제조기술에 의해서 실리콘과 같은 1개의 반도체기판에 형성되고, 명령제어부(1), 연산부(2), 명령캐시메모리(3) 및 데이타캐시메모리(4)가 대표적으로 도시되어 있다.

제9도의 연산부(2)에 도시되는 E3은 제2도~제8도에서 설명한 각종 기능을 갖는 레지스터의 집합 (이하, 단지 기능을 갖는 레지스터라고도 한다)이다. 연산부(2)에 있어서 기능을 갖는 레지스터E3의 기능실현에 전용화 된 회로블럭으로써 특히 제한되지 않지만 셀렉터 E1, 가산기 E2 및 반전회로E4가 새로이 마련된다. 마스크회로E5, 셀렉터E6 및 프리시프터E7은 데이타처리장치에는 통상 마련되어 있음과 동시에 상기 기능을 갖는 레지스터E3의 기능 실현에도 이용되는 회로블럭이다. 연산부(2)에는 그밖에 여러개의 레지스터로 이루어지는 범용레지스터E8, 프로그램 카운터(PC) E10, 산술논리연산기(ALU)E9 및 전회의 연산으로 발생한 자리올림 또는 빌림수의 값PSW [C/B]등을 유지하고 있는 프로세서 상태 워드레지스터(PSW)E11등이 마련되어 있다. B1, B2, B3, B4는 연산부(2)에 있어서 대표적으로 도시된 내부버스이고, DV0~DV9는 바이패스용 드라이버이다. 또, 기능을 갖는 레지스터내의 얼라인먼트 레지스터는 AL0이외에 AL1이 있고, 각각의 얼라인먼트 레지스터에 대응해서 마스크회로의 마스크패턴이 후술하는 제어회로S6에 의해 제어된다.

명령제어부(1)은 명령 캐시메모리(3)에서 페치한 명령을 해독해서 대표적으로 도시된 각종의 제어신호 S1~S17등의 제어신호를 소정의 타이밍으로써 연산부(2)등에 공급한다. 명령제어부(1)에는 오퍼랜드 디코더 I 1, 연산코드디코더I 2, 타겟디코더 I 3 및 제어블럭I 4, I 5가 포함된다. 연산코드디코더 I 2는 명령에 포함되는 연산 코드를 해독한다. 명령의 그밖의 필드는 특히 제한되지 않지만 연산코드의 종류에 따라서 오퍼랜드 디코더 I 1이나 타겟디코더 I 3에 공급된다. 오퍼랜드디코더 I 1은 예를들면 명령의 오퍼랜드 또는 소오스 지정영역을 해독한다. 이 영역에 기능을 갖는 레지스터로써의 상기 오퍼레이션 리드 데이타 레지스터Reg [R]이 지정되어 있는 경우에는 상기 레지스터의 지정이 해독된다. 타겟디코더 I 3은 예

를 들면 명령의 디스티네이션 또는 타겟지정영역을 해독한다. 이 영역에 기능을 갖는 레지스터로써의 상기 오퍼레이션 라이트 데이타 레지스터Reg[W]가 지정되어 있는 경우에는 상기 레지스터의 지정이 해독된다. 오퍼랜드디코더 I 1 및 타겟디코더 I 3에 의한 해독결과는 범용레지스터E8이나 기능을 갖는 레지스터E3의 지정에 이용된다. 또, 오퍼랜드디코더 I 1 및 타겟디코더 I 3에 의한 해독결과는 연산코드 디코더의 해독결과와 함께 제어블럭 I 4에도 공급되어 셀렉터 E1, 마스크회로E5, 프리시프터 E7의 제어 및 자리올림/빌림수제어, 그리고 바이패스용 드라이버 DV0~DV9의 개폐제어 등에 이용된다.

여기에서 제10도를 참조하면서 범용레지스터E8이나 기능을 갖는 레지스터E3의 지정방법에 대해서 설명한다. 쌍방의 레지스터는 모두 명령중에 있어서의 오퍼랜드 및 그밖의 정보를 위한 지정필드에서 지정된다. 실제로 명령중의 어느 영역, 즉 제 몇번째비트 부터 제 몇번째비트까지를 지정필드로써 이용하는가는 연산코드의 종류에 의해서 미리 규정되어 있다. 예를들면 제10도에 도시되는 바와 같이 LOAD명령이 연산코드, 오퍼랜드, 디스티네이션, 디스플레이스먼트(disc)의 각 필드를 가질때 오퍼랜드 또는 디스티네이션의 영역에서 레지스터가 지정된다. 예를들면 범용레지스터E8과 기능을 갖는 레지스터E3에 포함되는 레지스터가 전부 2^n개일때에는 n비트의 정보에 의해서 순번으로 레지스터번호를 지정할 수 있다. 또, 제10도에 도시되어 있는 바와 같이 예를들면 범용레지스터E8이 8개의 범용레지스터GEReg1~GEReg8을 포함하고, 기능을 갖는 레지스터E3이 7개의 기능을 갖는 레지스터 OPReg1~OPReg7을 포함할때 각각의 레지스터에는 제10도에 도시되는 4비트B3, B2, B1, B0으로 규정되는 번호를 할당 할 수 있다. 이와 같이 규정한 경우에는 레지스터지정정보 B3~B0의 최상위비트B3의 '0'은 범용레지스터E8을 지정하는 비트로 간주되고, 레지스터지정정보의 최상위비트B3의 '1'은 기능을 갖는 레지스터E3을 지정하는 비트로 간주된다. 따라서, 이와 같이 레지스터번호의 할당을 실행한 경우 명령포맷중의 레지스터 지정영역에는 범용 레지스터를 지정하는 것인가 도는 기능을 갖는 레지스터를 지정하는 것인가를 지정하는 지정비트필드(B3)가 존재하게 된다.

프로그램카운터E10으로 지정된 어드레스에 있는 명령은 명령캐시 메모리(3)에서 리드되어 명령제어부(1)로 전송된다. 명령제어부(1)은 명령중의 연산코드 및 오퍼랜드 등을 디코드하고, 그 디코드결과에 따른 각종 제어신호등으로 연산부(2)의 동작을 제어한다.

제11도 및 제12도에는 제2도~제8도에 도시되는 각종 기능을 갖는 레지스터에 할당되어 있는 오퍼레이션을 제9도의 구성으로 실행할때의 제어형태가 파이프라인형식으로 도시되어 있다. 동일 도면에 도시되는 파이프라인처리는 명령페치, 명령디코드, 연산, 메모리 액세스 및 레지스터라이트/메모리저장의 5단계의 파이프라인 스테이지로 구성되고, 기능을 갖는 레지스터를 지정한 처리마다 어느 스테이지에서 어떤 처리가 실행되는지가 도시되어 있다. 동일 도면에 있어서 실선화살표는 데이타의 흐름, 점선화살표는 어서트된 제어 신호 또는 어드레스신호의 흐름을 의미한다. 그리고, 신호에 붙여진 부호 및 블럭내의 부호는 제9도에 도시되는 부호에 대응하고 있고, 제2도~제8도에서 설명한 각종 기능을 갖는 레지스터에 할당되어 있는 오퍼레이션의 1예를 제9도의 구성에 따라서 설명한다.

LOAD명령등의 실행시에 명령디코드 스테이지에 있어서 타겟디코더 I 3이 디코드한 레지스터번호가 낫레지스터N인 경우는 마스크회로E5를 거쳐서 리드되는 데이타캐시메모리(4)의 리드데이타 또는 내부버스 B2(이하, 라이트버스라고도 한다)로 리드되어 있는 데이타의 비트값이 레지스터 라이트/메모리저장 스테이지에서 비트값을 반전시키는 반전회로E4에 의해 반전시켜지고 레지스터선택제어신호S7에 의해 선택되는 낫레지스터N으로 라이트된다. (제11도(1),(2)).

LOAD명령등의 실행시에 명령디코드 스테이지에 있어서 타겟디코더 I 3이 디코드한 레지스터번호가 얼라인먼트 레지스터AL0 또는 AL1인 경우는 타겟디코더 I 3의 디코드값에 의해 지정된 얼라인먼트 레지스터AL0 또는 AL1에 대응하는 마스크회로E5의 마스크패턴이 상기 디코드값에 따라서 제어블럭 I 4에서 출력되는 제어신호 S6에 의해 선택된다 (제12도(11)). 그리고, 레지스터 라이트/메모리저장 스테이지에 있어서 데이타캐시메모리(4)의 리드데이타, 내부버스B4(이하, 어드레스버스라고도 한다)로 리드외어 있는 데이타 또는 내부버스B5 (이하, 라이트버스라고도 한다)로 리드되어 있는 데이타와 선택된 마스크패턴이 마스크회로E5에 의해 논리곱이 취해지고, 그 논리곱이 실시된 데이타, 즉 마스크처리 (이하, 얼라인먼트처리라고도 한다) 가 실시된 데이타가 레지스터선택 제어신호S7에 의해 선택되는 얼라인먼트 레지스터AL0 또는 AL1로 라이트된다 (제12도(13)). 또, 얼라인먼트 레지스터AL0 또는 AL1을 제6도의 응용예2와 같이 중간버퍼로써 사용하는 경우 얼라인먼트 레지스터AL0 또는 AL1로 라이트된 데이타를 또 다른 레지스터, 예를들면 범용레지스터E8내의 소정의 레지스터등으로 내부버스B1 (이하, 소오스버스라고도 한다), B2등을 거쳐서 라이트한다.

오퍼랜드디코더 I 1이 디코드한 레지스터번호가 기능을 갖는 레지스터E3에 포함되는 레지스터를 지정하는 것으로써 그 레지스터가 상기 시프트 레지스터 (ni)라면 데이타는 i에 대응하는 비트수만큼 데이타를 좌측으로 시프트하도록 프리시프터E7이 제어신호S1에 의해 제어된다. 즉, 제12도에 도시되는 바와 같이 명령디코드스테이지에 있어서 오퍼랜드의 디코드가 실행되어 상기 시프트 레지스터ni의 지정이 해독되면 상기 스테이지에 있어서 레지스터선택제어신호S7에 의해 선택되는 기능을 갖는 레지스터E3에 포함되는 시프트 레지스터ni에서 데이타가 내부버스B1로 리드된다 (제12도(14), (15)). 다음의 연산스테이지에서는 오퍼랜드의 디코드에 의해서 얻어진 제어신호S1에 의해서 프리스프터E7이 제어되어 상기 리드데이타에 대한 시프트동작이 실행된다. 시프트된 데이타의 처리는 또 상기 시프트 레지스터를 지정한 명령의 연산코드에 따라서 처리된다 (제12도(16)). 제8도의 응용예의 경우이면 시프트처리된 데이타는 산술논리연산기E9에 입력된다. 그리고, 연산스테이지에서 산술논리연산기E9에 의해 범용레지스터E8내의 레지스터b에 저장되어 있는 데이타와 가산처리가 실시되어 레지스터라이트/메모리저장 스테이지에서 범용레지스터E8내의 레지스터 t에 그 가산처리결과의 데이타가 저장된다 (제12도(16), (17)).

