(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0035974
(43) 공개일자 2017년03월31일

(51) 국제특허분류(Int. Cl.)
G06F 9/30 (2017.01)
(52) CPC특허분류
G06F 9/30018 (2013.01)
G06F 9/30032 (2013.01)
(21) 출원번호 10-2017-7004342
(22) 출원일자(국제) 2015년08월19일
심사청구일자 2017년02월16일
(85) 번역문제출일자 2017년02월16일
(86) 국제출원번호 PCT/US2015/045827
(87) 국제공개번호 WO 2016/043908
국제공개일자 2016년03월24일
(30) 우선권주장
14/491,548 2014년09월19일 미국(US)

(71) 출원인
인텔 코포레이션
미합중국 캘리포니아 95054 산타클라라 미션 칼리지 블러바드 2200
(72) 발명자
메몬 마자르 아이
미국 텍사스주 78749 오스틴 스태거브러쉬 로드 #2225 4701
(74) 대리인
제일특허법인

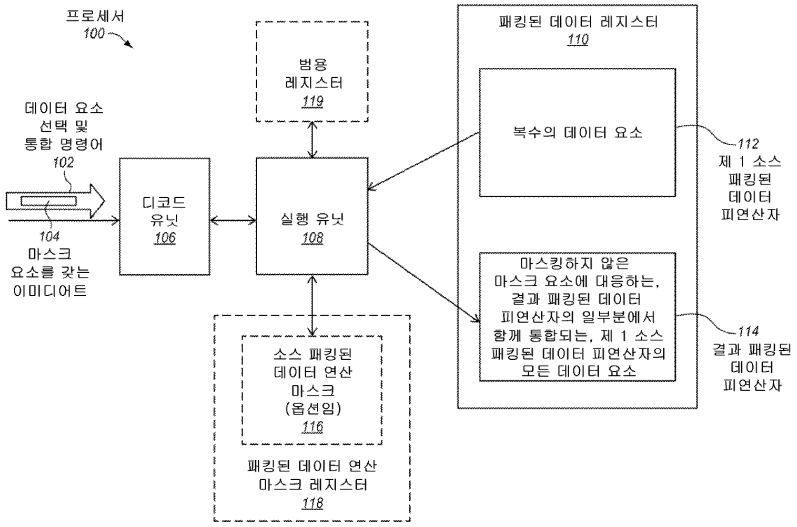
전체 청구항 수 : 총 24 항

(54) 발명의 명칭 **데이터 요소 선택 및 통합 프로세서, 방법, 시스템, 및 명령어**

(57) 요 약

프로세서는 패킹된 데이터 레지스터와, 데이터 요소 선택 및 통합 명령어를 디코딩하는 디코드 유닛을 포함한다. 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자 및 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는다. 각 마스크 요소는 동일한 상대적 위치에 있는 상이한 데이터 요소에 대응한다. 실행 유닛은 디코드 유닛과 연결된다. 명령어에 응답하여, 실행 유닛은 결과 패킹된 데이터 피연산자를 명령어에 의해 표시되는 목적지 저장 위치에 저장한다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함한다.

대 표 도



(52) CPC특허분류
G06F 9/30036 (2013.01)

명 세 서

청구범위

청구항 1

프로세서로서,

복수의 패킹된 데이터 레지스터(packed data register)와,

데이터 요소 선택 및 통합 명령어(data element selection and consolidation instruction)를 디코딩하는 디코드 유닛 - 상기 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자(source packed data operand)와, 상기 데이터 요소 선택 및 통합 명령어는 복수의 마스크 요소(mask element)를 갖는 제 2 소스 피연산자를 포함하고, 상기 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 상기 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응함 - 과,

상기 디코드 유닛에 연결된 실행 유닛 - 상기 실행 유닛은 상기 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자(result packed data operand)를 상기 데이터 요소 선택 및 통합 명령어에 의해 표시되는 목적지 저장 위치에 저장하며, 상기 결과 패킹된 데이터 피연산자는 상기 제 2 소스 피연산자의 마스킹하지 않은 마스크 요소(unmasked mask element)에 대응하는, 상기 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 상기 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함함 - 을 포함하는

프로세서.

청구항 2

제 1 항에 있어서,

상기 디코드 유닛은 상기 마스크 요소를 갖는 상기 제 2 소스 피연산자로서 이미디어트(immediate)를 갖는 상기 명령어를 디코딩하며, 상기 마스크 요소는 마스크 비트인

프로세서.

청구항 3

제 1 항에 있어서,

상기 디코드 유닛은 상기 프로세서의 한 세트의 패킹된 데이터 연산 마스크 레지스터 중의 패킹된 데이터 연산 마스크 레지스터가 되는 상기 제 2 소스 피연산자를 갖는 명령어를 디코딩하며, 상기 프로세서의 명령어 집합 중 복수의 다른 명령어는 예측 피연산자를 제공하기 위해 상기 패킹된 데이터 연산 마스크 레지스터 세트 중의 레지스터를 명시하는

프로세서.

청구항 4

제 1 항에 있어서,

상기 디코드 유닛은 마스크 비트인 상기 복수의 마스크 요소를 갖는 상기 제 2 소스 피연산자를 갖는 상기 명령어를 디코딩하는

프로세서.

청구항 5

제 1 항에 있어서,

상기 디코드 유닛은 패킹된 데이터 피연산자가 되는 상기 제 2 소스 피연산자를 갖는 상기 명령어를 디코딩하며, 상기 마스크 요소는 마스크 데이터 요소가 되는

프로세서.

청구항 6

제 1 항에 있어서,

상기 실행 유닛은, 상기 명령어에 응답하여, 상기 결과 패킹된 데이터 피연산자의 최하위 부분(a least significant portion)에서 함께, 상기 제 1 소스 패킹된 데이터 피연산자에서와 동일한 순서로 통합되는 상기 모든 데이터 요소를 포함하는 상기 결과 패킹된 데이터 피연산자를 저장하는

프로세서.

청구항 7

제 1 항에 있어서,

상기 실행 유닛은, 상기 명령어에 응답하여, 상기 결과 패킹된 데이터 피연산자의 최상위 부분(a most significant portion)에서 함께, 상기 제 1 소스 패킹된 데이터 피연산자에서와 동일한 순서로 통합되는 상기 모든 데이터 요소를 포함하는 상기 결과 패킹된 데이터 피연산자를 저장하는

프로세서.

청구항 8

제 1 항에 있어서,

상기 실행 유닛은, 상기 명령어에 응답하여, 상기 제 2 소스 피연산자의 마스킹한 마스크 요소(masked-out mask element)에 대응하는, 상기 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 제외하는 상기 통합된 데이터 요소를 갖는 상기 결과 패킹된 데이터 피연산자를 저장하며, 상기 제 1 소스 패킹된 데이터 피연산자는 마스킹하지 않는 마스크 요소에 대응하는 데이터 요소 사이에서 마스킹한 마스크 요소에 대응하는 적어도 하나의 데이터 요소를 갖는

프로세서.

청구항 9

제 1 항 내지 제 8 항 중 어느 한 항에 있어서,

상기 디코드 유닛은 상기 제 1 소스 패킹된 데이터 피연산자의 상기 데이터 요소의 크기를 표시하는 하나 이상의 비트를 갖는 명령어를 디코딩하는

프로세서.

청구항 10

제 1 항 내지 제 8 항 중 어느 한 항에 있어서,

상기 디코드 유닛은 적어도 128-비트의 비트 폭을 갖고 복수의 8-비트 데이터 요소 및 복수의 16-비트 데이터 요소로부터 선택된 복수의 데이터 요소를 갖는 명령어를 디코딩하며, 상기 목적지 저장 위치는 상기 프로세서의

패킹된 데이터 레지스터를 포함하는

프로세서.

청구항 11

제 1 항 내지 제 8 항 중 어느 한 항에 있어서,

상기 디코드 유닛은 상기 제 2 소스 피연산자의 상기 마스킹하지 않은 마스크 요소에 대응하는, 상기 제 1 소스 패킹된 데이터 피연산자의 상기 모든 데이터 요소가, 상기 제 1 소스 패킹된 데이터 피연산자 내의 상기 데이터 요소가 임의로 특정하게 배열되는 것과 상관 없이 그리고 상기 제 2 소스 피연산자 내의 상기 마스크 요소가 임의로 특정하게 배열되는 것과 상관 없이, 상기 결과 패킹된 데이터 피연산자의 상기 일부분에서 함께 통합되는 것을 표시하는 오피코드를 갖는 상기 명령어를 디코딩하는

프로세서.

청구항 12

제 1 항 내지 제 8 항 중 어느 한 항에 있어서,

상기 프로세서는 범용 프로세서를 포함하며, 상기 목적지 저장 위치는 상기 프로세서의 패킹된 데이터 레지스터를 포함하는

프로세서.

청구항 13

프로세서에서의 방법으로서,

데이터 요소 선택 및 통합 명령어를 수신하는 단계 - 상기 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자(source packed data operand)를 갖고, 상기 데이터 요소 선택 및 통합 명령어는 복수의 마스크 요소(mask element)를 갖는 제 2 소스 피연산자를 갖고, 상기 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 상기 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응함 - 와,

상기 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자(result packed data operand)를 목적지 저장 위치에 저장하는 단계 - 상기 목적지 저장 위치는 상기 데이터 요소 선택 및 통합 명령어에 의해 표시되고, 상기 결과 패킹된 데이터 피연산자는 상기 제 2 소스 피연산자의 마스킹하지 않은 마스크 요소(unmasked mask element)에 대응하며, 상기 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는 상기 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함함 - 를 포함하는

프로세서에서의 방법.

청구항 14

제 13 항에 있어서,

상기 수신하는 단계는 상기 마스크 요소를 갖는 상기 제 2 소스 피연산자로서 이미디어트(immediate)를 갖는 상기 명령어를 수신하는 단계를 포함하는

프로세서에서의 방법.

청구항 15

제 13 항에 있어서,

상기 수신하는 단계는 예측에 사용되는 한 세트의 전용의 패킹된 데이터 연산 마스크 레지스터 중의 패킹된 데이터 연산 마스크 레지스터인 상기 제 2 소스 피연산자를 갖는 상기 명령어를 수신하는 단계를 포함하는

프로세서에서의 방법.

청구항 16

제 13 항에 있어서,

상기 마스크 요소로서 마스크 비트를 갖는 상기 제 2 소스 피연산자에 액세스하는 단계를 더 포함하는

프로세서에서의 방법.

청구항 17

제 13 항에 있어서,

상기 수신하는 단계는 상기 제 1 소스 패킹된 데이터 피연산자의 상기 데이터 요소의 크기를 표시하는 하나 이상의 비트를 갖는 상기 명령어를 수신하는 단계를 포함하는

프로세서에서의 방법.

청구항 18

제 13 항에 있어서,

상기 수신하는 단계는 적어도 128-비트를 갖고 8-비트 데이터 요소 및 16-비트 데이터 요소 중 하나인 데이터 요소를 포함하는 상기 제 1 소스 패킹된 데이터 피연산자를 표시하는 명령어를 수신하는 단계를 포함하며,

상기 저장하는 단계는 상기 결과 패킹된 데이터 피연산자의 최상위 부분에서 함께 통합된 상기 모든 데이터 요소를 갖는 상기 결과 패킹된 데이터 피연산자를 상기 모든 데이터 요소가 상기 제 1 소스 패킹된 데이터 피연산자에서 출현하는 순서와 동일한 순서로 저장하는 단계를 포함하는

프로세서에서의 방법.

청구항 19

제 13 항에 있어서,

네트워크로부터 패킷을 수신하는 단계와,

상기 패킷의 일부분을 상기 제 1 소스 패킹된 데이터 피연산자로서 저장하는 단계와,

상기 패킷의 프로토콜을 디코딩하는 단계와,

상기 패킷의 상기 프로토콜을 디코딩한 것에 기초하여 상기 패킷의 상기 일부분에 있는 플로우 바이트(flow byte)의 위치를 결정하는 단계와,

상기 플로우 바이트 각각에 대해 상기 제 2 소스 피연산자 내의 마스킹하지 않은 마스크 요소를 저장하고 상기 패킷의 상기 일부분 내의 다른 바이트에 대해 상기 제 2 소스 피연산자 내의 마스킹한 요소를 저장하는 단계를 더 포함하는

프로세서에서의 방법.

청구항 20

제 13 항에 있어서,

상기 제 1 소스 패킹된 데이터 피연산자는 네트워크로부터 수신된 패킷의 데이터 요소를 가지며,

상기 방법은,

상기 결과 패킹된 데이터 피연산자의 상기 통합된 데이터 요소에 대해 암호화 동작을 수행하는 단계를 더 포함하는

프로세서에서의 방법.

청구항 21

명령어를 처리하는 시스템으로서,

인터커넥트와,

상기 인터커넥트에 연결된 프로세서 - 상기 프로세서는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자(source packed data operand)를 가지며, 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는 데이터 요소 선택 및 통합 명령어를 수신하며, 상기 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 상기 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응하며, 상기 명령어는 목적지 저장 위치를 표시하고, 상기 프로세서는 상기 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자를 상기 목적지 저장 위치에 저장하며, 상기 결과 패킹된 데이터 피연산자는 상기 제 2 소스 피연산자의 마스킹하지 않은 마스크 요소(unmasked mask element)에 대응하며, 상기 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는 상기 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함함 - 와,

상기 인터커넥트와 연결된 다이나믹 랜덤 액세스 메모리(dynamic random access memory, DRAM) - 상기 DRAM은 한 세트의 프로토콜 디코드 명령어를 저장함 - 를 포함하되,

상기 명령어 세트는 상기 프로세서에 의해 실행될 때, 상기 프로세서로 하여금,

제 1 네트워크 패킷의 프로토콜을 디코딩하고,

상기 제 2 소스 피연산자를 생성하여 상기 제 1 소스 패킹된 데이터 피연산자에 저장된 제 2 네트워크 패킷의 헤더의 적어도 일부분 내의 플로우 요소(flow element)의 마스킹을 해제하고 비 플로우 요소(non-flow element)를 마스킹하는 것을 포함하는 동작을 수행하게 하는

명령어를 처리하는 시스템.

청구항 22

제 21 항에 있어서,

상기 제 2 소스 피연산자는 이미디어트(immediate)를 포함하는

명령어를 처리하는 시스템.

청구항 23

청구항 제 13 항 내지 제 20 항 중 어느 한 항의 방법을 수행하기 위한 수단을 포함하는

장치.

청구항 24

머신에 의해 실행되는 경우, 상기 머신으로 하여금 청구항 제 13 항 내지 제 20 항 중 어느 한 항의 방법을 수행하게 하도록 작용하는 명령어를 저장하는 비일시적 머신 판독가능한 저장 매체를 포함하는

제조 물품.

발명의 설명

기 술 분 야

[0001] 본 명세서에서 기술되는 실시예는 일반적으로 프로세서에 관한 것이다. 특히, 본 명세서에서 기술되는 실시예는 일반적으로 프로세서에서 패킹된 데이터 요소(packed data element)를 처리하는 것에 관한 것이다.

