(19)대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl.
G06F 13/16 (2006.01)
G06F 12/00 (2006.01)

(11) 공개번호 10-2006-0129888
(43) 공개일자 2006년12월18일

(21) 출원번호 10-2005-0050604
(22) 출원일자 2005년06월13일
심사청구일자 2005년06월13일

(71) 출원인 재단법인서울대학교산학협력재단
서울특별시 관악구 봉천동 산 4-2

(72) 발명자 성원용
서울 강남구 대치2동 미도아파트 103동 202호
장호석
서울 관악구 봉천6동 우성아파트 104동 2103호

(74) 대리인 이헌수
유경열

전체 청구항 수 : 총 10 항

(54) 벡터 메모리, 이를 구비한 프로세서 및 그 데이터 처리방법

(57) 요약

하나의 명령어로 복수개의 데이터를 처리하는 단일 명령어 복수 데이터 처리(Single Instruction Multiple Data, SIMD) 구조 하에서의 벡터 메모리와, 이를 구비한 프로세서 및 그 데이터 처리 방법이 개시된다. 본 발명에 따라, 벡터 메모리는 데이터가 저장된 위치를 지시하는 주소를 생성하는 복수개의 주소 생성부; 상기 주소에 따라 데이터를 저장하는 복수개의 메모리 뱅크; 상기 메모리 뱅크에서 출력되는 데이터를 스위칭을 통해 재정렬하여 출력하는 스위치 매트릭스; 및 상기 주소 생성부의 주소 생성과 상기 스위치 매트릭스의 스위칭을 제어하는 제어부를 포함하는 것을 특징으로 한다. 이에 의해, 데이터의 병렬 처리 효율을 높일 수 있다.

대표도

도 2

특허청구의 범위

청구항 1.

데이터가 저장된 위치를 지시하는 주소를 생성하는 복수개의 주소 생성부;

상기 주소에 따라 데이터를 저장하는 복수개의 메모리 뱅크;

상기 메모리 뱅크에서 출력되는 데이터를 스위칭을 통해 재정렬하여 출력하는 스위치 매트릭스; 및

상기 주소 생성부의 주소 생성과 상기 스위치 매트릭스의 스위칭을 제어하는 제어부를 포함하는 것을 특징으로 하는 벡터 메모리.

청구항 2.

제1항에 있어서, 상기 메모리 뱅크는

상기 저장된 복수개의 데이터 들 중에서, 상기 제어부의 제어 신호와 주소 생성부의 주소에 따라 각각의 메모리 뱅크별로 하나의 데이터를 출력하거나 하나도 출력하지 않는 것을 특징으로 하는 벡터 메모리.

청구항 3.

제1항에 있어서, 상기 주소 생성부는

상기 메모리 뱅크에서 특정 메모리 접근 패턴의 경우에는, 뱅크 충돌을 피하기 위해 한 뱅크에 대해 함께 접근하게 되는 단위 데이터를 다른 뱅크에 배치되도록 주소를 생성하는 것을 특징으로 하는 벡터 메모리.

청구항 4.

제1항에 있어서,

상기 메모리 뱅크의 개수는, 상기 스위치 매트릭스에 의해서 병렬적으로 출력되는 데이터 개수와 동일하거나 많은 것을 특징으로 하는 벡터 메모리.

청구항 5.

제1항에 있어서, 상기 스위치 매트릭스는

상기 제어부의 신호에 따라, 상기 복수개의 메모리 뱅크 각각에서 출력되는 데이터를 크로스 매트릭스 스위치의 스위칭을 통해 재정렬하여 출력하는 것을 특징으로 하는 벡터 메모리.

청구항 6.

주 메모리;

병렬 처리될 데이터를 저장하는 벡터 메모리;

상기 주 메모리 또는 벡터 메모리에서 데이터를 읽어 데이터 폭(width)을 확장하여 출력하는 레지스터부; 및

상기 레지스터부에서 데이터를 읽어 연산을 수행하는 산술논리 연상장치를 포함하는 것을 특징으로 하는 프로세서.

청구항 7.

제6항에 있어서, 상기 벡터 메모리는

데이터가 저장된 위치를 지시하는 주소를 생성하는 복수개의 주소 생성부;

상기 주소에 따라 데이터를 저장하는 복수개의 메모리 뱅크;

상기 메모리 뱅크에서 출력되는 데이터를 스위칭을 통해 재정렬하여 출력하는 스위치 매트릭스; 및

상기 주소 생성부의 주소 생성과 상기 스위치 매트릭스의 스위칭을 제어하는제어부를 포함하는 것을 특징으로 하는 프로세서.