또, 명령디코드 스테이지에서 오퍼랜드디코더 I 1 이 디코드한 레지스터번호가 기능을 갖는 레지스터E3에 포함되는 인크리먼트 레지스터 In을 나타내는 경우 레지스터선택 제어신호S7에 의해 선택되고, 상기 레지스터에서 리드된 데이타는 내부버스B1 및 프리시프터 E7을 통해서 산술논리연산기E9로 전송됨과 동시에 내부버스B1을 통해서 가산기E2로 전송된다 (제11도 (3), (5)). 이 가산기E2의 다른쪽의 입력에는 오퍼랜드디코더 I 1에서 해독된 인크리먼트 레지스터 In의 선택제어신호S3에 의해서 셀렉터E1에 의해 고정값+1이 선택되어 공급된다. 이것에 의해서 연산스테이지에서 상기 인크리먼트 레지스터In의 리드값에 1이 가산된다(제11도(8)). 그리고, 레지스터라이트/메모리저장 스테이지에서 그 저장된 데이타가 라이트버스B5

를 통해서 재차 인크리먼트 레지스터 In으로 되돌려져 라이트된다 (제11도(9)). 인크리먼트 레지스터In의 값이 다음 스테이지에서 사용될때에는 제어블럭 I 4에서 그것을 검출하고, 바이패스용 드라이버DV0 또는 DV1을 제어신호S4 또는 S5를 어서트하는 것에 의해 인에이블상태 (열린상태)로 해서 가산기E2의 가산결과의 데이타를 소오스버스B1로 직접 출력해둔다(제11도(10)). 제3도의 응용예의 경우 산술논리연산기E9에는 인크리먼트 레지스터In의 리드데이타외에 프리시프터E7을 거쳐서 디스플레이스먼트disp가 공급되고 있다. 산술논리연산기E9는 이것을 가산하는 것에 의해 데이타캐시메모리(4)의 어드레스계산을 한다(제11도(6)). 그후의 스테이지(제11도(4), (7))에 대해서는 수정 레지스터M의 동작설명시에 설명한다.

명령디코드의 스테이지에서 오퍼랜드디코더 I 1이 디코드한 레지스터번호가 기능을 갖는 레지스터E3에 포함되는 상기 수정 레지스터M을 나타내는 경우 선택제어신호S3이 셀렉터E1의 출력으로써 디스플레이스먼트disp를 선택해서 가산기E2의 한쪽의 입력에 디스플레이스먼트disp를 공급하고, 레지스터선택 제어신호S7에 의해 선택되는 상기 수정 레지스터의 데이타가 내부버스B1을 거쳐서 가산기E2의 다른쪽의 입력에 공급된다. (제11도(3), (5)). 연산스테이지에서 가산기E2는 수정 레지스터M의 데이타와 디스플레이스먼트disp를 가산한다(제11도(8)). 그리고, 레지스터라이트/메모리저장 스테이지에서 그 가산된 데이타가 내부버스B5를 통해서 재차 수정 레지스터M으로 되돌려져 라이트된다 (제11도(9)). 다음 스테이지에서 수정 레지스터M의 값이 사용될때는 제어블럭 I 4에서 그것이 검출되고, 바이패스용 드라이버DV0 또는 DV1이 제어신호S4 또는 S5에 의해 인에이블상태로 되고, 가산기E2의 출력값이 직접 소오스버스B1로 출력된다(제11도(10)). 또, 수정 레지스터M에서 리드된 데이타는 내부버스B1을 거쳐서 프리시프터E7로도 전송되고 있다. 이때, 제어블럭 I 4는 상기 디스플레이스먼트disp가 부인지 0 또는 정인지를 판정하고, 또 연산코드 디코더 I 2의 해독결과에 따라서 연산코드로 규정되는 명령이 LOAD명령인지 아닌지를 판정한다. 그리고, 제어블럭 I 4는 상기 디스플레이스먼트disp의 판정결과와 상기 명령의 판정결과에 따라서 제어신호S1을 프리시프터E7로 출력한다. 프리스프터E7은 산술논리연산기E9의 한쪽의 입력에 수정 레지스터M에서 리드된 데이타를 공급한다. 상기 명령의 판정결과가 LOAD명령인 경우 다음에 프리시프터E7은 제어신호S 1에 따라서 디스플레이스먼트disp가 정 또는 0 이라면 0을, 부라면 디스플레이스먼트disp를 선택해서 산술논리연산기E9의 다른쪽의 입력에 공급한다.

또, 연산코드 디코더 I 2가 디코드한 명령이 LOAD명령이고, 그리고 오퍼랜드디코더 I 1이 디코드한 레지스터번호가 수정 레지스터M이외인 경우 프리스프터E7은 제어신호S 1에 따라서 어드레싱의 종류에 의해서 데이타를 선택하여 (인덱스어드레싱시는 소오스버스B 1의 데이타를, 디스플레이스먼트 어드레싱시는 디스플레이스먼트disp를 선택해서) 산술논리연산기E9로 전송한다. 연산스테이지에서 데이타캐시메모리(4)의 어드레스가 수정 레지스터M에서 리드된 데이타와 디스플레이스먼트disp 또는 0을 산술논리연산기E9가 가산하는 것에 의해 계산된다 (제11도(6)). 이 어드레스계산은 제4도의 응용예를 실현하는 것으로 된다. 즉, 산술논리연산기E9의 출력의 값은 수정 레지스터M의 리드된 데이타를 M'로 해서 표시하면 디스플레이스먼트disp가 부일때 'M'+disp'이고, 정 또는 0일때 'M''이고, 응용예의 메모리액세스 어드레스와 내용적으로는 동일하다. 산술논리연산기E 9에서 계산된 어드레스는 어드레스신호로써 어드레스버스B 4로 전송되고, 이것에 의해 데이타캐시메모리(4)에서는 그 어드레스에 저장되어 있는 데이타가 리드데이타로써 마스크회로E5로 출력된다(제11도(4)). 레지스터라이트/메모리저장 스테이지에서 마스크회로E5는 제어신호 S 6에 의해 제어된다. 타겟디코더 I 3의 디코드에 의해 얻어진 라이트지가 범용레지스터E8이나 얼라인먼트 레지스터이와의 오퍼레이팅 레지스터인 경우 제어신호S 6에 의한 제어에 의해서 마스크회로E5는 얼라인먼트처리를 실행하지 않고 상기 라이트지의 레지스터로 상기 리드데이타를 라이트한다. 이것에 대해서 라이트지가 얼라인먼트 레지스터AL0이라면 마스크회로E5는 그 레지스터AL0에 대응한 마스크패턴의 얼라인먼트처리를 상기 리드데이타에 대해서 실행하고, 얼라인먼트 레지스터AL0 또는 얼라인먼트 레지스터AL0을 중간버퍼로 해서 그밖의 레지스터로 데이타를 라이트한다(제11도(17)).

오퍼랜드디코더 I 1이 디코드한 레지스터번호가 수정 인덱스레지스터 Mi를 나타내는 경우 제어신호 S 3이 셀렉터E1의 출력으로써 소오스버스B 1을 선택하는 것에 의해 수정 인덱스레지스터Mi의 데이타와 임의의 레지스터의 데이타의 가산이 연산스테이지에 있어서 가산기E2에서 실행된다. 이후의 동작은 인크리먼트레지스터In 일때와 동일하다.

산술연산시의 데이타의 라이트지로써 타겟디코더 I 3의 디코드(명령 디코드 스테이지에서)한 레지스터번호가 자리올림/빌림수레지스터C/B를 지정하면 제어블럭 I 4가 그것을 검출해서 제어신호S 2에 의해 산술논리연산기E9의 최하위비트의 자리올림입력으로써 전회의 연산시의 자리올림 또는 빌림수의 값PSW[C/B]의 데이타가 선택된다(제12도(18)). 연산스테이지에서 산술논리연산기E9는 2개의 데이타와 PSW[C/B]의 데이타를 가산한다(제12도(19)). 그 가산결과의 데이타는 내부버스B 2를 통해서 자리올림/빌림수레지스터C/B로 라이트된다(제12도(20)). 또, 데이타의 라이트지가 자리올림/빌림수레지스터C/B이외인 경우는 셀렉터E6의 출력은 00이 선택된다.

제13도에는 상기 마스크회로E5의 1예가 도시된다. 또, 제13도중의 S6a~S6c는 상기 제어신호S 6에 포함되는 제어신호이고, G0a~G23a, G1b~G23b 및 G0c~G23c는 각각 제어신호S6a~S6c에 의해 개폐제어되는 게이트회로이다. 제13도에 도시되는 예는 레지스터의 비트수가 최대32비트인 경우이고, 입력에 대해서 마스크하지 않고 출력하는 형태, 입력의 상위3바이트(비트00~비트23)를 마스크해서 출력하는 형태, 입력의 상위하프워드(비트00~비트16)을 마스크해서 출력하는 형태를 갖고, 각 형태는 제어신호 S6a, S6b, S6c에 의해서 택일적으로 선택된다. 즉, 비트24~비트31은 입력에서 출력에 이르는 스루의 신호경로를 갖는다. 비트00~비트23에는 E5a의 영역에 표시되는 바와 같이 제어신호S6a로 개폐제어되는 게이트회로G0a~G23a를 거쳐서 입력을 선택적으로 출력에 전달하는 신호경로와 E5b의 영역에 표시되는 바와 같이 제어신호S6b로 개폐제어되는 게이트회로G0b~G23b를 거쳐서 논리값'0'의 신호비트를 선택적으로 출력에 전달하는 신호경로가 마련된다. 비트00~비트23에는 E5c-1의 영역에 표시되는 바와 같이 제어신호S6c로 개폐제어되는 게이트회로G0c~G15c를 거쳐서 논리값'0'의 신호비트를 선택적으로 출력에 전달하는 신호경로 및 E5c-2의 영역에 표시되는 바와 같이 제어신호S6으로 개폐제어되는 게이트회로 G16c~G23c를 거쳐서 입력을 선택적으로 전달하는 신호경로가 마련된다. 상기 각각의 게이트회로는 특히 제한되지 않지만 클럭 인버터와 같은 3상태 출력회로를 포함해서 구성된다. 타겟 레지스터로써 얼라인먼트 레지스터AL0 및 AL1 이 지정되지 않을때(통상시)는 제어신호S6a만 어서트되어 E5a에 포함되는 게이트회로G0a~G23a가 ON상태로 되고, 입력신호는 그대로 대응하는 비트로 출력된다. 타겟 레지스터로써 하위 1바이트만 '1'인 마스크패턴을 갖는 얼라인먼

트 레지스터AL0 또는 AL1이 선택되었을때는 제어신호S6B만이 어서트되어 E5b에 포함되는 게이트회로 G0b~B23b가 ON 상태로 되고, 입력신호의 하위 1 바이트가 대응하는 비트로 출력되고, 그밖의 상위 3바이트에는 '0'이 출력된다. 타겟 레지스터로써 하위 하프워드만 '1'인 마스크패턴을 갖는 얼라인먼트 레지스터AL1 또는 AL0이 선택되었을때는 제어신호S6c만이 어서트되어 E5c-1 및 E5c-2에 포함되는 게이트회로 G0c~G23c가 ON상태로 되고, 입력신호의 하위 하프워드가 대응하는 비트로 출력되고, 그밖의 상위 하프워드에는 '0'이 출력된다.