배 경 기 술

[0002] 많은 프로세서는 단일 명령어 다중 데이터(Single Instruction, Multiple Data, SIMD) 아키텍처를 갖고 있다. SIMD 아키텍처에서, 다중 데이터 요소는 하나의 레지스터 또는 메모리 위치 내에서 패킹된 데이터(packed data) 또는 벡터 데이터로서 패킹될 수 있다. 패킹된 데이터에서, 레지스터 또는 다른 저장 위치의 비트는 논리적으로 데이터 요소의 시퀀스로 나누어질 수 있다. 예를 들면, 128-비트 폭의 패킹된 데이터 레지스터는 두 개의 64-비트 데이터 요소, 네 개의 3-비트 요소, 여덟 개의 16-비트 데이터 요소, 또는 열여섯 개의 8-비트 데이터 요소를 가질 수 있다. 각 데이터 요소는 따로따로 그리고/또는 서로 독립적으로 연산될 수 있는 별개의 개별적인 데이터 조각(예를 들면, 픽셀 컬러, 복소수의 컴포넌트 등)을 나타낼 수 있다.

[0003] SIMD 아키텍처에서, 패킹된 데이터 명령어, 벡터 명령어, 또는 SIMD 명령어는 패킹된 데이터 피연산자(packed data operand)의 다중 데이터 요소, 또는 두 개의 패킹된 데이터 피연산자 내의 데이터 요소의 다중 쌍을 동시에 및/또는 병렬로 연산할 수 있다. 프로세서는 패킹된 데이터 명령어에 응답하여 다중 연산을 동시에 또는 병렬로 수행하는 병렬 실행 하드웨어를 가질 수 있다.

[0004] 다양한 여러 종류의 패킹된 데이터 명령어는 본 기술에서 공지되어 있다. 패킹된 데이터 명령어의 하나의 부류는 셔플 명령어(shuffle instruction)이다. 셔플 명령어는 소스 패킹된 데이터 피연산자(source packed data operand)로부터의 데이터 요소를 각 데이터 요소가 셔플되게 하는 대응하는 셔플 제어 비트 세트를 사용하여 결과 패킹된 데이터 피연산자(result packed data operand) 내의 다른 위치로 셔플하는데 사용될 수 있다. 그러나 그와 같은 셔플 명령어를 특정 애플리케이션에 사용함에 있어서의 한 가지 단점은 셔플되는 각 데이터 요소에 대해 많은 셔플 제어 비트가 필요하다는 것이다. 이것은 그러한 셔플 제어에 필요한 비트 수를 특정 애플리케이션의 한계를 넘어(예를 들면, 명령어의 이미디어트(immediate)에 알맞을 수 있는 비트 수를 넘어) 증가시키는 경향이 있을 수 있다. 또한, 이것은 셔플 제어 비트를 생성 또는 발생시키는데 추가 시간이 필요하다.

도면의 간단한 설명

[0005] 본 발명은 다음과 같은 설명 및 실시예를 예시하는데 사용되는 첨부 도면을 참조함으로써 가장 잘 이해될 수 있다. 도면에서,

도 1은 데이터 요소 선택 및 통합 명령어(data element selection and consolidation instruction)의 실시예를 수행하도록 동작 가능한 프로세서의 실시예의 블록도이다.

도 2는 데이터 요소 선택 및 통합 명령어의 실시예를 수행하는 프로세서에서의 방법의 실시예의 블록 흐름도이다.

도 3은 데이터 요소 선택 및 통합 연산의 실시예의 블록도이다.

도 4는 비트 마스크를 이용한 바이트 선택 및 통합 연산(byte selection with bit mask and consolidation operation)의 특정한 예시적인 실시예의 블록도이다.

도 5는 바이트 마스크 피연산자를 이용한 바이트 선택 및 통합 연산의 특정한 예시적인 실시예의 블록도이다.

도 6은 일부 실시예에서 데이터 요소 선택 및 통합 명령어가 네트워크 패킷의 데이터 요소를 처리하기 위해 어떻게 사용될 수 있는지를 도시하는 블록도이다.

도 7(a) 내지 도 7(c)는 데이터 요소 선택 및 통합 명령어의 여러 가지 실시예에 관한 블록도이다.

도 8은 적합한 한 세트의 패킹된 데이터 레지스터의 예시적인 실시예의 블록도이다.

도 9는 적합한 한 세트의 패킹된 데이터 연산 마스크 레지스터(packed data operation mask register)의 예시적인 실시예의 블록도이다.

도 10은 패킹된 데이터 연산 마스크 레지스터의 예시적인 실시예의 도면으로, 패킹된 데이터 연산 마스크로서 그리고/또는 마스킹을 위해 사용되는 비트 수가 패킹된 데이터 폭 및 데이터 요소 폭에 좌우된다는 것을 보여주는 도면이다.

도 11(a)는 순차적 파이프라인(in-order pipeline)의 실시예 및 레지스터 리네이밍 비순차적 발행(register renaming out-of-order issue)/실행 파이프라인(execution pipeline)의 실시예를 예시하는 블록도이다.

도 11(b)는 실행 엔진 유닛에 연결되고 메모리 유닛에도 모두 다 연결된 프론트 엔드 유닛(front end unit)을 포함하는 프로세서 코어의 실시예의 블록도이다.

도 12(a)는 단일 프로세서 코어와 함께, 이 코어의 온-다이 인터커넥트 네트워크(on-die interconnect network)와의 접속과, 이 코어의 레벨 2(L2) 캐시의 로컬 서브세트의 실시예의 블록도이다.

도 12(b)는 도 12(a)의 프로세서 코어의 일부의 확장된 모양의 실시예의 블록도이다.

도 13은 하나보다 많은 코어를 가질 수 있고, 통합된 메모리 제어기를 가질 수 있고, 그리고 통합된 그래픽스를 가질 수 있는 프로세서의 실시예의 블록도이다.

도 14는 컴퓨터 아키텍처의 제 1 실시예의 블록도이다.

도 15는 컴퓨터 아키텍처의 제 2 실시예의 블록도이다.

도 16은 컴퓨터 아키텍처의 제 3 실시예의 블록도이다.

도 17은 시스템-온-칩 아키텍처의 실시예의 블록도이다.

도 18은 본 발명의 실시예에 따라 소스 명령어 집합 내의 이진 명령어를 타겟 명령어 집합 내의 이진 명령어로 변환하는 소프트웨어 명령어 변환기의 사용에 관한 블록도이다.

발명을 실시하기 위한 구체적인 내용

[0006] 본 명세서에는 데이터 요소 선택 및 통합 명령어(data element selection and consolidation instruction), 명령어를 실행하는 프로세서, 명령어를 처리 또는 실행할 때 프로세서에 의해 수행되는 방법, 및 명령어를 처리 또는 실행하는 하나 이상의 프로세서를 포함하는 시스템이 개시된다. 다음과 같은 설명에서, 많은 특정 세부사항(예를 들면, 특정 명령어 연산, 데이터 포맷, 프로세서 구성, 마이크로아키텍처적 세부사항, 동작 시퀀스 등)이 언급된다. 그러나 실시예는 이러한 특정 세부사항 없이 실시될 수 있다. 다른 경우에, 널리 공지된 회로, 구조 및 기술은 설명의 이해를 어렵게 하지 않기 위해 상세하게 도시되지 않는다.

[0007] 도 1은 데이터 요소 선택 및 통합 명령어(102)의 실시예를 수행하도록 동작할 수 있는 프로세서(100)의 실시예의 블록도이다. 일부 실시예에서, 프로세서는 범용 프로세서(예를 들면, 데스크톱, 랩톱, 또는 다른 컴퓨터에서 사용되는 형태의 범용 마이크로프로세서 또는 중앙 처리 유닛(central processing unit, CPU))일 수 있다. 일 양태에서, 프로세서는 네트워크 및/또는 패킷 관련 처리를 통합하는 시스템-온-칩의 범용 코어를 나타낼 수 있다. 일 양태에서, 데이터 요소 선택 및 통합 명령어는 범용 프로세서 또는 코어가 네트워크 및/또는 패킷 처리를 더욱 효과적으로 수행하게 할 수 있다. 이와 달리, 프로세서는 특수 목적 프로세서일 수 있다. 적합한 특수 목적 프로세서의 예는 이것으로 제한되는 것은 아니지만, 네트워크 프로세서, 통신 프로세서, 패킷 프로세서, 임베디드 패킷 프로세싱 엔진, 스위치 칩, 데이터플랜 프로세서, 암호화 프로세서, 그래픽스 프로세서, 코-프로세서, 임베디드 프로세서, 디지털 신호 프로세서(digital signal processor, DSP), 및 컨트롤러(예를 들면, 마이크로컨트롤러)를 포함한다. 프로세서는 각종 복합 명령어 집합 컴퓨팅(complex instruction set computing, CISC) 아키텍처, 축소 명령어 집합 컴퓨팅(reduced instruction set computing, RISC) 아키텍처, 매우 긴 명령어 워드(very long instruction word, VLIW) 아키텍처, 하이브리드 아키텍처, 다른 유형의 아키텍처 중 임의의 아키텍처를 가질 수 있거나, 상이한 아키텍처의 조합을 가질 수 있다(예를 들면, 상이한 코어는 상이한 아키텍처를 가질 수 있다).

[0008] 연산 중에, 프로세서(100)는 데이터 요소 선택 및 통합 명령어(102)를 수신할 수 있다. 예를 들면, 명령어는 인터커넥트 상의 메모리로부터 수신될 수 있다. 명령어는 매크로명령어, 어셈블리 언어 명령어, 머신 코드

명령어, 또는 기타 명령어나 프로세서의 명령어 집합의 제어 신호를 나타낼 수 있다. 일부 실시예에서, 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자(source packed data operand)(112)를 (예를 들면, 하나 이상의 필드 또는 한 세트의 비트를 통해) 명시적으로 지정할 수 있고, 그렇지 않으면 표시할 수 있고(예를 들면, 암시적으로 표시할 수 있고), 또는 그렇지 않으면 가질 수 있다. 일부 실시예에서, 데이터 요소 선택 및 통합 명령어는 대응하는 복수의 마스크 요소(mask element)를 갖는 제 2 소스 패킹된 피연산자(104)를 (예를 들면, 하나 이상의 필드 또는 한 세트의 비트를 통해) 명시적으로 지정할 수 있고, 그렇지 않으면 표시할 수 있고(예를 들면, 암시적으로 표시할 수 있고), 또는 그렇지 않으면 가질 수 있다(예를 들면, 이미디어트(immediate)로서 가질 수 있다). 예시된 실시예에서, 제 2 소스 피연산자(104)는, 예를 들어, 몇 가지 예를 들자면, 4-비트 이미디어트(imm4), 8-비트 이미디어트(imm8), 16-비트 이미디어트(imm16), 또는 32-비트 이미디어트(imm32)와 같은 명령어의 이미디어트를 포함한다. 이미디어트는 마스크 비트를 마스크 요소로서 가질 수 있다. 각각의 마스킹하지 않는 마스크 비트(unmasked mask bit)는 제 1 값(예를 들면, 하나의 가능한 관례에 따라 이진수 1)을 갖는 반면, 각각의 마스킹하는 마스크 비트(masked-out mask bit)는 상이한 제 2 값(예를 들면, 하나의 가능한 관례에 따라 이진수 0)을 가질 수 있다. 다른 실시예에서, 선택적으로 제 2 소스 피연산자(104)는 한 세트의 패킹된 데이터 연산 마스크 레지스터(packed data operation mask register)(118)에 선택적으로 저장될 수 있는 소스 패킹된 데이터 피연산자 마스크(source packed data operation mask)(116)일 수 있다. 또 다른 실시예에서, 제 2 소스 피연산자(104)는 선택적으로 한 세트의 범용 레지스터(119)에 저장될 수 있다. 또 다른 실시예에서, 선택적으로 제 2 소스 피연산자(104)는 선택적으로 한 세트의 패킹된 데이터 레지스터(110)에 저장되는 제 2 소스 패킹된 데이터 피연산자일 수 있다. 명령어는 또한 결과 패킹된 데이터 피연산자(result packed data operand)(114)가 저장되는 목적지 저장 위치를 지정하거나 그렇지 않으면 표시할 수 있다. 하나의 예로서, 명령어는 피연산자를 위한 레지스터, 메모리 위치, 또는 다른 저장 위치를 지정하는 소스 및/또는 목적지 피연산자 지정 필드를 가질 수 있다. 이와 달리, 이러한 피연산자 중 하나 이상의 피연산자는 선택적으로 명령어에 암시적일 수 있다(예를 들면, 명령어의 오피코드(opcode)에 암시적일 수 있다).

[0009] 다시 도 1을 참조하면, 프로세서는 디코드 유닛 또는 디코더(106)를 포함한다. 디코드 유닛은 데이터 요소 선택 및 통합 명령어를 수신하고 디코딩할 수 있다. 디코드 유닛은 상대적으로 상위 레벨의 데이터 요소 선택 및 통합 명령어를 반영하고, 표현하고, 그리고/또는 그로부터 도출되는 상대적으로 하위 레벨의 하나 이상의 명령어 또는 제어 신호(예를 들면, 하나 이상의 마이크로명령어, 마이크로연산, 마이크로코드 엔트리 포인트, 디코딩된 명령어 또는 제어 신호 등)를 출력할 수 있다. 일부 실시예에서, 디코드 유닛은 데이터 요소 선택 및 통합 명령어를 수신하는 하나 이상의 입력 구조(예를 들면, 포트(들), 인터커넥트(들), 인터페이스)와, 디코드 유닛과 연결되어 데이터 요소 선택 및 통합 명령어를 인식하고 디코딩하는 명령어 인식 및 디코드 로직과, 디코드 유닛과 연결되어 하위 레벨 명령어(들) 또는 제어 신호(들)을 출력하는 하나 이상의 출력 구조(예를 들면, 포트(들), 인터커넥트(들), 인터페이스)를 포함할 수 있다. 디코드 유닛은, 이것으로 제한되는 것은 아니지만, 본 기술에서 공지된 디코드 유닛을 구현하는데 사용되는 마이크로코드 판독 전용 메모리(read-only memory, ROM), 룩업 테이블, 하드웨어 구현물, 프로그래머블 로직 어레이(programmable logic array, PLA), 및 다른 메커니즘을 비롯한 다양한 여러 메커니즘을 사용하여 구현될 수 있다.

[0010] 일부 실시예에서, 디코드 유닛에 직접 제공되는 데이터 요소 선택 및 통합 명령어 대신에, 명령어 에뮬레이터(emulator), 번역기(translator), 모퍼(morpher), 해석기(interpreter), 또는 다른 명령어 변환 모듈이 선택적으로 사용될 수 있다. 다양한 유형의 명령어 변환 모듈이 본 기술에서 공지되어 있으며 소프트웨어, 하드웨어, 펌웨어, 또는 이들의 조합으로 구현될 수 있다. 일부 실시예에서, 명령어 변환 모듈은 프로세서의 외부에, 예를 들면, 별개의 다이 상에 및/또는 (예를 들어, 정적, 동적, 또는 런타임 에뮬레이션 모듈로서) 메모리 내에 위치될 수 있다. 예로서, 명령어 변환 모듈은 제 1 명령어 집합의 명령어일 수 있는 데이터 요소 선택 및 통합 명령어를 수신할 수 있으며, 데이터 요소 선택 및 통합 명령어를 상이한 제 2 명령어 집합의 명령어일 수 있는 하나 이상의 대응하는 중간 명령어 또는 제어 신호로 에뮬레이트하거나, 번역하거나, 모핑하거나, 해석하거나, 그렇지 않으면 변환할 수 있다. 제 2 명령어 집합의 하나 이상의 중간 명령어 또는 제어 신호는 디코드 유닛(예를 들면, 디코드 유닛(106))에 제공될 수 있으며, 디코드 유닛은 이를 프로세서의 고유 하드웨어(예를 들면, 하나 이상의 실행 유닛)에 의해 실행 가능한 하나 이상의 하위 레벨 명령어 또는 제어 신호로 디코딩할 수 있다.