청구항 8.

제6항 또는 제7항에 있어서,

상기 레지스터부는, 상기 벡터 메모리에 포함된 메모리 뱅크 각각에서 출력된 데이터를 합하여 벡터 데이터로 구성하여 상기 산술논리 연산장치로 출력하거나, 상기 산술논리 연산장치에서 출력된 벡터 데이터를 저장하는 것을 특징으로 하는 프로세서.

청구항 9.

접근하고자 하는 벡터 데이터를 구성하는 각 단위 데이터를 저장하고 있는 각 메모리 뱅크에서의 주소를 계산하여 출력하는 단계;

상기 계산된 주소에 따라 각 메모리 뱅크를 접근하여 데이터를 출력하는 단계; 및

상기 선택된 데이터를 크로스 매트릭스 스위치의 스위칭을 통해 출력 위치를 재정렬하여 출력하는 단계를 포함하는 것을 특징으로 하는 데이터 처리 방법.

청구항 10.

제9항에 기재된 방법을 컴퓨터에서 실행시키기 위한 프로그램을 기록한 컴퓨터로 읽을 수 있는 기록매체.

명세서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 디지털 신호 처리에 관한 것으로, 보다 상세하게는 하나의 명령어로 복수개의 데이터를 처리하는 단일 명령어 복수 데이터 처리(Single Instruction Multiple Data, SIMD) 구조 하에서의 벡터 메모리와, 이를 구비한 프로세서 및 그 데이터 처리 방법에 관한 것이다.

단일 명령어 복수 데이터(SIMD) 구조의 프로세서는 하나의 명령어(instruction)로 복수개의 데이터를 한 번에 처리할 수 있는 산술논리 연산장치(ALU)를 구비하고 있다. 산술논리 연산장치(ALU)에서 데이터 처리를 수행하기 위하여 사용되는 오퍼랜드(operand)는 SIMD 레지스터에 저장되어 있으며, 산술논리 연산장치(ALU)는 SIMD 레지스터에서 데이터를 읽어온다. 이때, 산술논리 연산장치(ALU)에서 데이터를 입출력하여 처리하는 단위는 64 비트 또는 128 비트 등이며, 그 내부에서는 데이터를 8 비트, 16 비트 또는 32 비트 등의 단위로 처리하므로, 한 번의 데이터 입출력으로 동시에 여러 개의 단위 데이터를 처리할 수 있다.

그러나, 종래의 SIMD 구조의 프로세서는 신호 처리를 위한 애플리케이션 프로그램을 수행하는데 있어서, 메모리에 저장된 데이터를 읽어 오거나 저장하는 과정에서 병목현상이 발생한다. 보다 구체적으로, 신호 처리 애플리케이션에서 처리하는 데이터는 어레이(array)의 형태로 입력되는 경우가 많으며, 메모리에 저장되어 처리되어야 할 데이터 저장 형태와 SIMD 구조의 프로세서에서 처리되는 데이터의 형태가 상이하기 때문에 단순히 메모리를 통해 데이터를 읽어와 처리하는 경우에는 병렬 처리 성능이 떨어진다는 문제점이 있다.

발명이 이루고자 하는 기술적 과제

따라서, 본 발명이 이루고자 하는 기술적 과제는 복수개의 데이터를 하나의 명령어로 병렬적으로 가져올 수 있는 다중 출력 포트를 가지는 벡터 메모리, 이를 구비한 프로세서 및 그 데이터 처리 방법을 제공하는 것이다.

발명의 구성

상기 기술적 과제는 본 발명에 따라, 데이터가 저장된 위치를 지시하는 주소를 생성하는 복수개의 주소 생성부; 상기 주소에 따라 데이터를 저장하는 복수개의 메모리 뱅크; 상기 메모리 뱅크에서 출력되는 데이터를 스위칭을 통해 재정렬하여 출력하는 스위치 매트릭스; 및 상기 주소 생성부의 주소 생성과 상기 스위치 매트릭스의 스위칭을 제어하는 제어부를 포함하는 것을 특징으로 하는 벡터 메모리에 의해 달성된다.

상기 메모리 뱅크는, 상기 저장된 복수개의 데이터 들 중에서, 상기 제어부의 제어 신호와 주소 생성부의 주소에 따라 각각의 메모리 뱅크별로 하나의 데이터를 출력하거나 하나도 출력하지 않는 것이 바람직하다.