제14도에는 제9도에 도시되는 데이타처리장치의 칩평면이 개략적으로 도시된다. 제14도에 있어서, (3)은 명령캐시메모리(CC), (4)는 데이타캐시메모리(DC), (2)는 연산부(EU), (1)은 명령제어부(IU), (9)는 메모리제어용 랜덤논리(MU) 및 시스템버스제어부(PU), (6)은 명령용 태그 캐시메모리(CA), (4)는 데이타용 태그 캐시메모리(DC), (5)는 명령용 어드레스변환버퍼 (CT), (7)은 데이타용 어드레스변환버퍼(DT), (10)은 입출력부 (I/O)이다.

제15도에는 기능을 갖는 레지스터를 서포트하고 있지 않은 데이타처리장치와 본 발명에 관한 기능을 갖는 레지스터를 서포트하는 데이타처리장치의 파이프라인 스테이지로 구분한 동작흐름도의 1예가 도시되어 있다. 기능을 갖는 레지스터를 서포트하고 있지 않은 경우에는 명령페치사이클에서 제9도의 명령캐시메모리(3)으로 부터 명령을 페치하고, 디코드사이클에서 페치한 명령의 연산코드 디코드와 오퍼랜드디코드를 실행하고, 오퍼랜드디코드에 의해서 표시되는 레지스터의 내용을 범용레지스터에서 산술논리연산기로 전송한다. 연산사이클에서는 연산코드 디코드로 지시된 연산을 페치한 레지스터의 내용에 대해서 산술논리연산기가 실행한다. 지시된 연산이 LOAD 또는 STORE명령이라면 이어지는 메모리액세스사이클에서 액세스하는 메모리의 어드레스와 (STORE시는) 라이트하는 데이타를 데이타캐시메모리로 전송한다. 그리고, 레지스터라이트/메모리저장 사이클에서 연산한 데이타를 범용레지스터 또는 데이타캐시메모리로 라이트한다. 명령이 데이타의 얼라인먼트를 지시하는 것이라면 이 사이클에서 그것을 실행한후에 라이트를 실행한다.

기능을 갖는 레지스터를 서포트하는 경우에는 디코드사이클에서 디코드한 오퍼랜드가 인크리먼트레지스터 In, 수정 레지스터M, 인크리먼트 수정 레지스터Mi일때 다음의 연산스테이지에서 어드레스 수정 인크리먼트를 실행한다. 또, 디코드한 오퍼랜드가 시프트 레지스터ni인 경우 연산사이클에서 연산 메모리어드레스 계산을 실행할때 미리 i비트만큼 데이타를 프리시프트한다.

데이타의 라이트지의 레지스터로써 얼라인먼트 레지스터AL0 또는 AL1, 낫레지스터N이 지정되었을때는 레지스터라이트/메모리저장 스테이지에서 데이타의 마스크 또는 비트반전을 실행하고나서 데이타라이트를 실행한다. 또, 라이트지의 레지스터가 자리올림/빌림수레지스터C/B이라면 연산스테이지에서 연산을 실행할때 자리올림/빌림수의 값을 산술논리연산기 E9의 최하위비트에 입력한다.

상기 실시예에 의하면 다음의 작용효과가 있다.

(1) 본 실시예에 관한 기능을 갖는 레지스터를 서포트하는 데이타처리장치에서는 상술한 다양한 어드레싱모드나 복합명령 등을 레지스터에 기능을 갖게하는 것에 의해 실행한다. 즉, 명령수를 증가시키는 대신에 기능을 갖는 레지스터를 마련하고, 연산코드 영역의 비트의 증가를 오퍼랜드영역과 타겟영역의 비트수로 흡수한다. 이것에 비해서 기능을 갖는 레지스터를 서포트하고 있지 않은 데이타처리장치에서는 어드레싱의 방식을 다양하게 하거나 몇개인가의 명령을 조합한 복합명령을 추가해서 CISC적인 기능을 기본적인 명령세트에 갖게하고자 하면 명령수가 증가하는 것으로 되어 명령포맷중의 연산코드필드의 비트수가 증가해 버려 연산코드 디코더의 논리규모가 커지고 명령디코드시간의 증가, 즉 데이타처리장치의 동작주파수를 저하시켜버리게 된다. 이 차이는 제16도에서도 명확한 바와 같이 본 실시예의 데이타처리장치에서는 기능확장시에 연산코드영역의 비트수 및 연산코드디코더의 논리규모의 증대는 전혀 없다. 이것에 대해서 기능을 갖는 레지스터를 서포트하지 않는 방식에서는 연산코드영역의 비트수 및 연산코드디코더의 논리규모는 현저하게 증대한다.

(2) 기본명령으로써의 연산코드와 오퍼랜드 유지영역으로써의 기능을 갖는 레지스터의 조합에 의해 다양한 기능을 실현할 수 있으므로, 1개의 기능을 갖는 레지스터를 마련하면 여러개의 명령 또는 연산코드와 조합해서 사용할 수 있게 되어 여러 기능을 실현할 수 있다. 예를들면 로드 앤드 인크리먼트, 인크리먼트 및 스토어 앤드 인크리먼트의 기능을 부가하고자할때 기능을 갖는 레지스터가 없는 방식에서는 그들에 각각 대응한 명령 또는 연산코드를 추가하게 되지만 본 발명에서는 인크리먼트 레지스터를 1개 추가하는 것만으로 된다. 따라서, 데이타처리장치에 추가한 기능의 수에 비해서 추가해야할 기능을 갖는 레지스터의 수는 비교적 작은 수로 되어 오퍼랜드영역과 타겟영역의 비트수의 증가가 그다지 크게 되지 않는다. 따라서, 기능을 갖는 레지스터에 의해서 데이타처리장치의 기능을 확장해도 명령의 오퍼랜드나 타겟의 지정영역의 비트수, 그리고 오퍼랜드디코더 및 타겟디코더의 논리규모의 증대를 최소한으로 억제할 수 있다.

(3) 제2도~제8도의 오퍼레이팅 레지스터를 사용해서 상술한 PA-RISC칩으로 정의되는 명령과 동일한 기능을 실현한 경우의 명령수의 삭감을 본 발명자가 시험적으로 계산한 결과 예를들면 전체 명령세트 121명령중 상기 각종 기능을 갖는 레지스터를 사용한 효과에 의해 26명령을 삭감할 수 있다. 이것은 원래의 명령수의 21%에 해당한다.

이상 본 발명자에 의해서 이루어진 발명을 실시예에 따라서 구체적으로 설명했지만 본 발명은 그것에 한정되는 것은 아니고, 그 요지를 이탈하지 않는 범위에서 여러가지로 변경가능한 것은 물론이다. 예를들면 기능을 갖는 레지스터의 종류는 제2도~제8도에서 설명한 것에 한정되지 않고 그밖의 기능을 실현하도록할 수도 있다. 이상의 설명에서는 주로 본 발명자에 의해서 이루어진 발명을 그 배경으로 된 이용분야인 범용레지스터방식의 데이타처리장치에 적용한 경우에 대해서 설명했지만 본 발명은 그것에 한정되는 것은 아니고 어큐뮬레이터방식의 데이타처리장치 등에도 적용할 수 있다. 또, 전용적인 처리에 단지 적용되는 데이타처리장치의 경우에도 범용레지스터를 필요로 하지 않게 할 수 있다.

본원에서 개시되는 발명중 대표적인 것에 의해서 얻을 수 있는 효과를 간단히 설명하면 다음과 같다.

(1) 레지스터에 대한 데이타의 라이트 또는 리드시에 그 데이타에 대해서 특정한 처리를 실시하는 기능을 미리 레지스터(기능을 갖는 레지스터)에 할당하고, 그 기능을 갖는 레지스터를 명령중에서 지정하는 것에 의해 데이타에 대해서 명령의 연산코드에 의해서 규정되는 명령의 처리와 기능을 갖는 레지스터가 갖는

특정기능을 조합한 다양한 데이타처리를 실현할 수 있다. 즉, 명령수의 증가를 억제해서 어드레싱모드나 복합명령에 의한 처리와 동등한 기능을 실현할 수 있다.

(2) 명령의 연산코드에 의해서 규정되는 명령의 처리와 기능을 갖는 레지스터가 갖는 특정기능을 조합해서 다양한 데이타처리를 실현하는 것에의해 명령세트의 명령수를 증가하는 일없이 즉 연산코드의 비트수를 증대시키는 일없이 데이타처리장치에 의한 처리를 다양화 또는 다기능화할 수 있어 연산코드의 디코드 논리로 대표되는 명령제어부의 하드웨어의 증대를 극력 억제해서 다양한 기능을 실현 할 수 있다. 이것에 의해 또 명령제어부의 명령디코더등의 랜덤논리의 논리규모가 크게 되지 않아 LSI의 성능저하도 방지할 수 있다.