[0011] 다시 도 1을 참조하면, 프로세서(100)는 또한 한 세트의 패킹된 데이터 레지스터(110)를 포함한다. 일부 실시예에서, 프로세서는 한 세트의 범용 레지스터(119)를 포함할 수 있다. 일부 실시예에서, 프로세서는 또한 선택적으로 한 세트의 패킹된 데이터 연산 마스크 레지스터(118)를 포함할 수 있다. 이들 레지스터는 각기 데이터를 저장하도록 작용하는 온-다이 저장 위치를 나타낼 수 있다. 예를 들면, 패킹된 데이터 레지스터 각각은 패킹된

데이터, 벡터 데이터, 또는 단일 명령어 다중 데이터(SIMD) 데이터를 저장할 수 있다. 이러한 레지스터는 소프트웨어 및/또는 프로그래머에 가시적이고 그리고/또는 피연산자를 식별하기 위해 프로세서의 명령어 집합 중의 명령어에 의해 지시되는 레지스터인 아키텍처적으로 가시적이거나 아키텍처적인 레지스터를 나타낼 수 있다. 이러한 아키텍처적 레지스터는 특정 마이크로아키텍처 내의 다른 비 아키텍처적인 레지스터(non-architectural register)(예를 들면, 임시 레지스터, 재정렬 버퍼, 리타이어먼트 레지스터(retirement register) 등)와 정반대이다. 이러한 레지스터는 널리 공지된 기술을 사용하여 상이한 마이크로아키텍처로 상이한 방법으로 구현될 수 있으며 임의의 특정한 형태의 디자인으로 제한되지 않는다. 적합한 유형의 레지스터의 예는, 이것으로 제한되는 것은 아니지만, 전용의 물리 레지스터, 레지스터 리네이밍을 사용하는 동적 할당된 물리 레지스터, 및 이들의 조합을 포함한다.

[0012] 일부 실시예에서, 제 1 소스 패킹된 데이터 피연산자(112)는 선택적으로 패킹된 데이터 레지스터에 저장될 수 있으며, 결과 패킹된 데이터 피연산자(114)가 저장되는 목적지 저장 위치는 선택적으로 같은 또는 다른 패킹된 데이터 레지스터 중 어느 하나일 수 있다. 대안으로, 메모리 위치 또는 다른 저장 위치는 선택적으로 이러한 피연산자 중 하나 이상에 사용될 수 있다. 일부 실시예에서, 소스 패킹된 데이터 피연산자(예를 들면, 제 1 소스 패킹된 데이터 피연산자(112))에 사용되는 패킹된 데이터 레지스터는 결과 패킹된 데이터 피연산자(114)의 목적지 저장 위치로서 선택적으로 재사용될 수 있다. 예를 들면, 단일의 소스/목적지 레지스터는 소스 및 결과 패킹된 데이터 피연산자 모두 다에 사용되는 것으로 암시적으로 또는 묵시적으로 이해될 수 있다.

[0013] 다시 도 1을 참조하면, 실행 유닛(108)은 디코드 유닛(106) 및 패킹된 데이터 레지스터(110)와 연결된다. 실행 유닛은 하나 이상의 디코딩된 명령어나 제어 신호, 또는 그렇지 않으면 데이터 요소 선택 및 통합 명령어를 나타내고 그리고/또는 그로부터 도출되는 변환된 명령어나 제어 신호를 수신할 수 있다. 실행 유닛은 또한 제 1 소스 패킹된 데이터 피연산자(112) 및 복수의 마스크 요소를 갖는 제 2 소스 피연산자(104)(예를 들면, 이미디어트(104))를 수신할 수 있다. 데이터 요소 선택 및 통합 명령어에 응답하여 그리고/또는 그의 결과로서(예를 들면, 하나 이상의 명령어 또는 그 명령어로부터 디코딩된 하나 이상의 제어 신호에 응답하여), 실행 유닛은 결과 패킹된 데이터 피연산자(114)를 그 명령어에 의해 표시된 목적지 저장 위치에 저장하도록 동작할 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자(114)는 이미디어트(104) 또는 다른 제 2 소스 피연산자(104)의 마스킹하지 않는 마스크 요소에 대응하는, 그 결과 패킹된 데이터 피연산자(114)의 일부분에 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자(112)의 모든 데이터 요소를 포함할 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하는 마스크 요소에 대응하는 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 제외할 수 있다. 일부 실시예에서, 제 1 소스 패킹된 데이터 피연산자의 마스킹하지 않는 데이터 요소는 결과 패킹된 데이터 피연산자의 최하위 부분에서 함께 통합될 수 있다. 다른 실시예에서, 제 1 소스 패킹된 데이터 피연산자의 마스킹하지 않는 데이터 요소는 결과 패킹된 데이터 피연산자의 최상위 부분에서 함께 통합될 수 있다. 일부 실시예에서, 본 발명의 범위가 이렇게 제한되지는 않겠지만, 결과 패킹된 데이터 피연산자는 도 3 내지 도 5에서 도시되고 기술된 피연산자 중의 임의의 것일 수 있다.

[0014] 일부 실시예에서, 결과 패킹된 데이터 피연산자 내에 마스킹하지 않는 데이터 요소를 통합하는 것은, 명시적으로 지정되는 대신 및/또는 명령어에 융통적인 대신, 명령어(예를 들면, 명령어의 오피코드)에 암시적일 수 있고 그리고/또는 명령어에 대해 정해진 것일 수 있다. 예를 들면, 명령어는 마스킹하지 않는 데이터 요소가 결과 패킹된 데이터 피연산자 내에 통합되는 것을 표시하거나 명시하는 오피코드를 가질 수 있다. 즉, 오피코드 및/또는 명령어는 마스킹하지 않는 데이터 요소를 결과 패킹된 데이터 피연산자에 통합하는데 특별히 전용될 수 있다. 오피코드 및/또는 명령어에 대해 그와 같이 전용된 및/또는 정해진 및/또는 암시적인 특징을 사용하면 융통성 있는 명령어(예를 들면, 완전 셔플(full shuffle) 또는 치환(permute) 명령어)와 함께 사용될 명시적인 제어(예를 들면, 명시적인 제어 필드)를 만들어서 사용할 필요를 회피하는데 도움이 될 수 있다. 일 양태에서, 마스킹하지 않는 데이터 요소는, 제 1 소스 패킹된 데이터 피연산자 내의 데이터 요소가 임의로 특정하게 배열되는 것과 상관없이, 그리고 제 2 소스 피연산자 내의 마스크 요소가 임의로 특정하게 배열되는 것과 상관 없이, 결과 패킹된 데이터 피연산자 내에 통합될 수 있다.

[0015] 유리하게, 데이터 요소 선택 및 통합 명령어/피연산자는 데이터 요소의 서브세트가 단일 명령어의 실행의 범위 내에서 선택되어 함께 압축되게 한다. 압축 동작은 소스 데이터 요소의 선택된 서브세트만이 추가 처리를 받게 하는 실시예에서 유리하다. 일 예로서, 시작 패킷을 디코딩한 이후, 후속 패킷이 동일한 플로우에 속한다는 것을 확인하는 빠른 검사가 수행될 수 있도록 하기 위해 단지 특정 플로우 바이트(flow byte)(예를 들면, 소스와 목적지 어드레스 및 소스와 목적지 포트를 정의하는 4-튜플)만이 그러한 후속 패킷으로 선택되고 함께 통합될 수 있다. 다른 예로서, 해시, 체크섬(checksum), 또는 다른 암호화 기능이 상대적으로 엔트로피가 더 많은 바이

트에 대해 수행될 수 있도록 하기 위해 상대적으로 엔트로피가 더 많은 (예를 들어, 패킷의) 소스 데이터 요소의 서브세트만이 선택되고 함께 통합될 수 있다. 이러한 방식으로, 선택되지 않을 수 있는 상대적으로 적은 엔트로피의 데이터 요소에 대해 암호화 기능이 수행될 필요가 없다. 다른 가능한 용도는 그래픽 데이터의 서브샘플링이다. 다른 용도는 본 기술에서 통상의 지식을 갖고 본 개시의 이득을 받는 자에게 자명할 것이다. (예를 들면, 전체 피연산자를 시프트하는 하나 이상의 명령어, 마스킹되는 논리 연산을 수행하는 하나 이상의 명령어 등에 의해) 원하는 소스 데이터 요소 및 부가적인 명령어를 선택하여 그 선택된 소스 데이터 요소를 함께 통합하는 하나의 명령어를 사용하는 것 또한 가능할 것이다. 그러나, 단일의 데이터 요소 선택 및 통합 명령어는 특히 데이터 요소가 많은 경우에 복수의 또는 잠재적으로 많은 명령어가 연루되는 그런 소프트웨어 구현보다 성능을 높이는데 도움이 될 수 있다.

[0016] 실행 유닛 및/또는 프로세서는 데이터 요소 선택 및 통합 명령어에 응답하여 및/또는 그의 결과로서 (예를 들면, 데이터 요소 선택 및 통합 명령어로부터 디코딩된 하나 이상의 명령어 또는 제어 신호에 응답하여) 데이터 요소 선택 및 통합 명령어를 수행하며 그리고/또는 결과를 저장하도록 동작 가능한 특정 또는 특별한 로직 (예를 들면, 트랜지스터, 집적 회로, 또는 잠재적으로 펌웨어(예를 들면, 비휘발성 메모리에 저장된 명령어) 및/또는 소프트웨어와 조합된 다른 하드웨어)를 포함할 수 있다. 예로서, 실행 유닛은 산술 로직 유닛, 로직 유닛 등을 포함할 수 있다. 일부 실시예에서, 실행 유닛은 소스 피연산자를 수신하는 하나 이상의 입력 구조(예를 들면, 포트(들), 인터커넥트(들), 인터페이스)와, 그와 연결되어 소스 피연산자를 수신하고 처리하며 결과의 피연산자를 생성하는 회로 또는 로직과, 그와 연결되어 결과의 피연산자를 출력하는 하나 이상의 출력 구조(예를 들면, 포트(들), 인터커넥트(들), 인터페이스))를 포함할 수 있다. 일부 실시예에서, 실행 유닛은 소스 피연산자를 처리하고 결과의 피연산자를 생성하는 회로 또는 로직은 선택적으로 데이터 요소 선택 및 라우팅 로직을 포함할 수 있다. 예를 들면, 데이터 요소를 선택하는 멀티플렉서 및 선택된 데이터 요소를 라우팅하는 인터커넥트가 있다.

[0017] 설명을 불명료하게 하지 않기 위해, 비교적 간단한 프로세서가 도시되고 기술된다. 그러나 프로세서는 다른 널리 공지된 프로세서 컴포넌트를 선택적으로 포함할 수 있다. 그러한 컴포넌트의 가능한 예는 이것으로 제한되는 것은 아니지만, 범용 레지스터, 상태 레지스터(때로는 플래그 레지스터라고 호칭함), 시스템 제어 레지스터, 명령어 페치 유닛, 프리페치 버퍼(prefetch buffer), 한 레벨 이상의 캐시(예를 들면, 레벨 1(L1) 명령어 캐시, L1 데이터 캐시, 및 L2 데이터/명령어 캐시), 명령어 번역 룩어사이드 버퍼(instruction translation lookaside buffer, TLB), 데이터 TLB, 분기 예측 유닛, 비순차적 실행 유닛(예를 들면, 명령어 스케줄링 유닛, 레지스터 리네임 및/또는 할당 유닛, 명령어 디스패치 유닛, 재정렬 버퍼(reorder buffer, ROB), 예약 스테이션(reservation station), 메모리 정렬 버퍼, 리타이먼트 유닛(retirement unit) 등), 버스 인터페이스 유닛, 어드레스 생성 유닛, 디버그 유닛, 성능 모니터 유닛, 전력 관리 유닛, 프로세서에 포함된 다른 컴포넌트, 및 이들의 다양한 조합을 포함한다. 그러한 컴포넌트는 본 기술에서 공지된 각종의 여러 적합한 조합 및/또는 구성에서 함께 결합될 수 있다. 실시예는 그와 같은 임의의 조합 또는 구성으로 제한되지 않는다. 더욱이, 실시예는 복수의 코어를 갖는 프로세서에 포함될 수 있고, 그 중 적어도 하나의 코어는 데이터 요소 선택 및 통합 명령어를 수행하도록 동작한다.

[0018] 도 2는 데이터 요소 선택 및 통합 명령어의 실시예를 수행하는 프로세서에서의 방법(220)의 실시예의 블록 흐름도이다. 일부 실시예에서, 방법(220)은 도 1의 프로세서(100)에 의해 및/또는 프로세서 내에서 수행될 수 있다. 본 명세서에서 프로세서(100)에 대해 기술된 컴포넌트, 특징, 및 특정한 선택적인 세부내용은 선택적으로 방법(220)에도 적용된다. 대안으로, 방법(220)은 유사한 또는 상이한 프로세서 또는 다른 장치에 의해 및/또는 그 내부에서 수행될 수 있다. 더욱이, 프로세서(100)는 방법(220)과 동일한, 유사한, 또는 상이한 방법을 수행할 수 있다.

[0019] 블록(221)에서, 방법은 데이터 요소 선택 및 통합 명령어를 수신하는 단계를 포함한다. 다양한 양태에서, 명령어는 프로세서에서 또는 그의 일부분(예를 들면, 명령어 페치 유닛, 디코드 유닛, 버스 인터페이스 유닛 등)에서 수신될 수 있다. 다양한 양태에서, 명령어는 오프-프로세서 및/또는 오프-다이 소스로부터 (예를 들면, 메모리, 인터커넥트 등으로부터), 또는 온-프로세서 및/또는 온-다이 소스로부터 (예를 들면, 명령어 캐시, 명령어 큐 등으로부터) 수신될 수 있다. 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자를 지정하거나, 표시하거나, 그렇지 않으면 가질 수 있으며, 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 지정하거나, 표시하거나, 그렇지 않으면 가질 수 있다. 제 2 소스 피연산자의 각 마스크 요소는 (동일한 비트 위치에서, 피연산자 내의 동일한 상대 위치에서, 기타 등등) 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응할 수 있다.

[0020] 블록(222)에서, 결과 패킹된 데이터 피연산자는 데이터 요소 선택 및 통합 명령어에 응답하여 및/또는 그의 결과로서 목적지 저장 위치에 저장될 수 있다. 목적지 저장 위치는 데이터 요소 선택 및 통합 명령어에 의해 지정되거나 그렇지 않으면 표시될 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자는 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함할 수 있으며, 제 1 소스 패킹된 데이터 피연산자는 결과 패킹된 데이터 피연산자의 일부에서 함께 통합된, 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응한다.