상기 메모리 뱅크의 개수는, 상기 스위치 매트릭스에 의해서 병렬적으로 출력되는 데이터 개수와 동일하거나 많은 것이 바람직하다.

또한, 상기 기술적 과제는, 주 메모리; 병렬 처리될 데이터를 저장하는 벡터 메모리; 상기 주 메모리 또는 벡터 메모리에서 데이터를 읽어 데이터 폭(width)을 확장하여 출력하는 레지스터부; 및 상기 레지스터부에서 데이터를 읽어 연산을 수행하는 산술논리 연상장치를 포함하는 것을 특징으로 하는 프로세서에 의해서도 달성된다.

상기 레지스터부는, 상기 벡터 메모리에 포함된 메모리 뱅크 각각에서 출력된 데이터를 합하여 벡터 데이터로 구성하여 상기 산술논리 연산장치로 출력하거나, 상기 산술논리 연산장치에서 출력된 벡터 데이터를 저장하는 것이 바람직하다.

한편, 본 발명의 다른 분야에 따르면, 상기 기술적 과제는 접근하고자 하는 벡터 데이터를 구성하는 각 단위 데이터를 저장하고 있는 각 메모리 뱅크에서의 주소를 계산하여 출력하는 단계; 상기 계산된 주소에 따라 각 메모리 뱅크를 접근하여 데이터를 출력하는 단계; 및 상기 선택된 데이터를 크로스 매트릭스 스위치의 스위칭을 통해 출력 위치를 재정렬하여 출력하는 단계를 포함하는 것을 특징으로 하는 데이터 처리 방법에 의해서도 달성된다.

이하 첨부된 도면을 참조하여 본 발명의 바람직한 실시 예에 대해 상세히 설명한다.

도 1은 본 발명의 바람직한 일실시예에 따른 벡터 메모리를 구비한 프로세서의 블록도이다.

도 1을 참조하면, 프로세서는 주 메모리(110), 벡터 메모리(120), 레지스터부(130) 및 산술논리 연산장치(ALU)(140)를 포함함을 알 수 있다. 주 메모리(110)와 벡터 메모리(120)는 32 비트의 데이터 버스에 연결되어 데이터를 입출력한다. 한편 벡터 메모리(120)는, 산술논리 연산장치(140)에서 처리할 데이터를 저장하고 있고, 이 데이터는 128 비트 단위로 레지스터부(130)를 경유하여 산술논리 연산장치(140)로 전달된다. 산술논리 연산장치(140)는 레지스터부(130)에서 오퍼랜드(operand)와 데이터를 입력받아 연산을 수행한다. 벡터 메모리를 사용하는 명령어(instruction)의 일예는 다음 표 1에 도시한 바와 같다.

[표 1]

operation	Assembly mnemonic
vector load to register with stride	vld <addr>, <stride>, Rdst, <shuffle_type>
vector load from arbitrary locations	vld <addr>, Roffset, Rdst

vector store from register with stride	vst <addr>, <stride>, Rsrc, <shuffle_type>
vector store to arbitrary locations	vst <addr>, Roffset, Rdst

이러한 오퍼랜드는 벡터 메모리(120)에 저장되어 있어, 레지스터(130)를 통해 산술논리 연산장치(140)로 입력된다. 본 실시예에서는 벡터 메모리(120)와 레지스터부(130) 및 산술논리 연산장치(140)간의 데이터 전달은 128 비트로 수행되는 경우를 설명하였다. 산술논리 연산장치(140)에서 수행된 결과는 다시 레지스터부(130)로 전달되어, 32 비트 데이터 버스를 통해 시스템으로 전달된다.

도 2는 도 1에 도시한 벡터 메모리의 상세 블록도이다.

벡터 메모리(120)는 제어부(210), 복수개의 주소 생성부(220-1 내지 220-5), 메모리 뱅크(230-1 내지 230-5), 스위치 매트릭스(240)를 포함한다. 본 실시예에서 주소 생성부와 메모리 뱅크는 각각 5개가 존재하고 출력되는 데이터는 64 비트인 경우에 대하여 설명한다. 하나의 메모리 뱅크에서는 한 번에 하나의 데이터만 접근 할 수 있기 때문에, 필요로 하는 복수개의 데이터가 하나의 메모리 뱅크에 저장된 경우 데이터를 액세스 시 충돌이 발생한다. 이를 방지하기 위해서 메모리 뱅크의 개수를 조절하거나 주소 생성부의 구성을 변경한다. 주소 생성부는 해당 메모리 뱅크에, 필요로 하는 데이터가 저장되어 있는지 판단하는 뱅크 판단식과, 해당 메모리 뱅크내에서 어느 위치에 필요로 하는 데이터가 저장되어 있는지를 판단하는 주소 생성식에 따라 주소를 생성한다. 이 때, 메모리 뱅크의 개수를 조절하는 경우, 병렬적으로 출력되는 데이터의 개수보다 많도록 설계하여 충돌을 피한다. 예를 들어 도 2의 실시예에서, 출력되는 데이터가 64 비트로 16 비트 데이터 4개가 병렬적으로 출력되므로 메모리 뱅크의 수는 이보다 하나 많은 5개가 된다. 메모리 뱅크의 액세스시에 충돌을 피하기 위해 주소 생성부에서 한꺼번에 접근되는 데이터를 서로 다른 메모리 뱅크에 저장하여 충돌을 피한다.