(3) 상기에 의해서 실현되는 다양한 기능은 1개의 명령중의 오퍼랜드지정영역 등에서의 기능을 갖는 레지스터의 지정으로 실행되며, 또한 그 기능을 갖는 레지스터에 할당되는 고유의 기능은 상기 레지스터에 대한 데이타의 라이트 또는 리드시에 실행되므로, 기능확장에 따라서 실행해야할 명령수가 증가하지 않고, 이것은 고기능인처리를 적은 스텝수로써 실현할 수 있는 것도 의미한다.

(4) 기본명령으로써의 연산코드와 오퍼랜드유지영역으로써의 기능을 갖는 레지스터의 조합에 의해 다양한 기능을 실현할 수 있으므로, 1개의 기능을 갖는 레지스터를 마련하면 여러 명령 또는 연산 코드와 조합해서 사용할 수 있게 되어 여러 기능을 실현할 수 있다. 따라서, 데이타처리장치에 추가한 기능의 수에 비해서 추가해야할 기능을 갖는 레지스터의 수는 비교적 작은 수로 되어 오퍼랜드영역과 타겟영역의 비트수의 증가가 그다지 크게 되지 않는다. 이것에 의해 기능을 갖는 레지스터에 의해서 데이타처리장치의 기능을 확장해도 명령의 오퍼랜드나 타겟의 지정영역의 비트수, 그리고 오퍼랜드디코더 및 타겟디코더의 논리규모의 증대를 최소한으로 억제할 수 있다.

(57) 청구의 범위

청구항 1

각각이 연산코드부와 데아타부를 포함하는 여러개의 단일명령을 포함하는 여러개의 명령에 응답해서 처리수속을 실행하는 데이타처리장치로서, 각각 서로 다른 미리 결정된 처리수속이 할당되고 단일명령의 데이타부에 의해서 하나의 기능레지스터가 특정되는 여러개의 기능 레지스터, 단일 명령의 데이타부에 응답해서 기능레지스터로 액섹스하는 액세스수단 및 연산코드부와는 독립으로 기능레지스터로의 라이트데이타의 라이트를 위한 라이트액세스 및 기능레지스터로부터의 리드데이타의 리드를 위한 리드액세스중의 하나에 응답해서, 기능레지스터에서 리드될 리드데이타 및 기능레지스터로 라이트될 라이트데이타 중의 하나에 대해 특정된 기능레지스터에 할당된 미리 결정된 처리수속을 실행하는 처리수단을 포함하는 것을 특징으로 하는 데이타처리장치.

청구항 2

제1항에 있어서, 상기 처리수단은 단일명령의 데이타부에 의해서 특정된 기능레지스터에 할당된 미리 정해진 처리수속의 실행에 부가해서 상기 단일명령의 연산코드부에 따른 처리수속을 실행하는 수단을 포함하는 것을 특징으로 하는 데이타처리장치.

청구항 3

제2항에 있어서, 상기 처리수단은 상기 여러개의 기능레지스터 중의 하나로의 라이트액세스 또는 리드액세스에 응답해서 그 하나의 기능레지스터에 할당된 미리 정해진 처리수속을 실행하는 수단을 포함하는 것을 특징으로 하는 데이타처리장치.

청구항 4

제1항에 있어서, 상기 처리수단은 상기 여러개의 기능레지스터중의 하나로의 라이트액세스 또는 리드액세스에 응답해서 그 하나의 기능 레지스터에 할당된 미리 정해진 처리 수속을 상기 하나의 기능레지스터로 라이트되는 라이트데이타 또는 리드되는 리드데이타에 대해 실행하는 수단을 포함하는 것을 특징으로 하는 데이타처리장치.

청구항 5

연산코드 및 각각에 미리 정해진 처리수속이 할당된 여러개의 기능레지스터에서 하나의 기능레지스터를 특정하는 데이타를 갖는 단일명령에 응답해서 처리수속을 실행하는 데이타처리방법으로서,

[a] 상기 연산코드와는 독립으로 상기 단일명령의 데이타에 의해 특정된 기능레지스터에 할당된 미리 정해져 있는 처리수속을 상기 기능레지스터로 라이트될 라이트데이타 또는 상기 기능레지스터에서 리드될 리드데이타에 대하여 실행하는 공정 및

[b] 상기 단일명령의 연산코드에 의해서 지정된 처리수속을 실행하는 공정을 포함하는 데이타처리방법.

청구항 6

제5항에 있어서, 상기 공정 [a]는

[a1] 상기 기능레지스터에서 리드데이타를 리드하는 공정 및

[a2] 상기 기능레지스터에 할당된 미리 정해져 있는 제1 처리수속을 상기 리드데이타에 대해서 실행하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 7

제5항에 있어서, 상기 공정[b]는

[b1] 상기 연산코드에 의해서 지정된 제2 처리수속을 상기 제1 처리수속이 실행되고 있지 않은 라이트데이타 또는 상기 제1 처리수속이 실행된 리드데이타에 대해서 실행하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 8

제5항에 있어서, 상기 공정[a]는

[a1] 상기 기능레지스터에 할당된 처리수속을 라이트데이타에 대해서 실행하는 공정 및

[a2] 라이트데이타를 상기 기능레지스터로 라이트하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 9

제6항에 있어서, 상기 공정[b]는

상기 기능레지스터에 미리 정해진 처리수속이 실행된 후, 데이타에 대해서 상기 연산코드에 의해서 지정된 처리수속을 실행하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 10

제8항에 있어서, 상기 공정[b]는

상기 연산코드에 의해서 지정된 처리수속을 실행한 후, 데이타를 상기 기능레지스터로 라이트하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 11

명령제어부 및 상기 명령제어부의 제어에 따라 연산코드와 데이타부를 갖는 명령을 실행하는 실행부를 갖는 데이타처리장치로서, 여러개의 범용레지스터 및 각각에 미리 정해진 처리수속이 할당된 여러개의 기능레지스터를 포함하고, 상기 미리 정해진 처리수속은 기능레지스터로 라이트되는 라이트데이타 또는 기능레지스터에서 리드되는 리드데이타에 대해서 실행되고, 명령에 있어서의 연산코드에 의해서 지정된 처리수속과 명령에 있어서의 데이타부에 의해 지정된 기능 레지스터에 할당되어 있는 미리 정해져 있는 처리수속의 양쪽이 기능레지스터를 지정하는 명령의 실행에 응답해서 실행되는 것을 특징으로 하는 데이타처리장치.

청구항 12

제11항에 있어서, 상기 실행부는 기능레지스터의 입력에 접속된 인버터를 포함하고, 상기 기능레지스터에 할당된 미리 정해진 처리수속은 라이트될 라이트데이타를 반전하는 처리수속인 것을 특징으로 하는 데이타처리장치.

청구항 13

제11항에 있어서, 상기 실행부는 가산될 소정의 값을 선택적으로 출력하는 셀렉터 및 기능레지스터의 출력에 상기 소정의 값을 가산하고 그 결과를 상기 기능레지스터로 공급하는 가산기를 포함하고, 상기 기능레지스터에 할당된 미리 정해진 처리수속은 상기 기능레지스터에 저장되어 있는 데이타에 상기 소정의 값을 가산하고 그 결과를 상기 리드데이타로서 출력하는 처리수속인 것을 특징으로 하는 데이타처리장치.

청구항 14

제11항에 있어서, 상기 기능레지스터에 할당된 미리 정해진 처리수속은 명령이 부인 디스플레이스먼트를 나타내는 경우에 그 명령에 따른 처리를 위해 상기 기능레지스터에 저장되어 있는 데이타에서 디스플레이스먼트를 뺀 값을 리드데이타로서 공급하거나, 또는 명령이 0 또는 정인 디스플레이스먼트를 나타내는 경우에 저장된 데이타에 상기 디스플레이스먼트를 가산하는 것에 의해 기능레지스터에 저장되어 있는 데이타를 갱신하는 처리수속인 것을 특징으로 하는 데이타처리장치.

청구항 15

제11항에 있어서, 상기 기능레지스터에 할당된 처리수속은 마스크패턴과 라이트데이타 사이에서 논리곱을 취하고 래치하는 처리수속인 것을 특징으로 하는 데이타처리장치.

청구항 16

제11항에 있어서, 상기 할당된 미리 정해진 처리수속은 프로그램 상태 워드의 올림수 또는 빌림수의 상태를 나타내는 비트의 값을 라이트될 데이타에 가산하고, 그것에 의해 얻어진 데이타를 상기 기능레지스터로 라이트하는 처리수속인 것을 특징으로 하는 데이타처리장치.

청구항 17

제11항에 있어서, 상기 기능레지스터에 할당된 미리 정해진 처리수속은 상기 기능레지스터에 저장된 데이타를 소정방향으로 소정비트수 시프트하는 것을 특징으로 하는 데이타처리장치.

청구항 18

연산코드부 및 상기 연산코드부에 의해서 지정된 처리수속에 있어서 사용될 레지스터를 여러개의 레지스터에서 지정하는 정보부를 갖는 명령에 따라서 데이타를 처리하기 위한 데이타처리방법으로서, 각각 미리 정해진 고유의 동작이 할당된 여러개의 레지스터에서 명령의 정보부에 의해서 지정된 레지스터에 할당되어 있는 고유의 동작으로 그 레지스터에 저장되어 있는 데이타를 제공하는 공정 및 상기 고유의 동작의

결과를 상기 레지스터에서 리드데이타로서 연산코드부에 의해 지정된 처리로 공급하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 19

제18항에 있어서, 상기 공급하는 공정 후에 상기 공급된 리드데이타를 상기 레지스터에 리라이트하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 20

연산코드부 및 상기 연산코드부에 의해서 지정된 처리수속에 있어서 사용될 레지스터를 여러개의 레지스터에서 지정하는 정보부를 갖는 명령에 따라서 데이타를 처리하기 위한 데이타처리방법으로서, 각각 미리 정해진 고유의 동작이 할당된 여러개의 레지스터에서 명령의 정보부에 의해서 지정된 레지스터로 라이트될 데이타를 그 레지스터에 할당되어 있는 고유의 동작으로 제공하는 공정 및 상기 고유의 동작의 결과를 라이트데이타로서 상기 레지스터로 라이트하는 공정을 포함하는 것을 특징으로 하는 데이타처리방법.

청구항 21

제20항에 있어서, 상기 정보부는 여러개의 레지스터에서 고유의 동작이 할당된 레지스터를 선택하는 정보를 포함하고 있는 것을 특징으로 하는 데이타처리방법.