[0021] 예시된 방법은 아키텍처적 연산(예를 들면, 소프트웨어 관점에서 가시적인 연산)을 수반한다. 다른 실시예에서, 방법은 선택적으로 하나 이상의 마이크로아키텍처적 연산을 포함할 수 있다. 예로서, 명령어는 페치될 수 있으며, 디코딩되고, 비순차적으로 스케줄되며, 소스 피연산자는 액세스될 수 있으며, 실행 유닛은 마이크로아키텍처적 연산을 수행하여 명령어를 실시할 수 있다. 일부 실시예에서, 이것이 필수 사항은 아니지만, 명령어를 실시하는 마이크로아키텍처적 연산은, 선택적으로, 비트 마스크의 각 비트를 제 1 소스 패킹된 데이터 피연산자의 데이터 요소와 동일한 크기로 확장하고 확장된 데이터 요소 마스크 및 제 1 소스 패킹된 데이터 피연산자에 대해 논리 연산(예를 들면, 논리 AND)을 수행하는 것일 수 있다.

[0022] 도 3은 데이터 요소 선택 및 통합 명령어의 실시예에 대응하여 수행될 수 있는 데이터 요소 선택 및 통합 연산(330)의 실시예를 예시하는 블록도이다. 명령어는 복수의 패킹된 데이터 요소(E0-EN)를 갖는 제 1 소스 패킹된 데이터 피연산자(312)를 (예를 들어, 하나 이상의 필드 또는 한 세트의 비트를 통해) 명시적으로 지정하거나, 표시하거나(예를 들어, 암시적으로 표시하거나), 그렇지 않으면 가질 수 있다.

[0023] 제 1 소스 패킹된 데이터 피연산자(312) 내의 데이터 요소(E0 내지 EN)의 개수는 변할 수 있다. 보통, 데이터 요소(E0-EN)의 개수는 단일의 데이터 요소(예를 들면, E0)의 비트의 크기로 나누어진 제 1 소스 패킹된 데이터 피연산자의 비트의 크기와 동일할 수 있다. 다양한 실시예에서, 제 1 소스 패킹된 데이터 피연산자의 크기 또는 비트 폭은, 본 발명의 범위가 이렇게 제한되지 않지만, 64-비트, 128-비트, 256-비트, 512-비트, 또는 1024-비트일 수 있다. 다양한 실시예에서, 각 데이터 요소(예를 들면, E0)의 크기 또는 비트 폭은, 본 발명의 범위가 이렇게 제한되지 않지만, 8-비트, 16-비트, 32-비트, 또는 64-비트일 수 있다. 몇 가지 대표적인 예로서, 그저 몇 가지 예를 들자면, 128-비트 패킹된 8-비트 바이트 포맷은 열여섯 개의 8-비트 바이트 데이터 요소를 포함할 수 있고, 128-비트 패킹된 16-비트 데이터 요소 포맷은 여덟 개의 16-비트 데이터 요소를 포함할 수 있고, 256-비트 패킹된 바이트 포맷은 서른두 개의 8-트 바이트 데이터 요소를 포함할 수 있고, 256-비트 패킹된 16-비트 데이터 요소 포맷은 열여섯 개의 16-비트 데이터 요소를 포함할 수 있으며, 256-비트 패킹된 32-비트 데이터 요소 포맷은 여덟 개의 32-비트 데이터 요소를 포함할 수 있다. 다양한 실시예에서, 데이터 요소 선택 및 통합 연산을 이용하여 가장 큰 효율성을 강화하기 위하여, 제 1 소스 패킹된 데이터 피연산자에는 적어도 여덟, 적어도 열여섯, 또는 열여섯 개보다 많은 (예를 들어, 서른두 개, 육십사 개 등)의 데이터 요소(E0-EN)가 있을 수 있다.

[0024] 일부 실시예에서, 명령어는 제 1 소스 패킹된 데이터 피연산자의 데이터 요소의 크기 또는 비트 폭을 표시하는 한 세트의 하나 이상의 비트 및/또는 필드를 가질 수 있다. 예를 들면, 2-비트 필드는 데이터 요소의 네 개의 상이한 크기를 표시하는 네 개의 상이한 값(예를 들어, 8-비트에는 00, 16-비트에는 01, 32-비트에는 10, 그리고 64-비트에는 11)을 가질 수 있다. 다른 실시예에서, 제 1 소스 패킹된 데이터 피연산자의 데이터 요소의 크기 또는 비트 폭은 선택적으로 레지스터(예를 들어, 모드 레지스터)에서 지정되거나 표시될 수 있다. 예를 들면, 모드 레지스터 내의 제 1 값은 패킹된 데이터 레지스터의 데이터 요소가 8-비트 바이트 요소로 해석될 것이라는 것을 표시할 수 있는 반면, 모드 레지스터 내의 제 2 값은 패킹된 데이터 레지스터의 데이터 요소가 16-비트 데이터 요소로 해석될 것이라는 것을 표시할 수 있다.

[0025] 명령어는 또한 대응하는 복수의 마스크 요소(M0 내지 MN)를 갖는 제 2 소스 피연산자(304)를 (예를 들어, 하나 이상의 필드 또는 한 세트의 비트를 통해) 명시적으로 지정하거나, 표시하거나(예를 들어, 암시적으로 표시하거나), 그렇지 않으면 가질 수 있다(예를 들어, 이미디어트로서 가질 수 있다). 제 2 소스 피연산자는 제 1 소스 패킹된 데이터 피연산자의 각각의 대응하는 상이한 데이터 요소(E) 마다 상이한 마스크 요소(M)를 가질 수 있다. 제 2 소스 피연산자 내의 각 마스크 요소(M)는 (예를 들어, 피연산자 내의 동일한 비트 위치 및/또는 동일한 상대 위치에서) 제 1 소스 패킹된 데이터 피연산자 내의 상이한 데이터 요소(E)에 대응할 수 있다. 예를 들면, E_0는 M_0에 대응할 수 있고, E_1는 M_1에 대응할 수 있는 등 그렇게 대응할 수 있다. 이것이 필수 사항은 아니지만, 대응하는 데이터 요소 및 마스크 요소가 피연산자 내의 대응하는 상대 위치에 있으면 종종 편리하며, 대응관계에 대한 다른 관례가 또한 선택적으로 사용될 수 있다.

[0026] 일부 실시예에서, 각 마스크 요소는 단일의 마스크 비트일 수 있다. 이와 달리, 각 마스크 요소에 대해 둘 이상

의 비트가 선택적으로 사용될 수 있다. 예를 들면, 각 마스크 요소는 제 1 소스 패킹된 데이터 피연산자 내의 각각의 대응하는 데이터 요소와 동일한 비트 수(예를 들면, 8-비트, 16-비트, 32-비트 등)를 가질 수 있다. 사실상 마스크 비트로서 인식되지만 데이터 요소에는 포함되는 단일 비트(예를 들어, 데이터 요소의 최상위 비트 또는 최하위 비트)를 포함하는 특정 구현을 원할 때는 마스크 요소에 대해 임의의 비트 수가 사용될 수도 있다. 일부 실시예에서, 제 2 소스 피연산자는 비트 마스크 피연산자를 나타낼 수 있으며 마스크 비트를 마스크 요소로서 가질 수 있다. 그와 같은 비트 마스크 피연산자는 상이한 방법으로 상이한 실시예에서 제공될 수 있다. 일부 실시예에서, 제 2 소스 비트 마스크 피연산자는 명령어의 이미디어트를 나타낼 수 있다. 다른 실시예에서, 제 2 소스 비트 마스크 피연산자는 패킹된 데이터 연산 마스크 레지스터(예를 들면, 레지스터(118 및/또는 918) 중 하나)에 저장될 수 있다. 또 다른 실시예에서, 제 2 소스 비트 마스크 피연산자는 범용 레지스터(예를 들어, 범용 레지스터(119) 중 하나)에 저장될 수 있다. 대안으로, 비트 마스크 피연산자 대신에, 제 2 소스 피연산자는 패킹된 데이터 요소 마스크 피연산자를 나타낼 수 있으며 데이터 요소를 마스크 요소로서 가질 수 있다. 예를 들면, 제 2 소스 패킹된 데이터 요소 마스크 피연산자는 패킹된 데이터 레지스터(예를 들어, 패킹된 데이터 레지스터(110 및/또는 810) 중 하나)에 저장될 수 있다.

[0027] 각 마스크 요소는 제 1 소스 패킹된 데이터 피연산자로부터 대응하는 데이터 요소가 결과 패킹된 데이터 피연산자에서 출현하도록 선택되거나 구성되는지를 조건부로 제어하거나 마스킹하도록 작용할 수 있다. 마스킹은 각각의 상이한 데이터 요소가 별도로 및/또는 다른 것과 독립적으로 마스킹할 수 있거나 또는 마스킹하지 않을 수 있도록 데이터 요소별로 세분화될 수 있다. 각각의 마스킹하지 않는 마스크 요소는 결과 패킹된 데이터 피연산자(314)의 통합된 또는 그룹화된 데이터 요소 세트 내에 포함시키기 위해 대응하는 데이터 요소를 선택하는 제 1 값(예를 들어, 하나의 가능한 관례에 따라 일(1)이라는 값)을 가질 수 있는 반면, 각각의 마스킹하는 마스크 요소는 결과 패킹된 데이터 피연산자의 통합된 또는 그룹화된 데이터 요소 세트 내에 포함된 것으로부터 대응하는 데이터 요소를 제외하는 상이한 제 2 값(예를 들어, 하나의 관례에 따라 영(0)이라는 값)을 가질 수 있다. 도면에서는 이러한 관례를 채택한다. 통합된 것에서 대응하는 데이터 요소를 선택하거나 제외하는 값에 대해 다양한 다른 관례가 또한 가능하다.

[0028] 결과 패킹된 데이터 피연산자(314)는 (예를 들면, 실행 유닛(308)에 의해) 생성되며 데이터 요소 선택 및 통합 명령어/연산에 응답하여 목적지 저장 위치에 저장될 수 있다. 목적지 저장 위치는 명령어에 의해 지정되거나 그렇지 않으면 명령어에 의해 표시될 수 있다. 다양한 실시예에서, 목적지 저장 위치는 패킹된 데이터 레지스터, 메모리 위치, 또는 다른 저장 위치일 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자(314)는 제 2 소스 피연산자(304)의 마스킹하지 않는 마스크 요소에 대응하는, 그 결과 패킹된 데이터 피연산자(314)의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자(312)의 모든 데이터 요소를 포함할 수 있다. 도면에서 채택된 관례에 따르면, 본 발명의 범위가 이렇게 제한되는 것은 아니지만, 마스킹하는 마스크 요소는 영(0)이라는 값을 갖는 반면, 마스킹하지 않는 마스크 요소는 일(1)이라는 값을 갖는다. 예시된 실시예에서, 데이터 요소(E_1, E_2, E_3, 및 E_5)는 마스킹하지 않는 마스크 요소(M_1, M_2, M_3, 및 M_5)에 대응하는 반면, 데이터 요소(E_0, E_4, 및 E_N)는 마스킹하는 마스크 요소(M_0, M_4, 및 M_N)에 대응한다. 도시된 바와 같이, 일부 실시예에서, 마스킹하지 않는 또는 선택된 데이터 요소(예를 들면, E_1, E_2, E_3, 및 E_5)는 결과 패킹된 데이터 피연산자의 최하위 부분에서 한 세트의 통합된 선택된 또는 마스킹하지 않는 데이터 요소로서 함께 통합될 수 있으며, 이들이 제 1 소스 패킹된 데이터 피연산자에서 출현하거나 배열되었던 것과 동일한 순서로 출현하거나 배열될 수 있다. 대안으로, 마스킹하지 않는 또는 선택된 데이터 요소는 선택적으로 결과 패킹된 데이터 피연산자의 최상위 부분에서 함께 통합될 수 있으며, 이들이 제 1 소스 패킹된 데이터 피연산자에서 출현하거나 배열되었던 것과 동일한 순서로 출현하거나 배열될 수 있다. 마스킹하지 않는 또는 선택된 데이터 요소를 라우팅, 이동, 재배열, 이동, 재배열, 복사, 또는 그렇지 않으면 저장을 통해 적절한 위치에 포함시키도록 하는 다양한 방법이 예상된다.

[0029] 결과 패킹된 데이터 피연산자에서, 함께 통합된 마스킹하지 않는 데이터 요소(예를 들면, E_1, E_2, E_3, 및 E_5)는 마스킹하는 데이터 요소(예를 들면, E_0, E_4, 및 E_N)를 모두 제외시킬 것이다. 즉, 통합된 마스킹하지 않는 데이터 요소는 이들 사이에 어떠한 마스킹하는 데이터 요소도 배치되지 않게 할 수 있다. 예를 들면, 제 1 소스 피연산자에서, E_4는 E_3 와 E_5, 사이에 배치되지만, 결과의 패킹된 데이터 피연산자에서, E_5 및 E_3는 이들 사이에 E가 배치되지 않고 서로 인접하게 함께 통합된다. 따라서, 이들 사이에 마스킹하는 데이터 요소가 분산 배치되어 있는 마스킹하지 않는 데이터 요소의 잠재적으로 인접하지 않은 서브세트는 결과 피연산자의 최하위 또는 최상위 부분에서 함께 통합 또는 압축되어 저장될 수 있다. 일부 실시예에서, 마스킹하는 데이터 요소(예를 들면, E_0, E_4, 및 E_N)는 그저 폐기되거나 삭제될 수 있으며 목적지로 전달될 필요가 없을 수 있다. 그러한 경우, 그리

고 통합되는 마스킹하지 않는 데이터 요소가 결과 피연산자의 최하위 부분에 저장될 때, 전체 개수의 마스킹하지 않는 데이터 요소를 저장하는데 필요한 비트보다 중요한 비트는 선택적으로 미리 정해진 값을 저장할 수 있다. 예를 들면, 도면에 도시된 바와 같이, 이러한 비트 또는 데이터 요소는 선택적으로 결과 패킹된 데이터 피연산자에서 제로로 만들어질 수 있다. 다른 옵션으로서, 이들 비트에 제로를 저장하는 대신에, 기존의 비트 또는 데이터가 변경되지 않은 채로 남아 있을 수도 있다. 이것은 메모리 대역폭 액세스를 줄이는데 도움이 될 수 있다. 프로세서는 통합되는 데이터 요소의 개수를 알 수 있으며 그럼으로써 결과로 생긴 통합된 요소의 범위를 알 수 있다. 또 다른 옵션으로서, 다른 실시예에서, 마스킹하는 데이터 요소(예를 들면, E_0, E_4, 및 E_N)는 함께 통합된 마스킹하지 않는 데이터 요소를 저장하는데 사용되지 않는 결과 패킹된 데이터 피연산자의 다른 부분으로 선택적으로 라우팅되거나, 이동되거나, 복사되거나, 그렇지 않으면 저장될 수 있다.