메모리 뱅크의 수가 늘어나면 한 번에 가져올 수 있는 단위 데이터의 수가 많아져 병렬도(동시에 읽어올 수 있는 데이터의 개수)를 증가시킬 수 있고, 그에 따라 스위치 매트릭스(240)와 제어부(210)의 구성이 복잡해진다. 또한 멀티 뱅크 메모리 접근시 메모리 뱅크의 충돌을 감소시키기 위해 메모리 뱅크의 개수를 조절할 수 있는데 예를 들어 메모리 뱅크의 수를 (병렬도 + n)으로 정할 수 있다. 이때 n은 0보다 큰 정수이다.

각각의 메모리 뱅크(230-1 내지 230-5)는 8 비트, 16 비트, 32 비트 등의 단위로 데이터를 저장한다. 주소 생성부(220-1 내지 220-5)는 각각의 메모리 뱅크(230-1 내지 230-5)의 어느 위치에 데이터가 저장되어 있는가 하는 주소를 생성한다. 주소 생성부에서 만든 주소에 따라 메모리 뱅크의 데이터가 출력된다. 즉, 주소 생성부(220-1 내지 220-5)는, 접근하고자 하는 벡터 데이터가 어느 메모리 뱅크에 저장되어 있는지를 판단하고 메모리 뱅크에서의 위치를 계산하여 출력한다. 하나의 메모리 뱅크는 한 번에 하나의 데이터만 접근할 수 있기 때문에 벡터 데이터의 접근 패턴에 따라 메모리 뱅크에서 두 개 이상의 데이터로 접근하려는 충돌이 발생한다. 발생하는 메모리 뱅크 충돌을 피하기 위해 주소 생성부를 이용할 수 있다. 이 때 여러 가지 메모리 접근 패턴에 대해 메모리 뱅크 충돌을 피하기 위해서는 접근 패턴에 따라 주소 생성부가 각각 다른 모드로 동작하게 할 수 있다.

스위치 매트릭스(240)는 각각의 메모리 뱅크(230-1 내지 230-5)에서 출력되는 데이터를 스위칭하여 레지스터부(130)로 출력한다. 스위치 매트릭스(240)는 크로스 매트릭스 스위치(cross matrix switch)로써, 크로스 포인트의 스위칭을 통해, 데이터의 출력 경로를 결정한다. 이때, 레지스터부(130)에 연결된 어느 버스로, 어떤 메모리 뱅크에서 출력된 데이터를 전달한 것인가 하는 스위칭 정보는 제어부(210)에서 전달한다. 즉, 제어부(210)는 스위치 매트릭스(240)에서 출력된 데이터를 어느 버스를 통해 출력할 것인가를 결정하여 스위치 매트릭스(240)로 제어신호를 보낸다. 따라서 스위치 매트릭스(240)는 제어부(210)의 재정렬(reordering) 신호에 따라 데이터를 재정렬하여 레지스터부(130)로 전달한다.

도 3a는 뱅크 충돌을 피하기 위해 메모리 뱅크의 수를 늘린 경우, 메모리 뱅크에서 출력된 데이터를 스위칭하여, 레지스터부로 출력될 데이터를 만드는 것을 설명하기 위한 참조도이다.

즉, 도 3a는 메모리 뱅크의 수를 (병렬도 + 1) 로 설정하여 뱅크 충돌을 피하는 경우로, 각 메모리 뱅크(230-1 내지 230-5)의 주소 생성부(220-1 내지 220-5)는 <뱅크 안에서의 주소> = <접근 하려는 주소> / 5, <뱅크번호> = <접근하려는 주소> mod 5 의 수식을 이용하여 각각의 메모리 뱅크를 접근한다.