청구항 22

여러개의 처리수속중의 하나를 지정하는 연산코드부 및 상기 연산코드부에 의해서 지정된 처리수속에 있어서 사용될 레지스터를 지정하는 지정정보부를 갖는 명령에 따라서 데이타처리를 실행하는 데이타처리장치로서, 각각 고유의 동작이 할당된 여러개의 기능레지스터에서 상기 지정정보부에 의해 지정된 기능레지스터에 저장되어 있는 데이타의 상기 기능레지스터에 할당되어 있는 고유의 동작으로의 적용을 제어하는 명령제어부 및 할당된 고유의 동작을 저장되어 있는 데이타에 대해서 실행하고 그 결과를 리드데이타로서 상기 연산코드에서 연산코드부에 의해 지정된 처리수속으로 공급하는 실행부를 포함하는 것을 특징으로 하는 데이타처리장치.

청구항 23

제22항에 있어서, 상기 명령제어부는 상기 공급된 리드데이타를 상기 기능레지스터로 라이트하는 것을 특징으로 하는 데이타처리장치.

청구항 24

여러개의 처리수속중의 하나를 지정하는 연산코드 및 상기 연산코드부에 의해서 지정된 처리수속에 있어서 사용될 레지스터를 지정하는 지정정보부를 갖는 명령에 따라서 데이타처리를 실행하는 데이타처리장치로서, 각각 고유의 동작이 할당된 여러개의 기능레지스터에서 상기 지정정보부에 의해 지정된 기능레지스터로 라이트될 라이트데이타의 상기 기능레지스터에 할당되어 있는 고유의 동작으로의 적용을 제어하는 명령제어부 및 할당된 고유의 동작을 상기 라이트데이타에 대해서 실행하고 그 결과를 상기 기능레지스터로 라이트하는 실행부를 포함하는 것을 특징으로 하는 데이타 처리장치.

도면

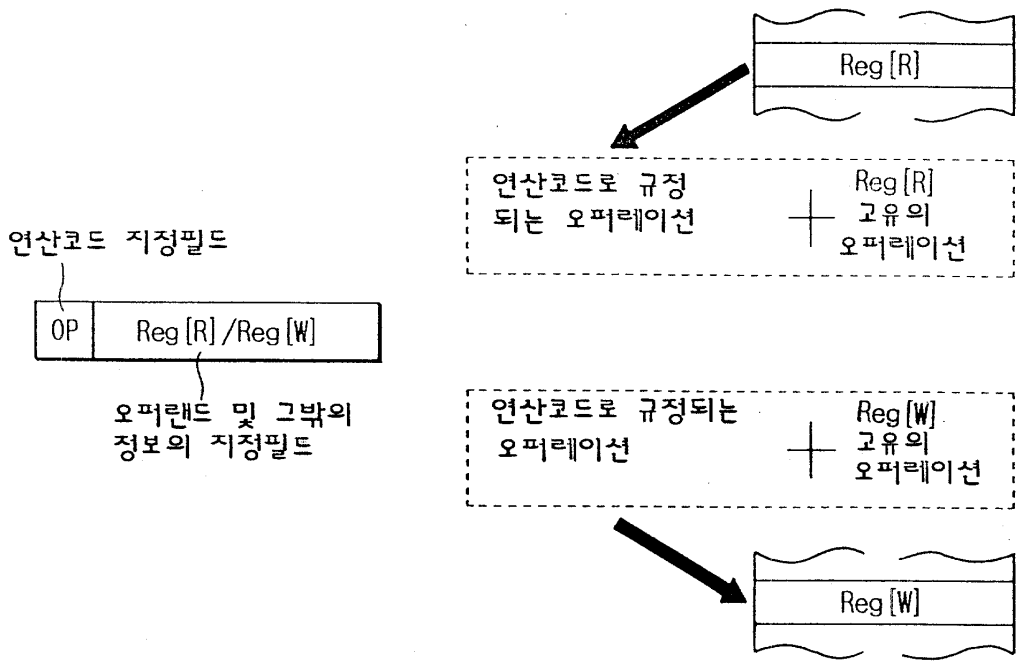
도면1

(1) Operating read data Register
if read data is read from Reg[R]
{read data ←Fr(i) (Reg[R])} andor {Reg[R]←Fr(i) (Reg[R]))};

Fr(i) : i번째의 명령의 특정필드의 값을 파라미터로 하는 레지스터 R 고유의 연산자

(2) Operating write data Register
if write data is written to Reg[W]
Reg[W] ←Fw(i) (Reg[W]);

Fw(i) : i 번째의 명령의 특정필드의 값을 파라미터로 하는 레지스터 W 고유의 연산자



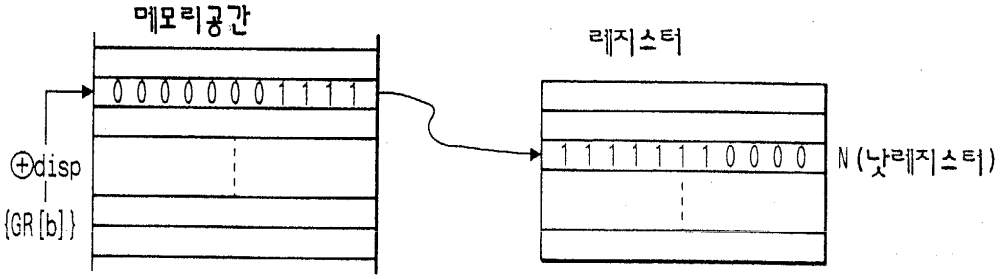
도면2

1. 낫레지스터

정의 if write data is written to Reg[N]
Reg[N]←complement(write data);

응용예

LOAD disp(b), N



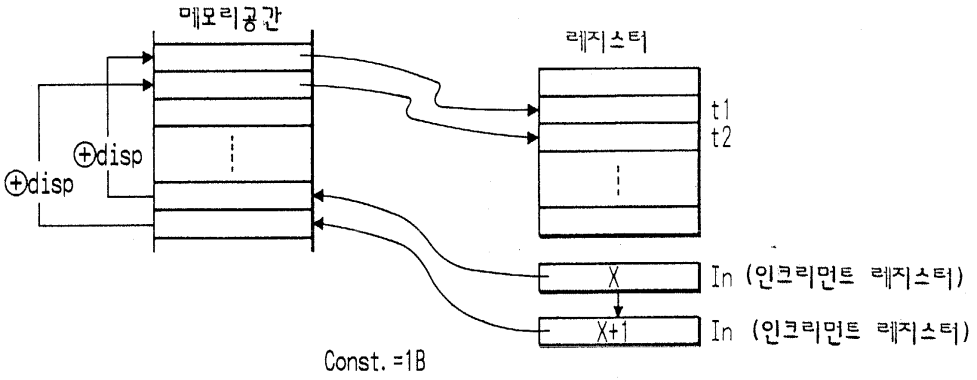
도면3

2. 인크리먼트 레지스터

정의 if read data is read from Reg[In]
Reg[In] ← Reg[In]+Const;

응용예

① LOAD disp(In), t1
② LOAD disp(In), t2



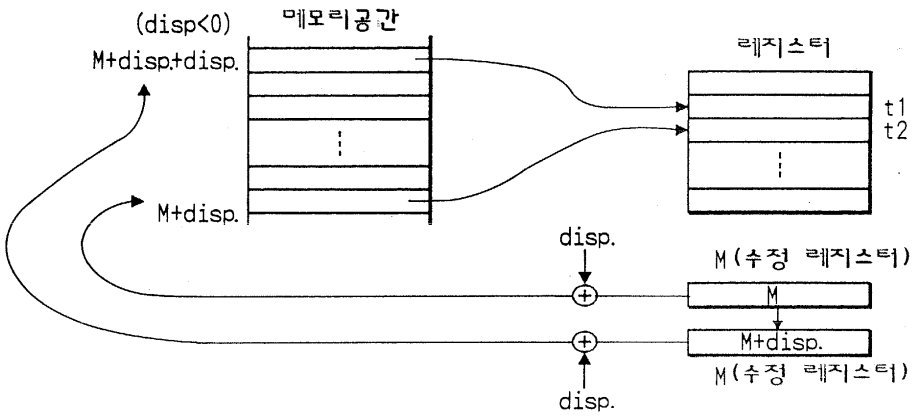
도면4

3. 수정 레지스터

정의
```
if read data is read from Reg[M] {
   if disp.(i)<0
        read data←Reg[M];
   else
        read data←Reg[M]-disp.;
   }
   Reg[M]←Reg[M]+disp.(i);
```

응용예

① LOAD disp(M), t2
② LOAD disp(M), t1



도면5

4. 수정 인덱스레지스터

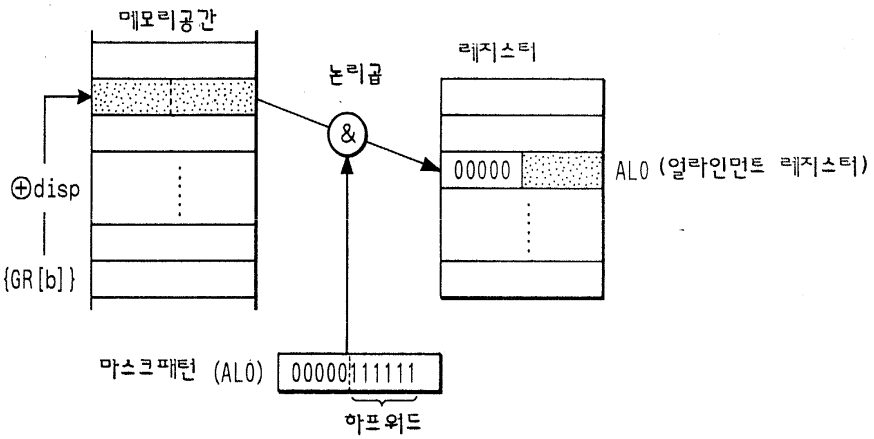
정의 if read data is read from Reg[Mi]
Reg[Mi]←Reg[Mi]+Reg[r](i);

5. 얼라인먼트 레지스터

정의 if write data is written to Reg[AL](i)
Reg[AL]←write data & maskpattern(AL(i));