[0030] 일부 실시예에서, 결과에서 함께 통합된 마스킹하지 않는 데이터 요소의 개수를 표시하는 값은 명령어에 응답하여 (예를 들면, 범용 또는 다른 레지스터에) 선택적으로 저장될 수 있다. 예를 들면, 도 4에서, 여덟 개의 마스킹하지 않는 데이터 요소가 결과에서 통합되었음을 표시하기 위해 8을 표시하는 값이 선택적으로 저장될 수 있다. 이렇게 하면 결과에서 통합된 마스킹하지 않는 데이터 요소의 개수만이 이 개수를 별도로 계산할 필요 없이 액세스되게 하는데 도움이 될 수 있는데, 이것은 옵션이지 필수 사항은 아니다. 이러한 양태는 또한 본 명세서 (예를 들면, 도 1 내지 도 5)에서 개시된 다른 실시예와 함께 사용될 수 있다.

[0031] 도 4는 비트 마스크 및 통합 명령어를 이용한 바이트 선택의 실시예에 대응하여 수행될 수 있는 비트 마스크 및 통합 연산(430)을 이용한 바이트 선택의 특정한 예시적인 실시예를 예시하는 블록도이다. 도 4의 예시적인 연산은 도 3의 더 일반화된 연산과 어느 정도 유사한 점이 있다. 설명을 불명료하게 하는 것을 피하기 위해, 도 3의 더 일반화된 연산과 관련하여 선택적으로 유사하거나 공통적인 특성 및 세부사항을 모두 되풀이 하여 설명하지 않고, 도 4의 예시적인 연산에 관한 상이한 및/또는 부수적인 특성이 주로 설명될 것이다. 그러나, 도 3의 더 일반화된 연산에 관해 이전에 설명된 특성 및 세부 사항은, 달리 언급이 없거나 명명백백하다면(예를 들면, 그 특성 및 세부 사항이 비트 마스크가 아닌 데이터 요소 마스크에 관한 것이라면), 선택적으로 도 4의 예시적인 연산에도 적용할 수 있다는 것을 인식하여야 한다.

[0032] 이러한 예시적인 실시예에서, 명령어는 달리 지적하지 않는 한, 열여섯 개의 8-비트 바이트 데이터 요소(B_0 내지 B_{15})를 갖는 제 1의 128-비트 소스 패킹된 바이트 피연산자(412)를 지정하거나, 지시하거나(indicate), 그렇지 않으면, 포함한다. 이것은 그저 하나의 예일뿐이다. 다른 실시예는 더 좁은(예를 들면, 64-비트) 또는 더 넓은(예를 들면, 256-비트, 512-비트, 1024-비트) 중 어느 하나를 가질 수 있으며, 다른 크기의 데이터 요소(예를 들면, 16-비트 데이터 요소, 32-비트 데이터 요소, 64-비트 데이터 요소 등)을 가질 수 있다.

[0033] 도 4의 예시적인 실시예에서, 명령어는 달리 지적하지 않는 한, 열여섯 개의 마스크 비트를 갖는 제 2의 16-비트 소스 비트 마스크 피연산자(404)를 지정하거나, 그렇지 않으면 이를 갖는다. 이들 각 비트는 피연산자 내의 동일한 상대적인 위치(예를 들면, 도면에서 수직으로 정렬된 데이터 요소)에 있는 제 1 소스 패킹된 바이트 피연산자의 한 바이트에 대응한다. 각각의 마스킹하지 않는 마스크 비트는 제 1 값(예를 들면, 도면에서 채택된 하나의 가능한 관례에 따른 이진수 1)을 가질 수 있는 반면, 각각의 마스킹하는 마스크 비트는 상이한 제 2 값(예를 들면, 도면에서 채택된 관례에 따른 이진수 0)을 가질 수 있다. 구체적으로, 예시된 예에서, 이들 비트의 값은 오른쪽의 최하위 끝에서부터 왼쪽의 최상위 끝까지 [0011101010001101]이다. 이것은 그저 하나의 예의 비트 값 세트일 뿐이다. 이러한 예에 따르면, 바이트(B_0, B_1, B_5, B_7, B_9, B_{10}, B_{11} 및 B_{14})는 마스킹하는 바이트인 반면, 바이트(B_2, B_3, B_4, B_6, B_8, B_{12}, B_{13}, 및 B_{15})는 마스킹하지 않는 바이트이다. 다른 실시예는 더 좁은 또는 더 넓은 비트 마스크 피연산자 중 어느 하나(예를 들면, 다른 소스 패킹된 데이터 피연산자 내의 각 데이터 요소에 대해 하나의 비트)를 사용할 수 있다.

[0034] 데이터 요소 마스크(예를 들면, 바이트 마스크)와 비교하여, 비트 마스크의 하나의 잠재적인 장점은 마스크를 이동 및/또는 활용하는 것과 연관된 자원 활용/소비를 감소하는데 도움이 될 수 있다는 것이다. 대표적으로, 잠재적으로 감소될 수 있는 가능한 유형의 자원 활용/소비의 예는, 이것으로 제한되는 것은 아니지만, 특정 구현에 따라, 메모리 버스, 중앙 처리 유닛(central processing unit, CPU), 그래픽스 프로세싱 유닛(graphics processing unit, GPU), 시스템 온 칩(system on a chip, SOC) 인터커넥트, 네트워크 인터페이스, 범용 입력 및/또는 출력(input and/or input) 버스, 캐시 포트 대역폭, 프로세서 재정렬 버퍼, 로드 버퍼, 메모리 조합 버퍼, 캐시 라인, 물리 및 가상 메모리 등등을 포함한다.

[0035] 일부 실시예에서, 제 2의 16-비트 소스 비트 마스크 피연산자(404)는 명령어의 이미디어트를 나타낼 수 있다.

예를 들면, 열여섯 개의 마스크 비트는 선택적으로 16-비트, 20-비트, 32-비트, 또는 다른 크기의 이미디어트에 저장될 수 있다. 일부 실시예에서, 이미디어트의 마스크 비트는 (예를 들면, 아래에서 자세히 논의되는 바와 같은 패킷 프로토콜 디코더에 의해, 런타임 컴파일러에 의해, 기타 등등) 런타임에서 결정될 수 있다. 다른 실시예에서, 제 2의 16-비트 소스 마스크 피연산자는 패킹된 데이터 연산 마스크 레지스터(예를 들면, 레지스터(118 및/또는 918) 중 하나)에 선택적으로 저장될 수 있다. 일 양태에서, 패킹된 데이터 연산 마스크 레지스터는 주로 예측에 전용될 수 있다. 일 양태에서, 프로세서의 명령어 집합 중의 다른 명령어는 패킹된 데이터 연산 마스크 레지스터를 패킹된 데이터 연산(예를 들면, 패킹된 곱셈 연산, 패킹된 덧셈 연산, 패킹된 회전 연산, 또는 패킹된 비교 연산 등)을 예측하는 예측 피연산자로서 표시할 수 있다. 또 다른 실시예에서, 제 2의 16비트 소스 비트 마스크 피연산자는 선택적으로 범용 레지스터(예를 들면, 레지스터(119) 중 하나)에 저장될 수 있다.

[0036] 결과 패킹된 데이터 피연산자(414)는 (예를 들면, 실행 유닛(408)에 의해) 생성될 수 있으며, 명령어/연산에 응답하여 목적지 저장 위치에 저장될 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자는 결과 패킹된 데이터 피연산자(414)의 일부에서 함께 통합된, 제 2 소스 비트 마스크 피연산자(404)의 마스킹하지 않는 마스크 비트에 대응하는, 제 1 소스 패킹된 바이트 피연산자(412)의 모든 데이터 요소를 포함할 수 있다. 도시된 바와 같이, 마스킹하지 않는 바이트(예를 들면, B_2, B_3, B_4, B_6, B_8, B_{12}, B_{13}, 및 B_{15})는 결과 패킹된 데이터 피연산자의 최하위 부분(예를 들면, 여덟 개의 최하위 바이트)에서 함께 통합될 수 있다. 마스킹하지 않는 모든 바이트를 저장할 필요 없는 결과 패킹된 데이터 피연산자의 최상위 바이트(이 경우에는 여덟 개의 최상위 바이트)는 선택적으로 미리 정해진 값을 가질 수 있다(모두 제로로 클리어될 수 있다). 다른 옵션으로서, 이들 비트에 제로를 저장하는 대신에, 기존의 데이터 또는 데이터가 변경되지 않은 채로 남아 있을 수도 있다. 이것은 메모리 대역폭 액세스를 감소하는데 도움이 될 수 있다. 프로세서는 통합되는 데이터 요소의 개수를 알 수 있으며 그럼으로써 결과로 생긴 통합된 요소의 범위를 알 수 있다. 대안으로, 마스킹하지 않는 바이트는 결과 패킹된 데이터 피연산자의 최상위 부분 또는 다른 부분에 함께 선택적으로 통합될 수 있다. 예시된 실시예에서, 마스킹하는 바이트(B_0, B_1, B_5, B_7, B_9, B_{10}, B_{12}, 및 B_{14})는 결과 패킹된 데이터 피연산자에서 제외될 수 있다. 대안으로, 다른 실시예에서, 마스킹하는 바이트는 선택적으로 함께 통합될 수 있으며 마스킹하지 않는 바이트를 저장하는데 사용되지 않는 결과 패킹된 데이터 피연산자의 다른 부분에 저장될 수 있다.

[0037] 도 5는 바이트 마스크 및 통합 명령어를 이용한 바이트 선택의 실시예에 대응하여 수행될 수 있는 바이트 마스크 및 통합 연산(530)을 이용한 바이트 선택의 특정한 예시적인 실시예를 예시하는 블록도이다. 도 5의 예시적인 연산은 도 3의 더 일반화된 연산과 어느 정도 유사한 점이 있다. 설명을 불명료하게 하는 것을 피하기 위해, 도 3의 더 일반화된 연산과 관련하여 선택적으로 유사하거나 공통적인 특성 및 세부사항을 모두 되풀이하여 설명하지 않고, 도 5의 예시적인 연산에 관한 상이한 및/또는 부수적인 특성이 주로 설명될 것이다. 그러나 도 3의 더 일반화된 연산에 관해 이전에 설명된 특성 및 세부 사항은, 달리 언급이 없거나 명명백백하다면(예를 들어, 그 특성 및 세부 사항이 비트 마스크가 아닌 데이터 요소 마스크에 관한 것이라면), 선택적으로 도 5의 예시적인 연산에도 적용할 수 있다는 것을 인식하여야 한다.

[0038] 이러한 예시적인 실시예에서, 명령어는 달리 지적하지 않는 한, 열여섯 개의 8-비트 바이트 데이터 요소를 갖는 제 1의 128-비트 소스 패킹된 바이트 피연산자(512)를 지정하거나, 그렇지 않으면, 이를 갖는다. 다른 실시예는 더 좁은(예를 들면, 64-비트) 또는 더 넓은(예를 들면, 256-비트, 512-비트, 1024-비트) 중 어느 하나를 가질 수 있으며, 이전에 기술된 바와 같이, 다른 크기의 데이터 요소(예를 들면, 16-비트 데이터 요소, 32-비트 데이터 요소, 64-비트 데이터 요소 등)을 가질 수 있다.

[0039] 이러한 예시적인 실시예에서, 명령어는 달리 지적하지 않는 한, 열여섯 개의 마스크 바이트를 갖는 제 2의 128-비트 소스 바이트 마스크 피연산자(504)를 명시하거나, 그렇지 않으면 이를 갖는다. 이들 마스크 바이트 각각은 피연산자 내의 동일한 상대적인 위치(예를 들면, 도면에서 수직으로 정렬로 도시된 동일한 비트 위치)에 있는 제 1 소스 패킹된 바이트 피연산자의 한 바이트에 대응한다. 각각의 마스킹하지 않는 마스크 바이트는 제 1 값(예를 들면, 도면에서 채택된 하나의 가능한 관례에 따르면 모두 1 [11111111])을 가질 수 있는 반면, 각각의 마스킹하는 마스크 바이트는 상이한 제 2 값(예를 들면, 도면에서 채택된 관례에 따르면 모두 0 [00000000])을 가질 수 있다. 예시된 예에서, 제 1 소스 패킹된 바이트 피연산자의 비트 [7:0]에 있는 최하위 바이트 및 비트 [127:120]에 있는 최상위 바이트는 마스킹하는 바이트인 반면, 비트[15:8] 및 [23:16]에 있는 바이트는 마스킹하지 않는 바이트이다. 일부 실시예에서, 제 2의 128-비트 소스 바이트 마스크 피연산자(504)는 패킹된 데이터 레지스터(예를 들면, 레지스터(110 및/또는 810) 중 하나)에 선택적으로 저장될 수 있다. 대안으로, 메모리 위치 또는 다른 저장 위치가 선택적으로 사용될 수 있다.

[0040] 결과 패킹된 데이터 피연산자(514)는 (예를 들면, 실행 유닛(508)에 의해) 생성될 수 있으며, 명령어/연산에 응답하여 목적지 저장 위치에 저장될 수 있다. 일부 실시예에서, 결과 패킹된 데이터 피연산자는 결과 패킹된 데이터 피연산자(514)의 일부에서 함께 통합된, 제 2 소스 바이트 마스크 피연산자(504)의 마스킹하지 않는 마스크 바이트에 대응하는, 제 1 소스 패킹된 바이트 피연산자(512)의 모든 데이터 요소를 포함할 수 있다. 도시된 바와 같이, 비트 [15:8] 및 [23:16]에 있는 마스킹하지 않는 바이트는 결과 패킹된 데이터 피연산자의 최하위 부분에서 함께 통합될 수 있다. 마스킹하지 않는 바이트를 모두 저장할 필요 없는 결과 패킹된 데이터 피연산자의 최상위 바이트는 선택적으로 미리 정해진 값을 가질 수 있다(예를 들면, 모두 제로로 클리어될 수 있다). 대안으로, 마스킹하지 않는 바이트는 선택적으로 결과 패킹된 데이터 피연산자의 최상위 부분 또는 다른 부분에 함께 통합될 수 있다. 예시된 실시예에서, 결과 패킹된 데이터 피연산자는 모든 마스킹하는 바이트를 배제 또는 제외시킬 수 있다. 대안으로, 마스킹하는 바이트는 선택적으로 함께 통합될 수 있으며 마스킹하지 않는 바이트를 저장하는데 사용되지 않는 결과 패킹된 데이터 피연산자의 다른 부분에 저장될 수 있다.