도 3a를 참조하면, 첫 번째 메모리 뱅크(230-1)에서 A[0] 데이터가 출력되고 이는 스위치 매트릭스(240)에서 Pbus0 (partitioned bus 0) 버스로 연결되어 출력된다. 그리고 세 번째 메모리 뱅크(230-3)에서 A[12] 데이터가 출력되고 이는 스위치 매트릭스(240)에서 Pbus3 버스로 연결되어 출력되고, 네 번째 메모리 뱅크(230-4)에서 A[8] 데이터가 출력되고

이는 스위치 매트릭스(240)에서 Pbus2 버스로 연결되어 출력된다. 또한, 다섯 번째 메모리 뱅크(230-5)에서 A[4] 데이터가 출력되고 이는 스위치 매트릭스(240)에서 Pbus1 버스로 연결되어 출력된다. 각각의 메모리 뱅크(230-1 내지 230-5)에서 어떤 어드레스의 데이터가 출력되는가 하는 것은 주소 생성부(220-1 내지 220-5)에 의해 결정되고, 스위치 매트릭스(240)의 스위칭은 제어부(210)에 제어 신호에 따라 수행된다.

도 3b는 메모리 뱅크의 수가 병렬도와 같은 경우로, 메모리 뱅크에서 출력된 데이터를 스위칭하여, 레지스터부로 출력될 데이터를 만드는 것을 설명하기 위한 참조도이다.

도 3b의 실시예에서는, 한 번에 접근하고자 하는 데이터의 주소가 0, 4, 8, 12 라 할 때 메모리 뱅크 충돌을 방지하기 위하여, 주소 생성부에서는 <뱅크 안에서의 주소> = <접근 하려는 주소> / 4, <뱅크번호> = (<접근하려는 주소> + <뱅크 안에서의 주소>) mod 4 수식을 이용하여 접근하는 메모리 뱅크 번호와 각 메모리 뱅크에서의 주소가 계산된다. 이때 모든 접근 패턴에 대해 메모리 뱅크 충돌을 피할 수는 없지만, 산술논리 연산장치에서 수행되는 소프트웨어를 분석하여 주된 접근 패턴을 찾아내 그 패턴의 경우에 메모리 뱅크 충돌을 피할 수 있도록 주소 생성부를 구성한다. 또한 주소 생성부에 모드를 두어서 접근하는 데이터의 종류에 따라 모드를 달리하여 각각 다른 뱅크 판단식과 주소 생성식을 이용하면 뱅크 충돌을 피할 수 있는 메모리 접근 패턴이 늘어난다.

도 4는 벡터 메모리에서 SIMD 처리를 위한 데이터를 생성하는 과정의 흐름도이다.

우선, 복수개의 메모리 뱅크 각각에서 주소 생성부의 신호에 따라 데이터를 출력한다(S410). 그리고 스위칭 매트릭스는, 제어부의 신호에 따라 상기 출력된 데이터 중에서 복수개를 선택한다(S420). 이렇게 선택된 데이터를 재정렬하여 출력한다(S430).

한편, 표 2는 본 발명에 따른 벡터 메모리를 사용한 경우에 벤치마크 테스트 결과를 도시한 테이블이다. 표 2를 참조하면, 벡터 메모리(VM)를 사용하는 경우 명령어(instruction)의 수가 줄어들고, 수행되는 연산의 종류에 따라 최대 345%의 성능 향상이 있음을 알 수 있다.

[표 2]

Benchmarks		no. of dynamic instructions		속도 향상률(%)
		SIMD	VM 사용시	
C compiler	Matrix multiply	784	176	345
	2D DCT	4839	1279	278
	FIR	105	42	250
IPP (Intel Performance Primitive)	IIR	65	49	32
	FFT	171	150	14
	2D DCT	164	159	3

한편, 전술한 데이터 처리 방법은 컴퓨터 프로그램으로 작성 가능하다. 상기 프로그램을 구성하는 코드들 및 코드 세그먼트들은 당해 분야의 컴퓨터 프로그래머에 의하여 용이하게 추론될 수 있다. 또한, 상기 프로그램은 컴퓨터가 읽을 수 있는 정보저장매체(computer readable media)에 저장되고, 컴퓨터에 의하여 읽혀지고 실행됨으로써 데이터 처리 방법을 구현한다. 상기 정보저장매체는 자기 기록매체, 광 기록매체, 및 캐리어 웨이브 매체를 포함한다.