응용예1

LOAD disp(b), AL0

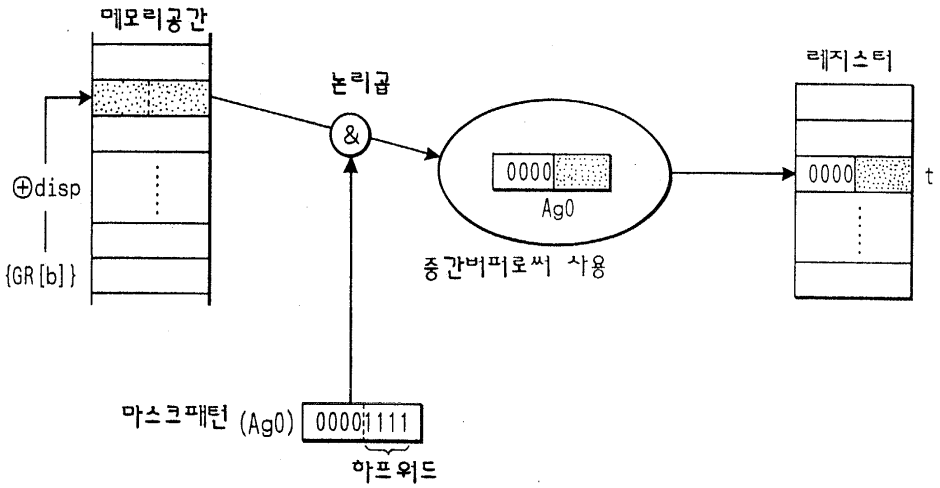


도면6

얼라인먼트 레지스터

응용예2

LOAD disp(b), Ag0, t



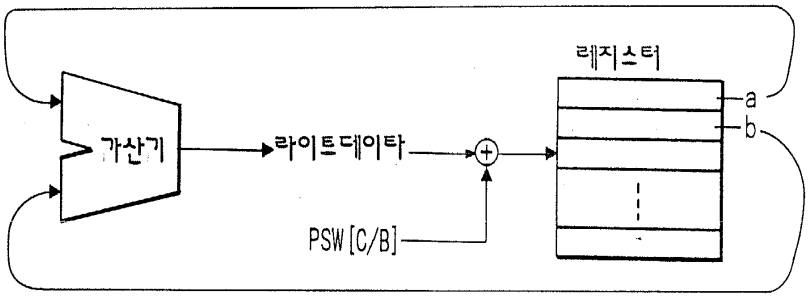
도면7

6. 자리올림/빌림수 레지스터

정의 if write data is written to Reg[C/B]
Reg[C/B]←write data+PSW[C/B];

응용예

ADD a, b, C/B



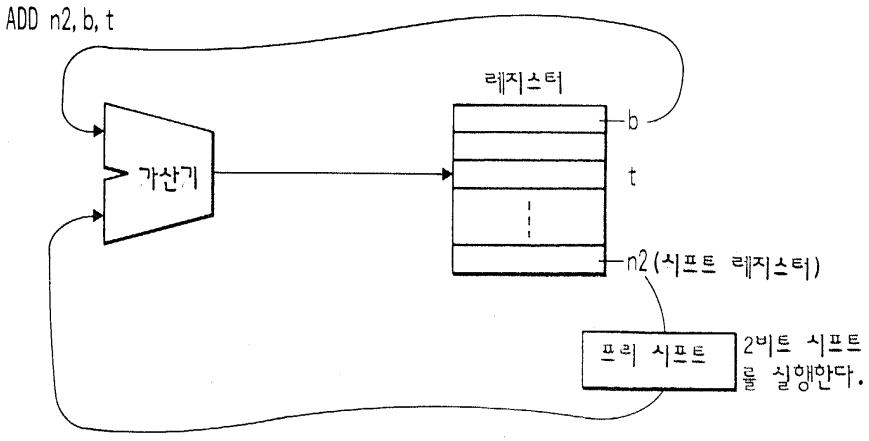
C/B(자리올림/빌림수 레지스터)

도면8

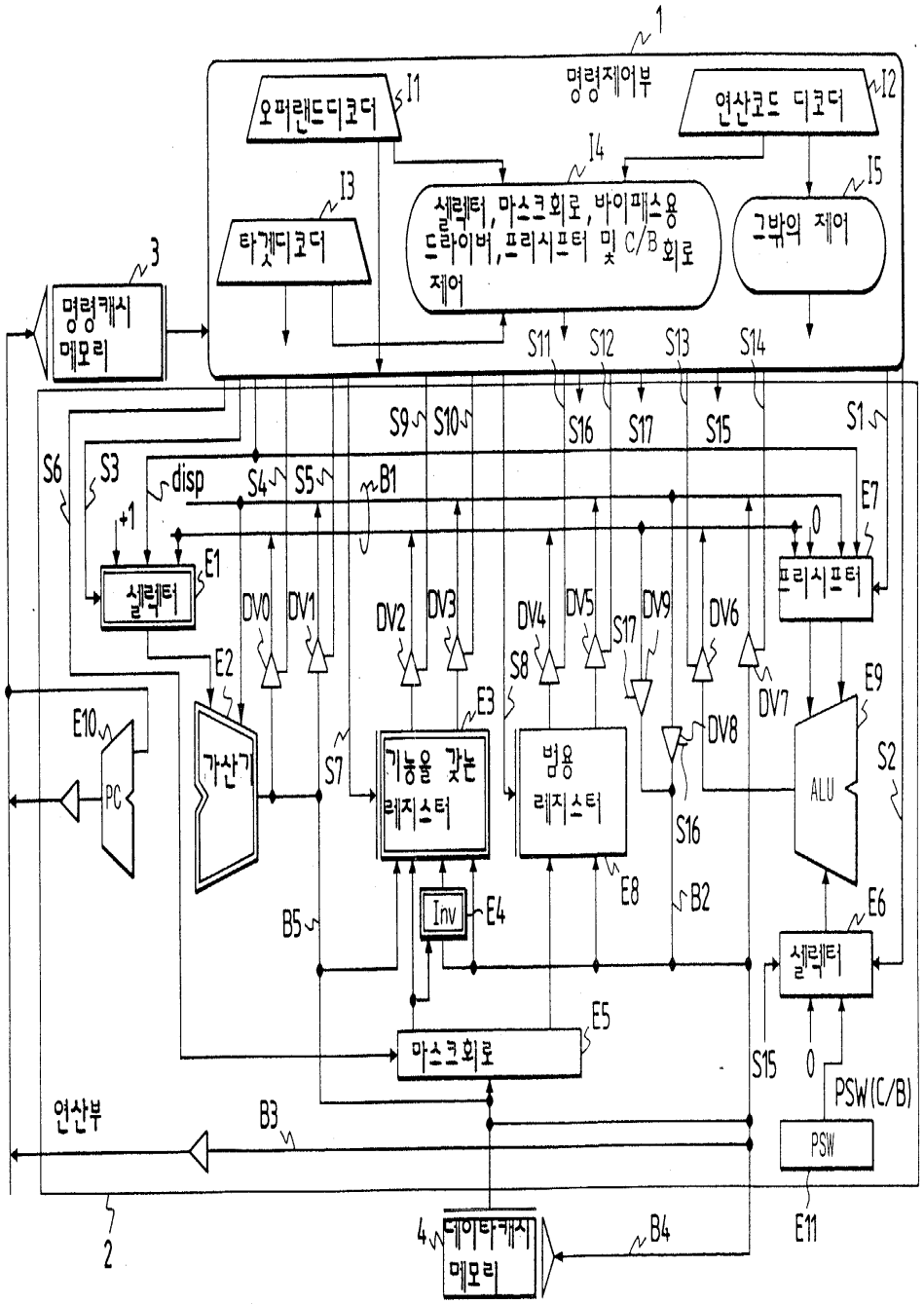
7. 시프트 레지스터

정의 if read data is read from Reg[ni]
read data←shift(Reg[ni], i);