[0041] 본 명세서에서 개시된 데이터 요소 선택 및 통합 명령어는 범용의 명령어이며 다양한 여러 목적을 위해 사용될 수 있다. 이들 명령어는 단독으로 또는 다른 명령어와 조합하여, 특정 애플리케이션, 알고리즘, 또는 코드에 유용한 다양한 여러 방식으로 데이터 요소를 선택하고 통합하는데 사용될 수 있다. 이들 명령어의 하나의 가능한 용도는 네트워크 패킷의 데이터 요소를 선택하고 통합하는 것이다. 예를 들면, 네트워크 패킷의 데이터 플로우의 소스와 목적지 어드레스 및 소스와 목적지 포트를 정의하는 4-튜플의 데이터 요소를 나타내는 패킷의 플로우 데이터 요소가 추출될 수 있다. 이러한 플로우 요소의 추출은 이를테면, 예를 들어, 컨텍스트 룩업을 수행하기 위해, 네트워킹할 때 다양한 여러 목적에 유용할 수 있다. 패킷 처리 및/또는 네트워킹과 함께 다양한 다른 용도는 이것으로 제한되는 것은 아니지만, 헤더 체크섬, 부분적 디지털 서명의 계산, ARP 캐시, 패킷 스위칭 또는 다른 포워딩, 필터링, 콘텐츠 기반의 부하 밸런싱, 패킷 콘텐츠의 해싱, 패킷 분류, 및 애플리케이션 지향 네트워킹 중 하나 이상의 목적을 위해 패킷의 여러 부분의 선택 및 통합을 포함한다. 다른 양태에서, 이들 명령어는 또는 암호화 기능(예를 들면, 해시, 체크섬 등)에 입력하기 위한 데이터 요소(예를 들면, 상대적으로 더 많은 엔트로피의 데이터 요소)를 선택하고 통합하기 위해 네트워킹할 때 또는 다른 애플리케이션에서 사용될 수 있다. 이들 명령어의 다른 가능한 용도는 그래픽스 서브샘플링을 위해 데이터 요소를 선택하고 통합하는 것이다. 또 다른 용도는 확장 가능한 마크업 언어(extensible markup language, XML) 처리, 데이터베이스 애플리케이션, 이미지 처리, 및 비교를 가속화하는 것이다. 본 기술에서 통상의 지식을 갖고 본 개시의 이득을 보는 자들에게는 다양한 다른 애플리케이션이 자명할 것이다. 특정의 개념을 추가로 예시하기 위해, 본 발명의 범위가 이렇게 제한되는 것은 아니지만, 네트워크 패킷을 처리할 때 데이터 요소 선택 및 통합 명령어(602)의 상세한 사용 예가 제공될 것이다.

[0042] 도 6은 일부 실시예에서 데이터 요소 선택 및 통합 명령어(602)가 네트워크 패킷의 데이터 요소를 처리하기 위해 어떻게 사용될 수 있는지를 도시하는 블록도이다. 프로세서는 프로토콜 디코더(662)를 포함한다. 프로토콜 디코더는 (예를 들면, 네트워크 인터페이스로부터) 네트워크 패킷을 수신하도록 연결된다. 프로토콜 디코더는 이들 네트워크 패킷의 네트워크 프로토콜을 디코딩하도록 동작한다. 이것은 기존의 방법으로 수행될 수 있다. 일부 실시예에서, 프로토콜 디코더는 (예를 들면, 프로세서 외부의 메모리에 저장된) 소프트웨어 모듈로서 구현될 수 있다. 그러한 프로토콜 디코더 모듈은 다양한 각종 프로토콜을 더 융통성 있게 디코딩하는 데 종종 사용된다. 다른 실시예에서, 프로토콜 디코더는 하드웨어 온-다이(hardware on-die) 및/또는 온-프로세서(on-processor)에서 구현될 수 있다. 예를 들면, 하드웨어 프로토콜 디코더는 상대적으로 더 많이 사용되는 프로토콜 용도로 선택적으로 포함될 수 있다.

[0043] 도 6을 다시 참조하면, 프로토콜 디코더는 제 1 패킷(660-1)을 수신할 수 있다. 프로토콜 디코더는 제 1 패킷의 프로토콜을 엄격하게 디코딩하거나 결정할 수 있다. 프로토콜 디코더의 출력은 모든 관련 계층(예를 들면, TCP/IPV4)에서 프로토콜이 결정되었다는 것이다. 이것을 기초로 하여, 프로토콜 디코더는 마스크를 발생할 수 있다. 예를 들면, 프로토콜 디코더는 (예를 들면, 이미디어트(104)와 유사한) 이미디어트 비트 마스크(604)를 생성하여 디코드 유닛(606)에 제공할 수 있다. 이미디어트 비트 마스크 또는 다른 마스크는 동일한 플로우의 패킷의 관심 있는 그리고/또는 동일한 네트워크 프로토콜(들)을 제 1 패킷으로서 사용하는 특정 데이터 요소를 선택하도록 작용할 수 있고 동작할 수 있다.

[0044] 이후, 동일 플로우 또는 동일 커넥션의 제 2 패킷(660-2)이 프로토콜 디코더에 의해 수신될 수 있다. 프로토콜 디코더, 또는 헤드 추출 컴포넌트 또는 다른 컴포넌트(도시되지 않음)는 제 2 패킷의 헤더의 적어도 일부를 패킹된 데이터 피연산자(612)로서 저장할 수 있다. 패킹된 데이터 피연산자(612)는 제 2 패킷 헤더의 복수의 패킹된 데이터 요소를 가질 수 있다. 피연산자는 앞에서 기술한 바와 같이 한 세트의 패킹된 데이터 레지스터(610)

에 저장될 수 있다.

[0045] 패킹된 데이터 피연산자(612)를 지시하는 데이터 요소 선택 및 통합 명령어(602)는 디코드 유닛(606)에서 수신될 수 있다. 디코드 유닛은 또한 마스크(예를 들면, 이미디어트 비트 마스크(604))를 수신할 수 있다. 디코드 유닛은 앞에서 기술한 바와 같이 명령어를 디코딩할 수 있으며 실행 유닛(608)을 제어하거나 실행 유닛으로 하여금 패킹된 데이터 피연산자(612) 및 마스크를 사용하여 데이터 요소 선택 및 통합 연산을 수행할 수 있게 하고 결과 패킹된 데이터 피연산자(614)를 저장할 수 있다. 결과 패킹된 데이터 피연산자는 제 2 패킷의 헤더의 통합된 선택된/마스킹하지 않는 데이터 요소를 가질 수 있다.

[0046] 제 2 패킷의 헤더의 어느 특정 데이터 요소가 선택되는지는 특정한 구현에 달려 있다. 일부 실시예에서, 플로우 바이트 또는 요소가 선택되고 통합될 수 있다. 예를 들면, 프로토콜 디코딩에 기초하여, 프로토콜 디코더는 플로우 바이트의 위치(예를 들면, TCP/IP 소스/목적지 어드레스 및 소스/목적지 포트 번호)가 제 2 패킷의 헤더 내에 상주한다는 것을 알 수 있다. 프로토콜 디코더는 이들 플로우 바이트가 마스킹되지 않거나 선택되도록 마스크(예를 들면, 각각의 플로우 바이트에 대한 마스킹하지 않는 마스크 요소 및 다른 바이트에 대한 마스킹하는 마스크 요소)를 발생할 수 있다. 이들 플로우 바이트의 추출 및 통합은 다른 연산을 용이하게 해줄 수 있다. 예를 들면, 다른 컴포넌트(664)는 이들 플로우 바이트를 사용하여 RSS 해싱 등을 위한 컨텍스트 룩업 등을 수행할 수 있다. 이렇게 하면 패킷 처리 시 종종 계산 상 비용이 드는 연산이 될 경향이 있는 플로우 추출을 가속하는 데 도움이 될 수 있다. 다른 실시예에서, 프로토콜 바이트 또는 다른 요소가 선택되어 통합될 수 있다. 이렇게 하면 그와 같은 엄격한 프로토콜 디코딩을 거칠 필요 없이 후속 패킷의 프로토콜의 검사를 신속히 수행되게 할 수 있다. 또 다른 실시예에서, 제 2 패킷에서 관심의 다른 바이트 또는 데이터 요소가 선택되어 통합될 수 있다. 예를 들면, 헤더 중 상대적으로 엔트로피한 요소가 추출된 다음 다른 컴포넌트(664), 예를 들면, 암호화 모듈이 상대적으로 엔트로피한 요소에 대해 해시 또는 다른 암호화 연산을 수행할 수 있다.

[0047] 도 7(a)는 데이터 요소 선택 및 통합 명령어(702A)의 제 1 실시예의 블록도이다. 이 명령어는 연산 코드 또는 오피코드(740A)를 포함한다. 오피코드는 수행될 명령어 및/또는 연산(예를 들면, 데이터 요소 선택 및 통합 연산)을 식별하도록 작용 가능한 복수의 비트 또는 하나 이상의 필드를 나타낼 수 있다. 명령어는 또한 제 1 소스 패킹된 데이터 피연산자를 식별하는 제 1 소스 패킹된 데이터 피연산자 지정자(first source packed data operand specifier)(7412A)를 포함한다. 명령어는 결과 패킹된 데이터 피연산자가 저장되는 목적지 저장소 위치를 지정하는 목적지 지정자(744A)를 선택적으로 포함할 수 있다. 예를 들어, 이들 지정자(742A, 744A) 각각은 관련된 피연산자를 위한 레지스터, 메모리 위치, 또는 다른 저장 장소의 어드레스를 명시적으로 지정하는 한 세트의 비트 또는 하나 이상의 필드를 포함할 수 있다. 대안으로, 다른 실시예에서, 지정자 중 하나 이상의 지정자는 명시적으로 지정되는 대신에, 선택적으로 명령어에 암시적일 수 있다(예를 들면, 오피코드에 암시적일 수 있다. 예를 들면, 명령어는 명시적으로 지정될 필요가 없는 소스 및/또는 목적지로서 암시적인 고정 레지스터를 가질 수 있다. 다른 예로서, 일부 실시예에서, 목적지 지정자(744A)의 대신에, 선택적으로 제 1 소스 패킹된 데이터 피연산자를 위해 사용되는 동일 레지스터 또는 다른 저장 위치가 암시적으로 목적지 저장 위치로서 사용될 수 있다. 예로서, 명령어는 별개의 소스 및 목적지 지정자(742A, 744A)를 처음에는 소스 피연산자에 사용되고 나중에는 결과 피연산자를 저장하는데 사용되는 레지스터 또는 다른 저장 위치를 식별하는 단일의 소스/목적지 지정자로 대체할 수 있다. 이 실시예에서, 명령어는 또한 소스 비트 마스크 피연산자를 제공하는 이미디어트(704)를 가질 수도 있다. 일부 실시예에서, 명령어는 또한 제 1 소스 패킹된 데이터 피연산자의 데이터 요소의 크기를 지정하거나 최소한 표시하는 옵션의 데이터 요소 크기 지정자(746A)를 가질 수도 있다. 데이터 요소 크기 지정자는 하나 이상의 비트 또는 필드를 포함할 수 있다.

[0048] 도 7(b)는 데이터 요소 선택 및 통합 명령어(702B)의 제 2 실시예의 블록도이다. 이 명령어는 연산 코드 또는 오피 코드(740B), 제 1 소스 패킹된 데이터 피연산자 지정자(742B), 옵션의 목적지 지정자(744B), 및 옵션의 데이터 요소 크기 지정자(746B)를 포함한다. 이들은 각기 명령어(702A)의 대응하는 명칭을 가진 컴포넌트와 동일하거나 유사할 수 있으며, 동일한 변형 및 대안을 가질 수 있다. 이 실시예에서, 이미디어트(704)의 대신에, 명령어(702B)는 패킹된 데이터 연산 마스크 지정자(748)를 선택적으로 포함한다. 패킹된 데이터 연산 마스크 지정자는 패킹된 데이터 연산 마스크 레지스터를 지정할 수 있다. 대안으로, 패킹된 데이터 연산 마스크 레지스터는 암시적으로 표시될 수 있다.

[0049] 도 7(c)는 데이터 요소 선택 및 통합 명령어(702C)의 제 3 실시예의 블록도이다. 이 명령어는 연산 코드 또는 오피코드(740C), 제 1 소스 패킹된 데이터 피연산자 지정자(742C), 옵션의 목적지 지정자(744C), 및 옵션의 데이터 요소 크기 지정자(746C)를 포함한다. 이들은 각기 명령어(702A)의 대응하는 명칭을 가진 컴포넌트와 동일하거나 유사할 수 있으며, 동일한 변형 및 대안을 가질 수 있다. 이 실시예에서, 이미디어트(704) 및/또는 패킹

된 데이터 연산 마스크 지정자(748)의 대신에, 명령어(702C)는 제 2 소스 패킹된 데이터 피연산자 지정자(750)를 선택적으로 포함한다. 제 2 소스 패킹된 데이터 피연산자 지정자(750)는 패킹된 데이터 요소 마스크가 저장되는 제 2 소스 패킹된 데이터 레지스터 또는 다른 패킹된 데이터 피연산자 저장 위치를 명시적으로 지정할 수 있다. 대안으로, 패킹된 데이터 요소 마스크에 대해 암시적 저장 위치가 선택적으로 사용될 수 있다.

[0050] 이러한 것들은 적합한 명령어의 그저 몇 가지 예시적인 예의 실시예일뿐이라는 것을 인식하여야 한다. 대안의 실시예는 지정자의 서브세트를 포함할 수 있고, 부가적인 지정자 또는 필드를 추가할 수 있다. 또한, 필드의 예시된 순서/배열은 필수 사항은 아니며, 오히려 필드는 다양하게 배열될 수 있다. 더욱이, 필드는 인접한 비트 시퀀스를 포함할 필요가 없고, 오히려 인접하지 않거나 분리된 비트 등을 포함할 수 있다. 일부 필드는 어쩌면 겹칠 수 있다. 일부 실시예에서, 일부 실시예에서, 본 발명의 범위가 이렇게 제한되는 것은 아니지만, 명령어 포맷은 본 명세서 어디에선가 기술된 바와 같은 명령어 포맷(예를 들면, VEX 또는 EVEX 인코딩 또는 명령어 포맷)을 가질 수 있다.

[0051] 도 8은 적합한 한 세트의 패킹된 데이터 레지스터(810)의 예시적인 실시예의 블록도이다. 패킹된 데이터 레지스터는 ZMM0 내지 ZMM31로 표시된 32개의 512-비트 패킹된 데이터 레지스터를 포함한다. 예시된 실시예에서, 이 또한 필수 사항은 아니지만, 하위의 열여섯 레지스터, 즉, ZMM0-ZMM15의 하위 256-비트는 YMM0-YMM15으로 표시된 각 256-비트 패킹된 데이터 레지스터상에서 엘리어싱(alias)되거나 또는 덮인다. 마찬가지로, 필수 사항은 아니지만, 예시된 실시예에서, 레지스터(YMM0-YMM15)의 하위 128-비트는 XMM0-XMM15으로 표시된 각 128-비트 패킹된 데이터 레지스터상에서 엘리어싱되거나 덮인다. 512-비트 레지스터(ZMM0 내지 ZMM31)는 512-비트 패킹된 데이터, 256-비트 패킹된 데이터, 또는 128-비트 패킹된 데이터를 보유하도록 동작할 수 있다. 256-비트 레지스터(YMM0-YMM15)는 256-비트 패킹된 데이터 또는 128-비트 패킹된 데이터를 보유하도록 동작할 수 있다. 128-비트 레지스터(XMM0-XMM15)는 128-비트 패킹된 데이터를 보유하도록 동작할 수 있다. 일부 실시예에서, 레지스터 각각은 패킹된 부동 소수점 데이터 또는 패킹된 정수 데이터 중 어느 하나를 저장하는데 사용될 수 있다. 적어도 8-비트 바이트 데이터, 16-비트 워드 데이터, 32-비트 2배 단어(doubleword), 32-비트 단정도 부동 소수점 데이터(single-precision floating point data), 64-비트 쿼드워드(quadword), 및 64-비트 배정도 부동 소수점 데이터(double-precision floating point data)를 비롯한 여러 데이터 요소 크기가 지원된다. 대안의 실시예에서, 상이한 개수의 레지스터 및/또는 상이한 크기의 레지스터가 사용될 수 있다. 또 다른 실시예에서, 레지스터는 더 큰 레지스터를 더 작은 레지스터에 엘리어싱하는 것을 사용하거나 사용하지 않을 수도 있고 그리고/또는 부동 소수점 데이터를 저장하는데 사용하거나 사용하지 않을 수도 있다.