이제까지 본 발명에 대하여 그 바람직한 실시예들을 중심으로 살펴보았다. 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명이 본 발명의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현될 수 있음을 이해할 수 있을 것이다. 그러므로 개시된 실시예들은 한정적인 관점이 아니라 설명적인 관점에서 고려되어야 한다. 본 발명의 범위는 전술한 설명이 아니라 특허청구범위에 나타나 있으며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명에 포함된 것으로 해석되어야 할 것이다.

발명의 효과

전술한 바와 같이 본 발명에 따르면, 종래의 SIMD 구조의 프로세서 발생하는 메모리 입출력의 병목현상을 방지할 수 있으며, 데이터의 병렬 처리 효율을 높일 수 있고, 벡터 메모리와 이를 구비한 프로세서를 간단하게 구현할 수 있다는 효과가 있다. 또한 벡터 메모리를 멀티 뱅크 구조를 이용해 구현하고, SIMD 구조의 병렬도 확장에 따라 메모리 뱅크를 추가함으로써 효과적으로 데이터의 처리 성능이 향상된다는 효과가 있다.

도면의 간단한 설명

도 1은 본 발명의 바람직한 일실시예에 따른 벡터 메모리를 구비한 프로세서의 블록도,

도 2는 도 1에 도시한 벡터 메모리의 상세 블록도,

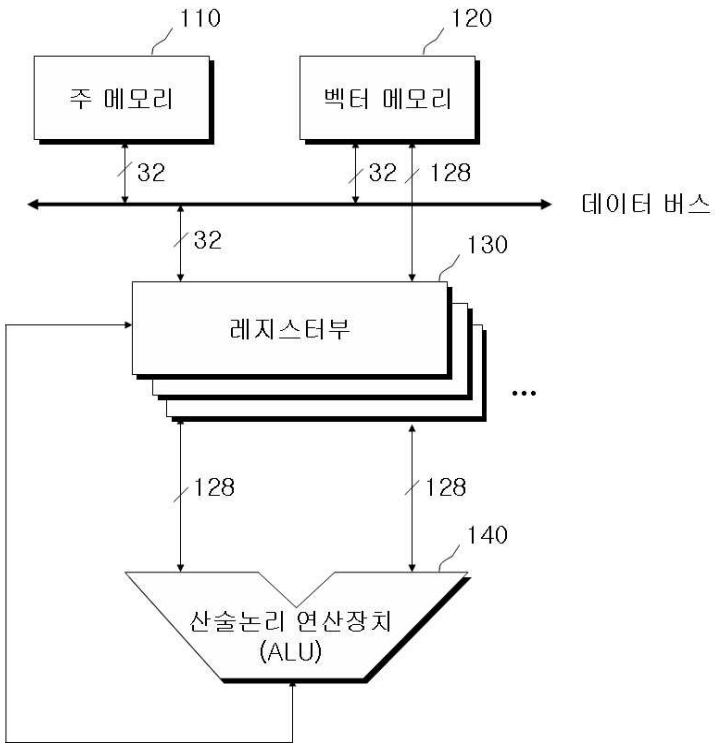
도 3a는 뱅크 충돌을 피하기 위해 메모리 뱅크의 수를 늘린 경우, 메모리 뱅크에서 출력된 데이터를 스위칭하여, 레지스터부로 출력될 데이터를 만드는 것을 설명하기 위한 참조도,

도 3b는 메모리 뱅크의 수가 병렬도와 같은 경우에, 메모리 뱅크에서 출력된 데이터를 스위칭하여, 레지스터부로 출력될 데이터를 만드는 것을 설명하기 위한 참조도,