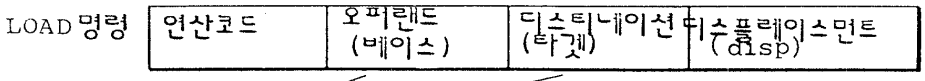
응용예



도면9

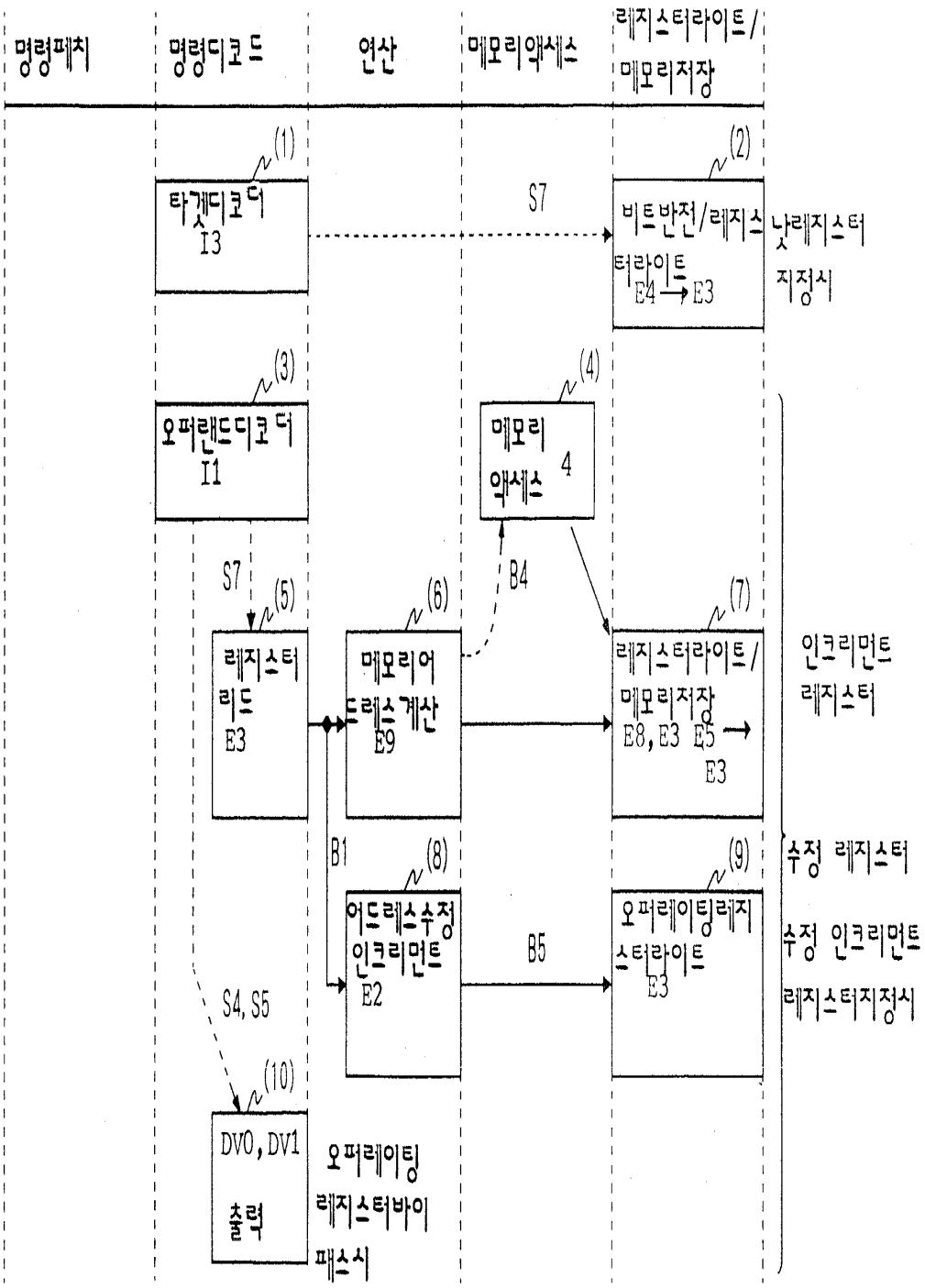

1
명령제어부
오퍼랜드디코더
I1
연산코드 디코더
I2
I3
타겟디코더
I4
셀렉터, 마스크회로, 바이패스용 드라이버, 프리시프터 및 C/B 회로 제어
I5
그밖의 제어
3
명령캐시 메모리
S11
S12
S13
S14
S9
S10
S16
S17
S15
S1
S6
S3
disp
S4
S5
B1
E7
+1
0
E1
셀렉터
프리시프터
DV0
DV1
DV2
DV3
DV4
DV5
S17
DV9
DV6
DV7
E2
S8
E9
E10
E3
DV8
PC
가산기
S7
기능을 갖는 레지스터
범용 레지스터
ALU
S2
S16
E8
B2
B5
Inv
E4
E6
셀렉터
E5
마스크회로
S15
0
PSW(C/B)
연산부
B3
PSW
2
4
데이타캐시 메모리
B4
E11

도면10

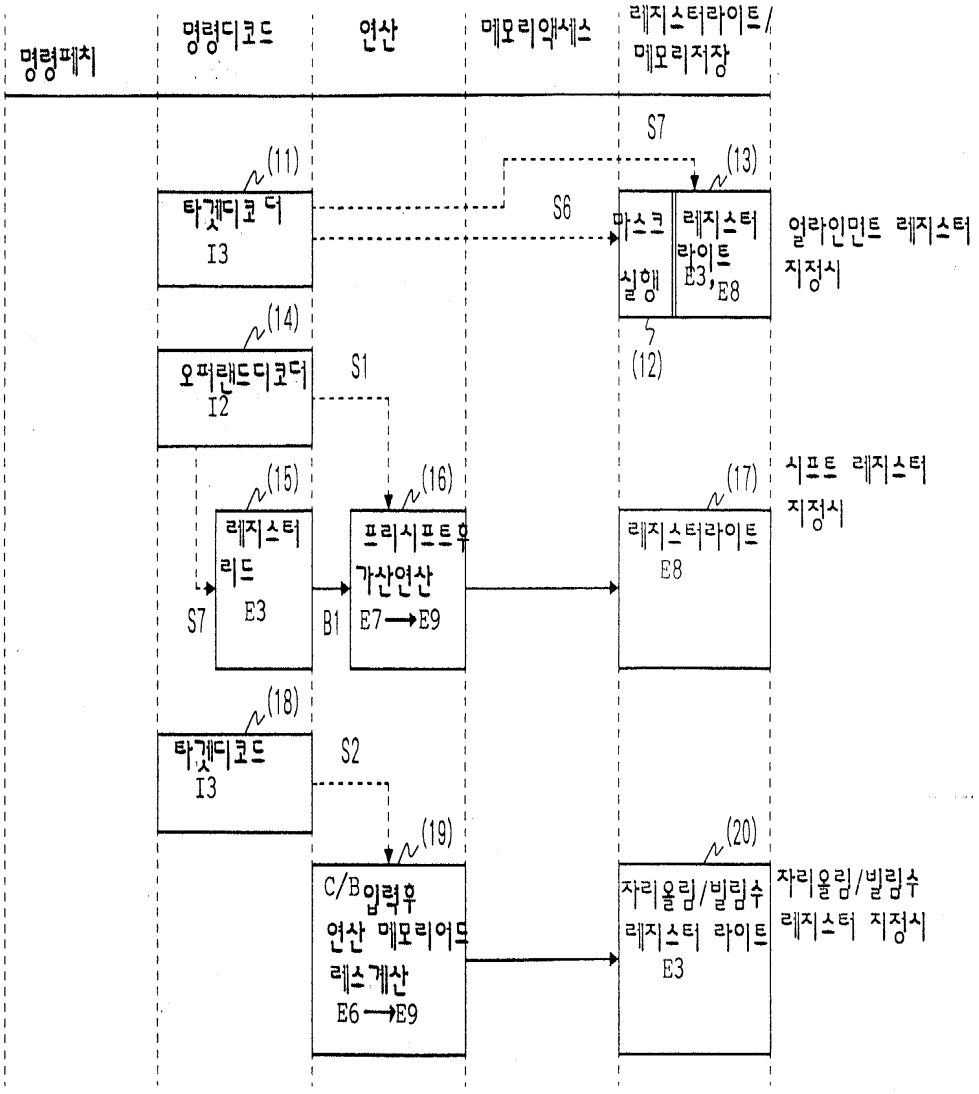


레지스터추정영역 B3 B2 B1 B0	대응 레지스터	
0000	GEReg1	범용레지스터 E8
0001	GEReg2	
0010	GEReg3	
0011	GEReg4	
0100	GEReg5	
0101	GEReg6	
0110	GEReg7	
0111	GEReg8	
1000	OPReg1	기능을 갖는 레지스터 E3
1001	OPReg2	
1010	OPReg3	
1011	OPReg4	
1100	OPReg5	
1101	OPReg6	
1110	OPReg7	
1111	-------	

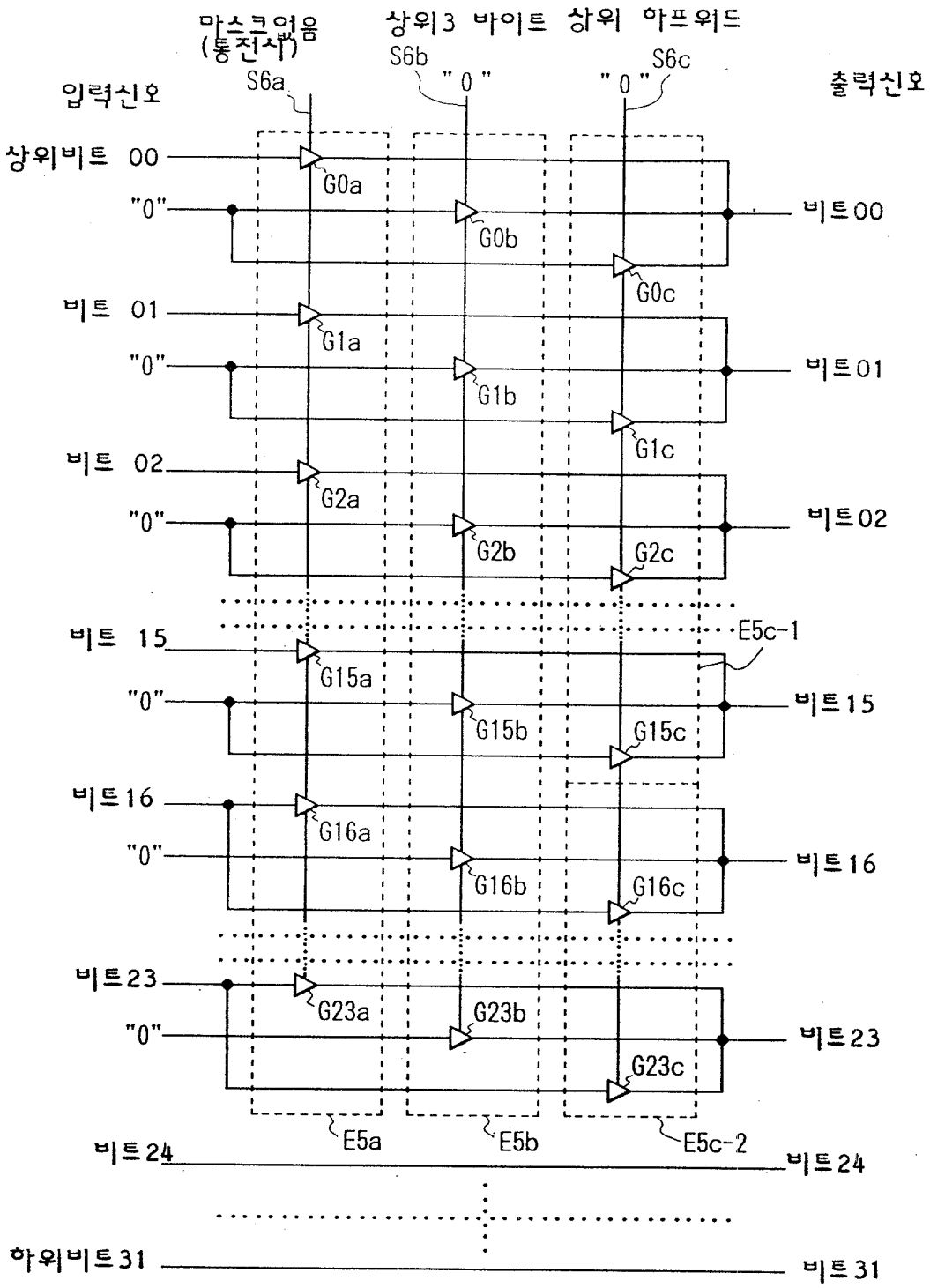
도면11


명령페치
명령디코드
연산
메모리액세스
레지스터라이트/
메모리저장
(1)
타겟디코더
I3
S7
(2)
비트반전/레지스
터라이트
E4→E3
나레지스터
지정시
(3)
오퍼랜드디코더
I1
(4)
메모리
액세스 4
B4
S7
(5)
레지스터
리드
E3
(6)
메모리어
드레스계산
E9
(7)
레지스터라이트/
메모리저장
E8,E3 E5→
E3
인크리먼트
레지스터
B1
(8)
어드레스수정
인크리먼트
E2
(9)
오퍼레이팅레지
스터라이트
E3
B5
수정 레지스터
수정 인크리먼트
레지스터지정시
S4,S5
(10)
DV0,DV1
출력
오퍼레이팅
레지스터바이
패스시