[0052] 도 9는 적합한 한 세트의 패킹된 데이터 연산 마스크 레지스터(918)의 예시적인 실시예의 블록도이다. 예시된 실시예에서, 세트는 k_0 내지 k_7로 표시된 여덟 개의 레지스터를 포함한다. 대안의 실시예는 여덟 개보다 적은 수(예를 들면, 2, 4, 6개 등)의 레지스터 또는 여덟 개보다 많은 수(예를 들면, 16, 32개 등) 중 어느 하나를 포함할 수 있다. 이들 레지스터 각각은 패킹된 데이터 연산 마스크를 저장하는데 사용될 수 있다. 예시된 실시예에서, 각 레지스터는 64-비트이다. 대안의 실시예에서, 레지스터의 폭은 64-비트보다 넓거나(예를 들면, 80-비트, 128-비트 등), 64-비트보다 좁거나(예를 들면, 8-비트, 16-비트, 32-비트 등) 어느 하나일 수 있다. 레지스터는 널리 공지된 기술을 사용하여 상이한 방식으로 구현될 수 있으며, 임의의 공지된 특정한 형태의 회로로 제한되지 않는다. 적합한 레지스터의 예는 이것으로 제한되는 것은 아니지만, 전용의 물리 레지스터, 레지스터 리네이밍을 이용하는 동적으로 할당된 물리 레지스터, 및 이들의 조합을 포함한다.

[0053] 일부 실시예에서, 패킹된 데이터 연산 마스크 레지스터(918)는 주로 예측을 위해 전용되는 (예를 들면, 데이터 요소를 극 세분화하여 패킹된 데이터 연산을 예측하는) 별개의 한 세트의 전용 아키텍처적 레지스터일 수 있다. 일부 실시예에서, 명령어는 다른 유형의 레지스터(예를 들면, 패킹된 데이터 레지스터, 범용 레지스터 등)를 인코딩하거나 지정하는데 사용되는 것과 상이한 명령어 포맷의 비트 또는 상이한 명령어 포맷의 하나 이상의 필드에서 패킹된 데이터 연산 마스크 레지스터를 인코딩하거나 지정할 수 있다. 예로서, 명령어는 여덟 개의 패킹된 데이터 연산 마스크 레지스터(k_0 내지 k_7) 중 어느 하나를 인코딩하거나 지정하기 위해 세 개의 비트(예를 들면, 3-비트 필드)를 사용할 수 있다. 대안의 실시예에서, 패킹된 데이터 연산 마스크 레지스터가 더 적거나 더 많을 때는 각기 더 적거나 더 많은 비트가 사용될 수 있다. 특정의 일 실시예에서, 패킹된 데이터 연산 마스크 레지스터(k_1 내지 k_7) (k_0 는 아님)만이 마스킹되는 패킹된 데이터 연산을 예측하는 예측 피연산자로서 어드레싱될 수 있다. 이것이 필수 사항은 아니지만, 레지스터(k_0)는 정규의 소스 또는 목적지로서 사용될 수 있으나, (예를 들어, 만일 k_0가 "마스크 없는(no mask)" 인코딩을 수행하는 것으로 지정된다면) 예측 피연산자로서 인코딩되지 않을 수 있다.

[0054] 도 10은 패킹된 데이터 연산 마스크 레지스터(1018)의 예시적인 실시예의 도면으로, 패킹된 데이터 연산 마스크로서 및/또는 마스킹을 위해 사용되는 비트 수가 패킹된 데이터 폭 및 데이터 요소 폭에 좌우된다는 것을 보여주는 도면이다. 필수 사항은 아니지만, 패킹된 데이터 연산 마스크 레지스터의 예시된 예시적인 실시예는 64-비트 폭이다. 패킹된 데이터 폭 및 데이터 요소 폭의 조합에 따라, 모든 64-비트 또는 64-비트의 서브세트만이 마스킹을 위한 패킹된 데이터 연산 마스크로서 사용될 수 있다. 일반적으로, 요소별로 단일의 마스킹 제어 비트가 사용될 때, 마스킹을 위해 사용되는 패킹된 데이터 연산 마스크 레지스터의 비트 수는 패킹된 데이터 요소 폭으로 나눈 패킹된 데이터 폭과 비트에 있어서 동일하다. 예시된 실시예에서, 필수 사항은 아니지만, 레지스터의 최하위 서브세트 또는 부분은 마스킹을 위해 사용된다. 대안의 실시예에서, 최상위 서브세트 또는 일부 다른 서브세트는 선택적으로 사용될 수 있다.

[0055] 예시적인 코어 아키텍처, 프로세서, 및 컴퓨터 아키텍처

[0056] 프로세서 코어는 상이한 방식으로, 상이한 목적을 위해, 그리고 상이한 프로세서에서 구현될 수 있다. 예를 들어, 그러한 코어의 구현은, 1) 범용 컴퓨팅 용도로 의도된 범용 순차 코어; 2) 범용 컴퓨팅 용도로 의도된 고성능의 범용 비순차 코어; 3) 그래픽스 및/또는 과학 (처리량) 컴퓨팅 용도로 주로 의도된 특수 목적 코어를 포함할 수 있다. 상이한 프로세서의 구현은, 1) 범용 컴퓨팅 용도로 의도된 하나 이상의 범용 순차 코어 및/또는 범용 컴퓨팅 용도로 의도된 하나 이상의 범용 비순차 코어를 포함하는 CPU; 및 2) 그래픽스 및/또는 과학 (처리량) 용도로 주로 의도된 하나 이상의 특수 목적 코어를 포함하는 코프로세서를 포함할 수 있다. 그와 같은 상이한 프로세서는 상이한 컴퓨터 시스템 아키텍처에 이르게 되고, 이런 아키텍처는, 1) CPU와 별개의 칩 상의 코프로세서; 2) CPU와 동일한 패키지 내 별개의 다이 상의 코프로세서; 3) CPU와 동일한 다이 상의 코프로세서 (이 경우에, 그러한 코프로세서는 때때로 통합된 그래픽스 및/또는 과학 (처리량) 로직과 같은 특수 목적 로직, 또는 특수 목적 코어라고 지칭됨); 및 4) 기술된 CPU(때때로 애플리케이션 코어(들) 또는 애플리케이션 프로세서(들)라고 지칭됨), 전술한 코프로세서, 및 부가 기능성을 동일한 다이 상에서 포함할 수 있는 칩 상의 시스템을 포함할 수 있다. 예시적인 코어 아키텍처는 아래에서 기술된 다음, 예시적인 프로세서 및 컴퓨터 아키텍처에 관한 설명이 이어진다.

[0057] 예시적인 코어 아키텍처

[0058] 순차 또는 비순차 코어의 블록도

[0059] 도 11(a)는 본 발명의 실시예에 따른 예시적인 순차 파이프라인 및 예시적인 레지스터 리네이밍, 비순차 발행/실행 파이프라인 모두를 예시하는 블록도이다. 도 11(b)는 본 발명의 실시예에 따라 프로세서에 포함될 순차 아키텍처 코어 및 예시적인 레지스터 리네이밍, 비순차 발행/실행 아키텍처 코어의 실시예 모두를 예시하는 블록도이다. 도 11(a) 및 도 11(b)에서 실선 박스는 순차 파이프라인 및 순차 코어를 예시하는 반면, 점선 박스의 선택적으로 부가된 것은 레지스터 리네이밍, 비순차 발행/실행 파이프라인 및 코어를 예시한다. 순차의 양태가 비순차 양태의 서브세트라는 것을 감안하여, 비순차 양태가 기술될 것이다.

[0060] 도 11(a)에서, 프로세서 파이프라인(1100)은 페치 스테이지(1102), 길이 디코드 스테이지(1104), 디코드 스테이지(1106), 할당 스테이지(1108), 리네이밍 스테이지(1110), 스케줄링(디스패치(dispatch) 또는 발행이라고도 알려져 있음) 스테이지(1112), 레지스터 읽기/메모리 읽기 스테이지(1114), 실행 스테이지(1116), 라이트 백(write back)/메모리 쓰기 스테이지(1118), 예외 처리 스테이지(1122), 및 커밋(commit) 스테이지(1124)를 포함한다.

[0061] 도 11(b)는 실행 엔진 유닛(1150)에 연결된 프론트 엔드 유닛(front end unit)(1130) - 둘 다 메모리 유닛(1170)에 연결되어 있음 - 을 포함하는 프로세서 코어(1190)를 도시한다. 코어(1190)는 축소 명령어 집합 컴퓨팅(reduced instruction set computing, RISC) 코어, 복합 명령어 집합 컴퓨팅(complex instruction set computing, CISC) 코어, 매우 긴 명령어 워드(very long instruction word, VLIW) 코어, 또는 복합체 또는 대안적 코어 유형일 수 있다. 또 다른 옵션으로서, 코어(1190)는, 예를 들면, 네트워크 또는 통신 코어, 압축 엔진, 코프로세서 코어, 범용 컴퓨팅 그래픽스 프로세싱 유닛(general purpose computing graphics processing unit, GPGPU) 코어, 또는 그래픽스 코어 등과 같은 특수 목적 코어일 수 있다.

[0062] 프론트 엔드 유닛(1130)은 명령어 캐시 유닛(1134)에 연결된 분기 예측 유닛(1132)을 포함하며, 명령어 캐시 유닛은 명령어 변환 룩어사이드 버퍼(translation lookaside buffer, TLB)(1136)에 연결되고, 명령어 TLB는 명령어 페치 유닛(1138)에 연결되고, 명령어 페치 유닛은 디코드 유닛(1140)에 연결된다. 디코드 유닛(1140)(또는 디코더)은 명령어를 디코딩할 수 있으며, 원래의 명령어로부터 디코딩되거나, 또는 그렇지 않으면 원래의 명령

어를 반영하거나, 또는 원래의 명령어로부터 유도되는, 하나 이상의 마이크로 연산, 마이크로 코드 엔트리 포인트, 마이크로명령어, 다른 명령어 또는 다른 제어 신호를 출력으로서 생성할 수 있다. 디코드 유닛(1140)은 다양하고 상이한 메커니즘을 사용하여 구현될 수 있다. 적절한 메커니즘의 예는, 이것으로 제한되는 것은 아니지만, 룩업 테이블, 하드웨어 구현, 프로그래머블 로직 어레이(programmable logic array, PLA), 마이크로코드 판독 전용 메모리(read only memory, ROM) 등을 포함한다. 일 실시예에서, 코어(1190)는 (예를 들어, 디코드 유닛(1140) 내 또는 그렇지 않으면 프론트 엔드 유닛(1130) 내의) 특정 매크로 명령어에 대한 마이크로코드를 저장하는 마이크로코드 ROM 또는 다른 매체를 포함한다. 디코드 유닛(1140)은 실행 엔진 유닛(1150) 내의 리네임/할당기 유닛(1152)에 연결된다.

[0063] 실행 엔진 유닛(1150)은 리타이어먼트 유닛(retirement unit)(1154) 및 한 세트의 하나 이상의 스케줄러 유닛(들)(1156)에 연결된 리네임/할당기 유닛(1152)을 포함한다. 스케줄러 유닛(들)(1156)은 예약 스테이션, 중앙 명령어 윈도우 등을 포함하는 임의의 수의 상이한 스케줄러를 나타낸다. 스케줄러 유닛(1156)은 물리 레지스터 파일(들) 유닛(들)(1158)에 연결된다. 각 물리 레지스터 파일(들) 유닛들(1158)은 하나 이상의 물리 레지스터 파일을 나타내고, 이들 중 상이한 물리 레지스터 파일은 스칼라 정수, 스칼라 부동 소수점, 패킹된 정수, 패킹된 부동 소수점, 벡터 정수, 벡터 부동 소수점, 상태(예를 들어, 실행될 다음 명령어의 어드레스인 명령어 포인터) 등과 같은 하나 이상의 상이한 데이터 타입을 저장한다. 일 실시예에서, 물리 레지스터 파일(들) 유닛(1158)은 벡터 레지스터 유닛, 쓰기 마스크 레지스터 유닛, 및 스칼라 레지스터 유닛을 포함한다. 이들 레지스터 유닛은 아키텍처적 벡터 레지스터, 벡터 마스크 레지스터, 및 범용 레지스터를 제공할 수 있다. 레지스터 리네이밍 및 비순차 실행이 (예를 들면, 재정렬된 버퍼(들) 및 리타이어먼트 레지스터 파일(들)을 사용하여; 미래 파일(future file)(들), 이력 버퍼(history buffer)(들), 및 리타이어먼트 레지스터 파일(들)을 사용하여; 레지스터 맵 및 레지스터의 풀(pool)을 사용하여, 등등) 구현될 수 있는 다양한 방식을 보여주기 위해, 물리 레지스터 파일(들) 유닛(들)(1158)이 리타이어먼트 유닛(1154)과 겹쳐져 있다. 리타이어먼트 유닛(1154) 및 물리 레지스터 파일(들) 유닛(들)(1158)은 실행 클러스터(들)(1160)에 연결된다. 실행 클러스터(들)(1160)는 한 세트의 하나 이상의 실행 유닛(1162) 및 한 세트의 하나 이상의 메모리 액세스 유닛(1164)을 포함한다. 실행 유닛(1162)은 다양한 종류의 데이터(예를 들어, 스칼라 부동 소수점, 패킹된 정수, 패킹된 부동 소수점, 벡터 정수, 벡터 부동 소수점)에 대해 다양한 연산(예를 들어, 시프트, 더하기, 빼기, 곱하기)을 수행할 수 있다. 일부 실시예는 특정 기능 또는 일련의 기능에 전용되는 복수의 실행 유닛을 포함할 수 있지만, 다른 실시예는 단 하나의 실행 유닛 또는 모두가 모든 기능을 수행하는 복수의 실행 유닛을 포함할 수 있다. 스케줄러 유닛(들)(1156), 물리 레지스터 파일(들) 유닛(들)(1158), 및 실행 클러스터(들)(1160)는 아마도 복수개인 것으로 도시되어 있는데, 그 이유는 특정의 실시예가 특정한 형태의 데이터/연산에 대해 별개의 파이프라인(예를 들어, 각자 자체의 스케줄러 유닛, 물리 레지스터 파일(들) 유닛, 및/또는 실행 클러스터를 갖는 스칼라 정수 파이프라인, 스칼라 부동 소수점/패킹된 정수/패킹된 부동 소수점/벡터 정수/벡터 부동 소수점 파이프라인, 및/또는 메모리 액세스 파이프라인 - 그리고 별개의 메모리 액세스 파이프라인의 경우에는 이 파이프라인의 실행 클러스터만이 메모리 액세스 유닛(들)(1164)을 갖는 특정의 실시예가 구현됨)을 생성하기 때문이다. 별개의 파이프라인이 사용되는 경우에, 이들 파이프라인 중 하나 이상은 비순차적 발행/실행일 수 있고 나머지는 순차적일 수 있다는 것을 또한 이해하여야 한다.