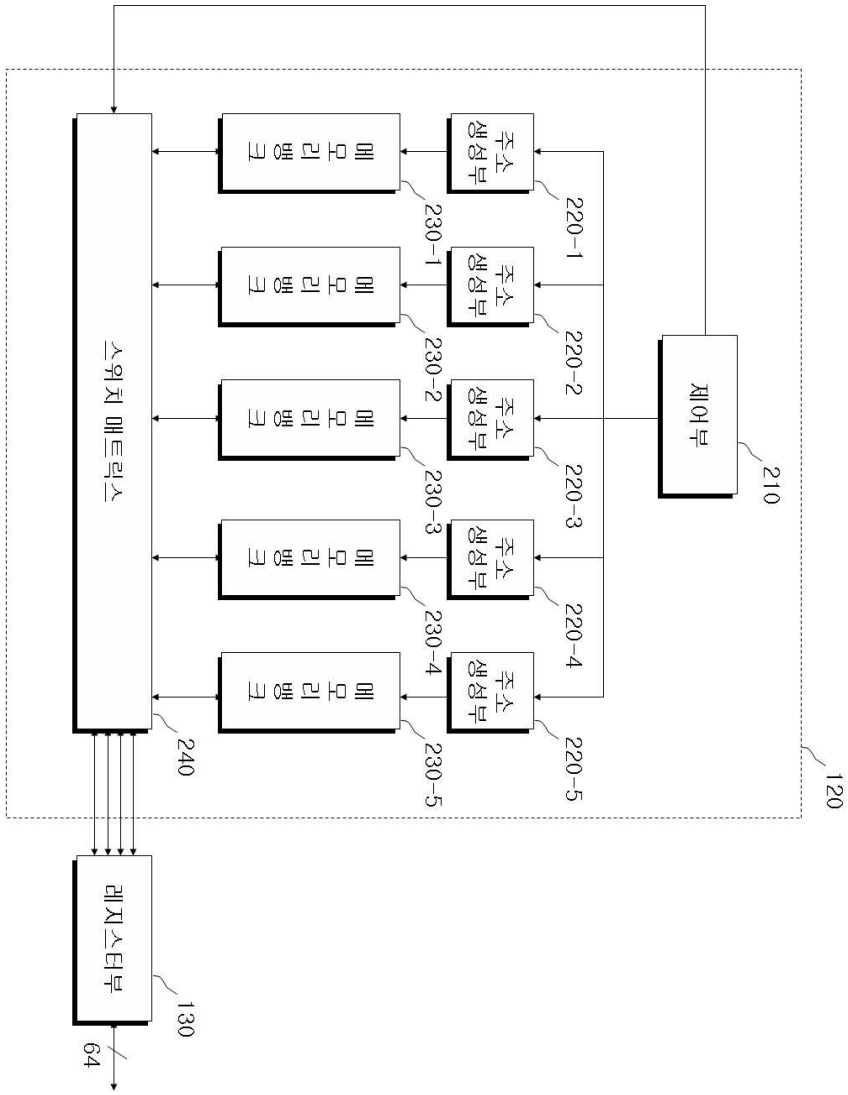
도 4는 벡터 메모리에서 SIMD 처리를 위한 데이터를 생성하는 과정의 흐름도이다.