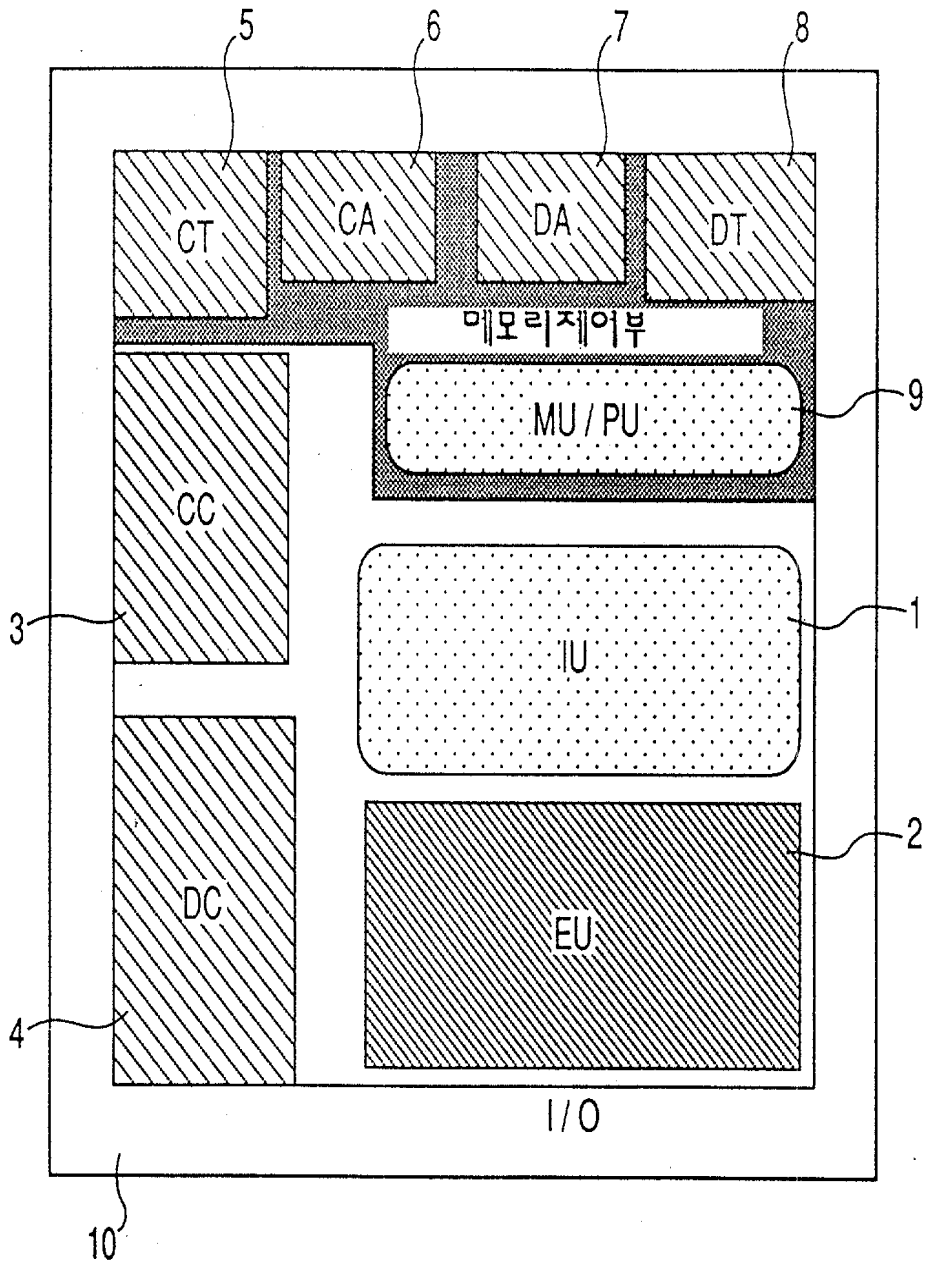
도면12


명령페치
명령디코드
연산
메모리액세스
레지스터라이트/
메모리저장
S7
(11)
타겟디코더
I3
S6
(13)
마스크
실행
레지스터
라이트
E3, E8
얼라인먼트 레지스터
지정시
(12)
(14)
오퍼랜드디코더
I2
S1
(15)
레지스터
리드
E3
S7
(16)
프리시프트후
가산연산
E7→E9
B1
(17)
레지스터라이트
E8
시프트 레지스터
지정시
(18)
타겟디코드
I3
S2
(19)
C/B입력후
연산 메모리어드
레스계산
E6→E9
(20)
자리올림/빌림수
레지스터 라이트
E3
자리올림/빌림수
레지스터 지정시

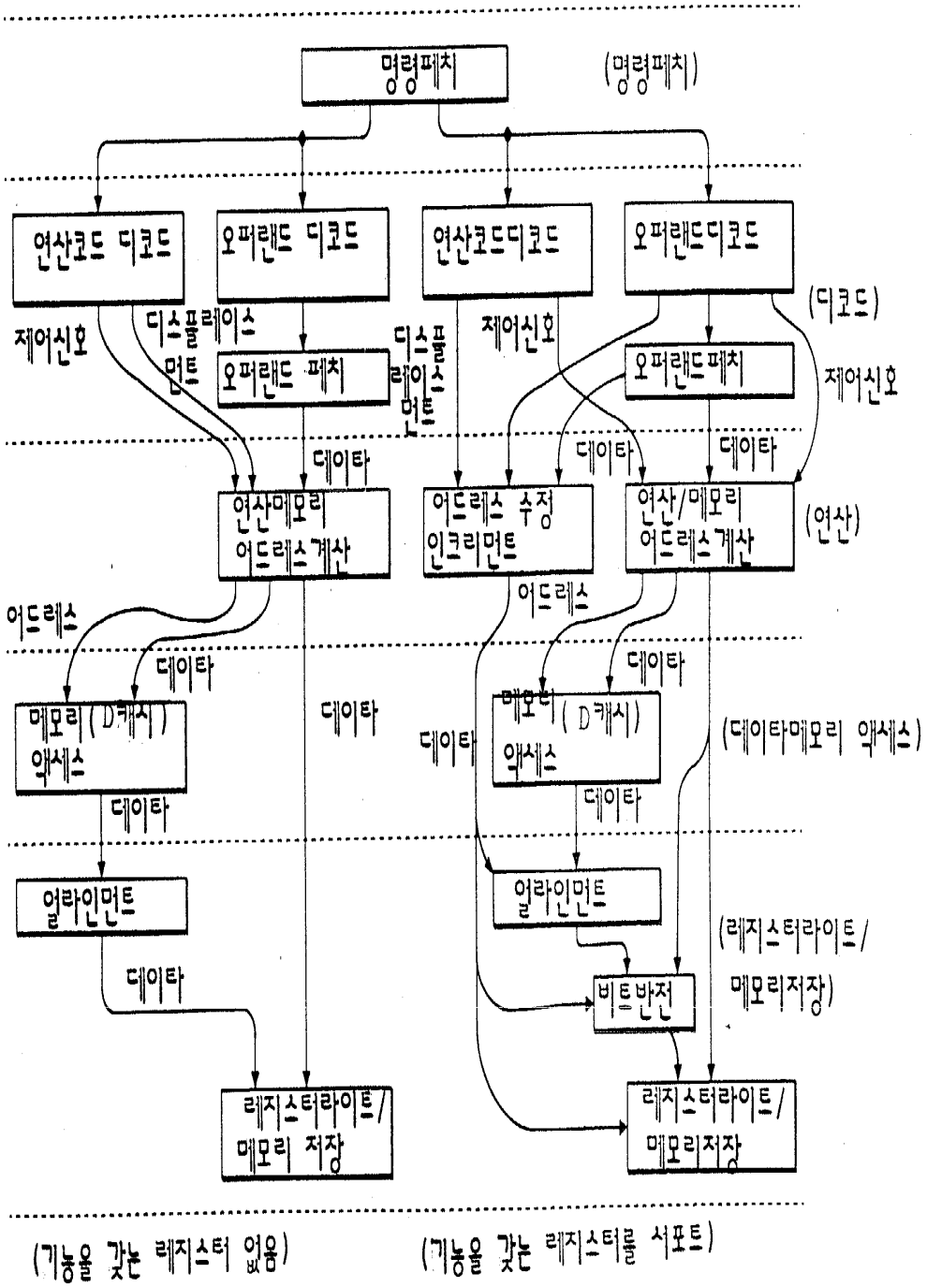
도면13


마스크없음
(통전시)
상위3 바이트 상위 하프워드
입력신호
출력신호
S6a
S6b
S6c
"0"
"0"
상위비트 00
"0"
비트 01
"0"
비트 02
"0"
비트 15
"0"
비트16
"0"
비트23
"0"
비트24
하위비트31
G0a
G0b
G0c
G1a
G1b
G1c
G2a
G2b
G2c
G15a
G15b
G15c
G16a
G16b
G16c
G23a
G23b
G23c
E5a
E5b
E5c-1
E5c-2
비트00
비트01
비트02
비트15
비트16
비트23
비트24
비트31

도면14



도면15


명령페치
(명령페치)
연산코드 디코드
오퍼랜드 디코드
연산코드디코드
오퍼랜드디코드
(디코드)
제어신호
디스플레이스먼트
오퍼랜드 페치
디스플레이스먼트
제어신호
오퍼랜드페치
제어신호
데이터
데이터
데이터
연산메모리 어드레스계산
어드레스 수정 인크리먼트
연산/메모리 어드레스계산
(연산)
어드레스
어드레스
데이터
데이터
메모리(D캐시) 액세스
데이터
데이터
메모리(D캐시) 액세스
(데이터메모리 액세스)
데이터
얼라인먼트
얼라인먼트
(레지스터라이트/
메모리저장)
데이터
비트반전
레지스터라이트/ 메모리 저장
레지스터라이트/ 메모리저장
(기능을 갖는 레지스터 없음)
(기능을 갖는 레지스터를 서포트)

도면16

	기능을 갖는 레지스터 없음	기능을 갖는 레지스터를 서포트
연산코드영역의 비트수	X	O
오퍼랜드영역의 비트수	O	△
타겟영역의 비트수	O	△
연산코드디코더의 논리규모	X	O
오퍼랜드디코더의 논리규모	O	△
타겟디코더의 논리규모	O	△

O 증가없음
△ 증가 작음
X 증가 큼