도면

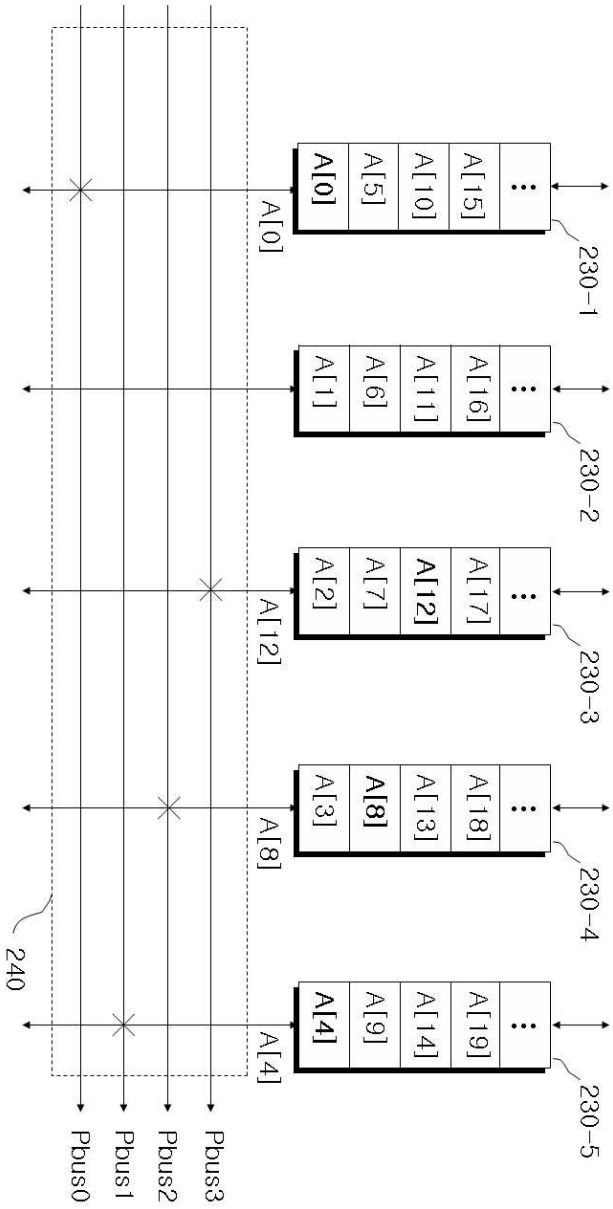
도면1



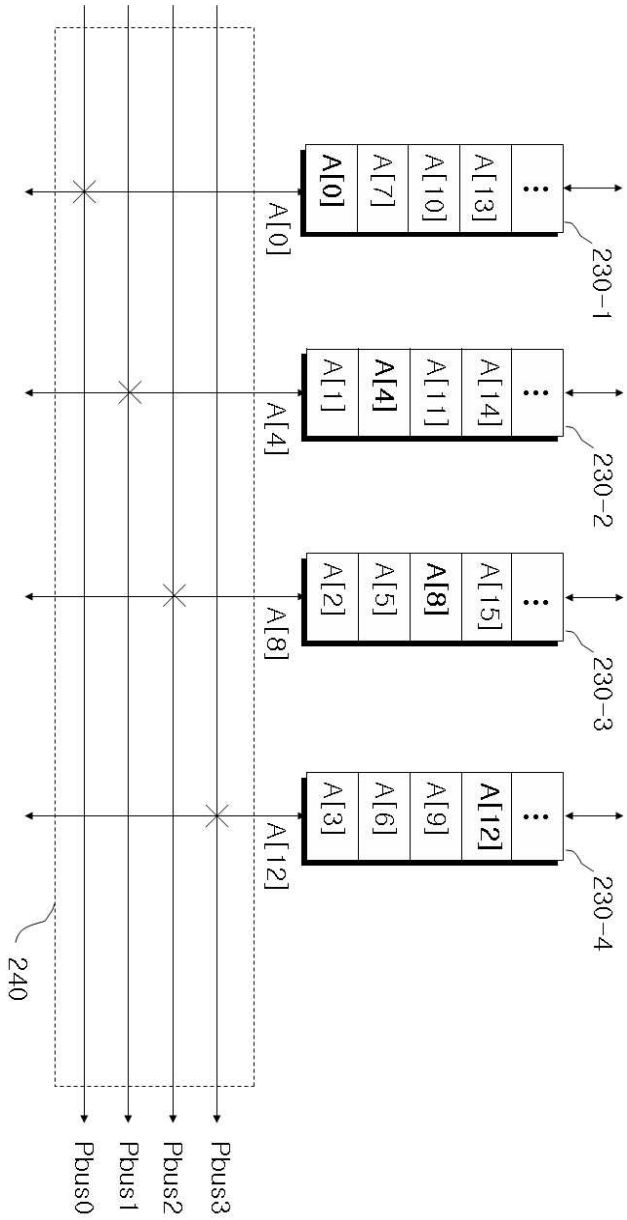
도면2


120
제어부
210
주소 생성부
220-1
220-2
220-3
220-4
220-5
메모리 뱅크
230-1
230-2
230-3
230-4
230-5
스위치 매트릭스
240
레지스터부
130
64

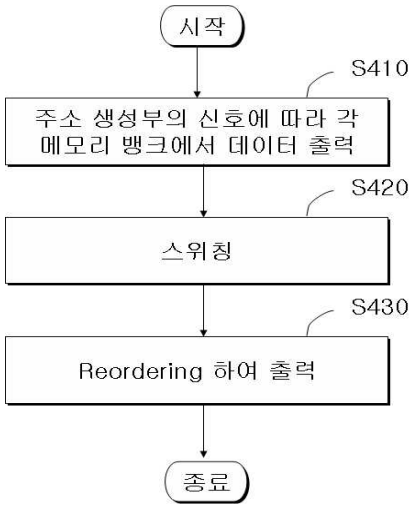
도면3a


A[0]
A[5]
A[10]
A[15]
230-1
A[1]
A[6]
A[11]
A[16]
230-2
A[2]
A[7]
A[12]
A[17]
230-3
A[3]
A[8]
A[13]
A[18]
230-4
A[4]
A[9]
A[14]
A[19]
230-5
A[0]
A[12]
A[8]
A[4]
240
Pbus0
Pbus1
Pbus2
Pbus3

도면3b


A[0]
A[7]
A[10]
A[13]
A[0]
230-1
A[1]
A[4]
A[11]
A[14]
A[4]
230-2
A[2]
A[5]
A[8]
A[15]
A[8]
230-3
A[3]
A[6]
A[9]
A[12]
A[12]
230-4
240
Pbus0
Pbus1
Pbus2
Pbus3

도면4


시작
S410
주소 생성부의 신호에 따라 각 메모리 뱅크에서 데이터 출력
S420
스위칭
S430
Reordering 하여 출력
종료