[0064] 메모리 액세스 유닛(1164)의 세트는 메모리 유닛(1170)에 연결되며, 메모리 유닛은 데이터 TLB 유닛(1172)와, 이에 연결된 데이터 캐시 유닛(1174)과, 이에 연결된 레벨 2(level 2, L2) 캐시 유닛(1176)을 포함한다. 예시적인 일 실시예에서, 메모리 액세스 유닛(1164)은 로드 유닛(load unit), 스토어 어드레스 유닛(store address unit), 및 스토어 데이터 유닛(store data unit)을 포함할 수 있으며, 이들 각각은 메모리 유닛(1170) 내의 데이터 TLB 유닛(1172)에 연결된다. 명령어 캐시 유닛(1134)은 또한 메모리 유닛(1170) 내의 L2(level 2) 캐시 유닛(1176)에 연결된다. L2 캐시 유닛(1176)은 하나 이상의 다른 레벨의 캐시에 그리고 궁극적으로는 주 메모리에 연결된다.

[0065] 예로서, 예시적인 레지스터 리네이밍, 비순차 발행/실행 코어 아키텍처는 다음과 같이 파이프라인(1100)을 구현할 수 있다. 즉, 1) 명령어 페치(1138)는 페치 스테이지(1102) 및 길이 디코딩 스테이지(1104)를 수행한다; 2) 디코드 유닛(1140)은 디코드 스테이지(1106)를 수행한다; 3) 리네이밍/할당기 유닛(1152)은 할당 스테이지(1108) 및 리네이밍 스테이지(1110)를 수행한다; 4) 스케줄러 유닛(들)(1156)은 스케줄링 스테이지(1112)를 수행한다; 5) 물리 레지스터 파일(들) 유닛(들)(1158) 및 메모리 유닛(1170)은 레지스터 읽기/메모리 읽기 스테이지(1114)를 수행한다; 실행 클러스터(1160)는 실행 스테이지(1116)를 수행한다; 6) 메모리 유닛(1170) 및 물리 레지스터 파일(들) 유닛(들)(1158)은 라이트 백/메모리 쓰기 스테이지(1118)를 수행한다; 7) 각종 유닛은 예외처리 스테이지(1122)에 연루될 수 있다; 그리고 8) 리타이어먼트 유닛(1154) 및 물리 레지스터 파일(들) 유닛

(들)(1158)은 커밋 스테이지(1124)를 수행한다.

[0066] 코어(1190)는 하나 이상의 명령어 집합(예를 들면, (최신의 버전에서 추가된 일부 확장을 갖는) x86 명령어 집합; 캘리포니아 서니베일 소재의 MIPS Technologies의 MIPS 명령어 집합; 본 명세서에 기술된 명령어(들)를 비롯한 (캘리포니아 서니베일 소재의 (ARM Holdings의 NEON과 같은 선택적인 부가적인 확장을 갖는)) ARM 명령어 집합을 지원할 수 있다. 일 실시예에서, 코어(1190)는 패킹된 데이터 명령어 집합 확장(예를 들면, AVX1, AVX2)을 지원하는 로직을 포함하며, 그럼으로써 많은 멀티미디어 애플리케이션에 의해 사용되는 연산이 패킹된 데이터를 사용하여 수행될 수 있게 한다.

[0067] 코어는 (스레드 또는 연산의 두 개 이상의 병렬 세트를 실행하는) 멀티스레딩을 지원할 수 있으며, 시간 분할된 멀티스레딩(time sliced multithreading), (단일의 물리 코어는, 물리 코어가 동시에 멀티스레딩하는 각 스레드에 논리적 코어를 제공하는) 동시적 멀티스레딩(simultaneously multithreading), 또는 이들의 조합(예를 들어, 인텔® 하이퍼스레딩(Hyperthreading) 기술에서와 같이 시간 분할된 페칭과 디코딩 및 그 이후의 동시적 멀티스레딩)을 비롯한 다양한 방식으로 그렇게 할 수 있다는 것을 이해하여야 한다.

[0068] 레지스터 리네이밍이 비순차 실행의 맥락에서 기술되지만, 레지스터 리네이밍은 순차 아키텍처에서 사용될 수 있다는 것을 이해하여야 한다. 프로세서의 예시된 실시예가 또한 별개의 명령어 및 데이터 캐시 유닛(1134/1174) 및 공유된 L2 캐시 유닛(1176)을 포함하고 있지만, 대안의 실시예는, 예를 들어, 레벨 1(Level 1, L1) 내부 캐시 또는 다중 레벨의 내부 캐시와 같이, 명령어 및 데이터 둘 다를 위한 단일의 내부 캐시를 가질 수 있다. 일부 실시예에서, 시스템은 내부 캐시와 코어 및/또는 프로세서의 외부에 있는 외부 캐시와의 조합을 포함할 수 있다. 대안으로, 모든 캐시가 코어 및/또는 프로세서의 외부에 있을 수 있다.

[0069] 특정의 예시적인 순차 코어 아키텍처

[0070] 도 12(a) 및 도 12(b)는 보다 구체적인 예시적인 순차 코어 아키텍처의 블록도를 예시한 것으로, 이 코어는 칩에 있는 (동일한 종류 및/또는 상이한 종류의 다른 코어를 포함하는) 몇 개의 로직 블록 중 하나일 것이다. 로직 블록은 애플리케이션에 따라서, 고 대역폭 인터커넥트 네트워크(예를 들어, 링 네트워크)를 통해 일부 고정된 기능 로직, 메모리 I/O 인터페이스, 및 다른 필요한 I/O 로직과 통신한다.

[0071] 도 12(a)는 본 발명의 실시예에 따라, 단일 프로세서 코어와 함께, 이 코어의 온-다이 인터커넥트 네트워크(on-die interconnect network)(1202)와의 접속과, 이 코어의 레벨 2(L2) 캐시의 로컬 서브세트(1204)의 블록도이다. 일 실시예에서, 명령어 디코더(1200)는 패킹된 데이터 명령어 집합 확장을 갖는 x86 명령어 집합을 지원한다. L1 캐시(1206)는 스칼라 및 벡터 유닛으로 캐시 메모리로의 저 지연시간 액세스를 가능하게 한다. (설계를 단순화하기 위해) 일 실시예에서, 스칼라 유닛(1208) 및 벡터 유닛(1210)이 별개의 레지스터 세트(각기 스칼라 레지스터(1212) 및 벡터 레지스터(1214))를 사용하고 이들 사이에서 전달되는 데이터가 메모리에 기록된 다음, 레벨 1(L1) 캐시(1206)로부터 다시 판독되지만, 본 발명의 대안의 실시예는 (예를 들면, 단일의 레지스터 세트를 사용하거나, 기록되고 다시 판독되는 일 없이 2개의 레지스터 파일 간에 데이터가 전달될 수 있게 하는 통신 경로를 포함하는) 상이한 접근법을 사용할 수 있다.

[0072] L2 캐시의 로컬 서브세트(1204)는, 프로세서 코어당 하나씩, 별개의 로컬 서브세트로 나누어진 전역적 L2 캐시의 부분이다. 각 프로세서 코어는 자체의 L2 캐시의 로컬 서브세트(1204)에의 직접적인 액세스 경로를 갖는다. 프로세서 코어에 의해 판독된 데이터는 그들 자체의 L2 캐시 서브세트(1204)에 저장되고, 그들 자체의 로컬 L2 캐시 서브세트에 액세스하는 다른 프로세서 코어와 병렬로 빠르게 액세스될 수 있다. 프로세서 코어에 의해 기록된 데이터는 그들 자체의 L2 캐시 서브세트(1204)에 저장되고, 필요한 경우, 다른 서브세트부터 플러시(flush)된다. 링 네트워크는 공유 데이터에 대한 일관성(coherency)을 보장한다. 링 네트워크는 프로세서 코어, L2 캐시 및 다른 로직 블록과 같은 에이전트가 칩 내에서 서로 통신할 수 있게 하기 위해 양방향성이다. 각각의 링 데이터 경로는 방향당 1012-비트 폭이다.

[0073] 도 12(b)는 본 발명의 실시예에 따른 도 12(a)의 프로세서 코어의 부분의 확대도이다. 도 12(b)는 L1 캐시(1204)의 L1 데이터 캐시(1206A) 부분은 물론, 벡터 유닛(1210) 및 벡터 레지스터(1214)에 관한 더 상세한 사항을 포함한다. 구체적으로, 벡터 유닛(1210)은 정수, 단정도(single-precision) 부동 소수점, 및 배정도(doubleprecision) 부동 소수점 명령어 중 하나 이상을 실행하는 16-폭의(16-wide) 벡터 프로세싱 유닛(vector processing unit, VPU)이다(16-폭의 ALU(1228)를 참조할 것). VPU는 뒤섞기 유닛(swizzle unit)(1220)을 사용하여 레지스터 입력을 뒤섞기, 수치 변환 유닛(1222A 및 1222B)을 사용한 수치 변환, 및 메모리 입력에 대해 복제 유닛(1224)을 이용한 복제를 지원한다. 쓰기 마스크 레지스터(1226)는 결과로 생긴 벡터 기록값을 예측하는

것을 가능하게 한다.

[0074] 통합된 메모리 제어기 및 그래픽스를 갖는 프로세서

[0075] 도 13는 본 발명의 실시예에 따라 하나보다 많은 코어를 가질 수 있고, 통합된 메모리 제어기를 가질 수 있고, 통합된 그래픽스를 가질 수 있는 프로세서(1300)의 블록도이다. 도 13에서의 실선 박스는 단일의 코어(1302A), 시스템 에이전트(1310), 및 한 세트의 하나 이상의 버스 제어기 유닛(1316)을 갖는 프로세서(1300)를 예시하는 반면, 점선 박스의 선택적으로 추가된 것은 복수의 코어(1302A-N), 시스템 에이전트 유닛(1310) 내의 한 세트의 하나 이상의 통합된 메모리 제어기 유닛(들)(1314), 및 특수 목적 로직(1308)을 갖는 대안의 프로세서(1300)를 예시한다.

[0076] 그러므로 프로세서(1300)의 상이한 구현은 1) (하나 이상의 코어를 포함할 수 있는) 통합된 그래픽스 및/또는 과학적(처리량) 로직인 특수 목적 로직(1308), 및 하나 이상의 범용 코어(예를 들어, 범용 순차 코어, 범용 비순차 코어, 이 둘의 조합)인 코어(1302A-N)를 갖는 CPU; 2) 주로 그래픽스 및/또는 과학(처리량) 용도로 의도된 많은 수의 특수 목적 코어인 코어(1302A-N)를 갖는 코프로세서; 및 3) 많은 수의 범용 순차 코어인 코어(1302A-N)를 갖는 코프로세서를 포함할 수 있다. 따라서, 프로세서(1300)는, 예를 들면, 네트워크 또는 통신 프로세서, 압축 엔진, 그래픽스 프로세서, GPGPU(general purpose graphics processing unit), 고 처리량의 다중 통합 코어(many integrated core, MIC) 코프로세서(30개 이상의 코어를 포함함), 또는 임베디드 프로세서 등과 같은 범용 프로세서, 코프로세서 또는 특수 목적 프로세서일 수 있다. 프로세서는 하나 이상의 칩 상에 구현될 수 있다. 프로세서(1300)는, 예를 들어, BiCMOS, CMOS, 또는 NMOS와 같은 복수의 프로세스 기술 중 임의의 기술을 사용하여 하나 이상의 기판의 일부일 수 있고 그리고/또는 이들 기판상에 구현될 수 있다.

[0077] 메모리 계층(memory hierarchy)은 코어 내의 하나 이상 레벨의 캐시, 한 세트 또는 하나 이상의 공유된 캐시 유닛(1306), 및 통합된 메모리 제어기 유닛 세트(1314)에 연결된 외부 메모리(도시되지 않음)를 포함한다. 공유된 캐시 유닛 세트(1306)는 레벨 2(L2), 레벨 3(L3), 레벨 4(L4), 또는 다른 레벨의 캐시 중 하나 이상의 중간 레벨 캐시, 최종 레벨 캐시(last level cache, LLC), 및/또는 이들의 조합을 포함할 수 있다. 일 실시예에서, 링 기반 인터커넥트 유닛(ring based interconnect unit)(1312)은 통합된 그래픽스 로직(1308), 공유된 캐시 유닛 세트(1306), 및 시스템 에이전트 유닛(1310)/통합된 메모리 제어기 유닛(들)(1314)을 상호접속하지만, 대안의 실시예는 이러한 유닛을 상호접속시키기 위한 임의의 수의 공지된 기술을 사용할 수 있다. 일 실시예에서, 하나 이상의 캐시 유닛(1306)과 코어(1302A-N) 사이에 일관성이 유지된다.

[0078] 일부 실시예에서, 코어(1302A-N)의 하나 이상의 코어는 멀티스레딩을 수행할 수 있다. 시스템 에이전트(1310)는 코어(1302A-N)를 조정하고 동작시키는 그러한 컴포넌트를 포함한다. 시스템 에이전트 유닛(1310)은 예를 들면, 전력 제어 유닛(power control unit, PCU) 및 디스플레이 유닛을 포함할 수 있다. PCU는 코어(1302A-N) 및 통합된 그래픽스 로직(1308)의 전력 상태를 조절하는 데 필요한 로직 및 컴포넌트이거나 이들을 포함할 수 있다. 디스플레이 유닛은 하나 이상의 외부에 접속된 디스플레이를 구동하기 위한 유닛이다.

[0079] 코어(1302A-N)는 아키텍처 명령어 집합의 관점에서 동종이거나 이종일 수 있는데, 즉, 코어(1302A-N) 중 둘 이상의 코어는 동일한 명령어 집합을 실행할 수 있는 반면, 다른 코어는 오로지 그 명령어 집합의 서브세트 또는 상이한 명령어 집합을 실행할 수 있다.

[0080] 예시적인 컴퓨터 아키텍처

[0081] 도 14 내지 도 17은 예시적인 컴퓨터 아키텍처의 블록도이다. 랩톱, 데스크톱, 휴대형 PC, 개인 휴대정보 단말기, 엔지니어링 워크스테이션, 서버, 네트워크 디바이스, 네트워크 허브, 스위치, 임베디드 프로세서, 디지털 신호 프로세서(digital signal processor, DSP), 그래픽스 디바이스, 비디오 게임 디바이스, 셋톱 박스, 마이크로컨트롤러, 셀 폰, 휴대용 미디어 플레이어, 휴대형 디바이스 및 다양한 다른 전자 디바이스에 대한 기술에서 공지된 다른 시스템 설계 및 구성이 또한 적합하다. 일반적으로, 본 명세서에 개시된 바와 같은 프로세서 및/또는 다른 실행 로직을 통합할 수 있는 매우 다양한 시스템 또는 전자 디바이스는 대체로 적합하다.

[0082] 이제 도 14를 참조하면, 본 발명의 일 실시예에 따른 시스템(1400)의 블록도가 도시된다. 시스템(1400)은 하나 이상의 프로세서(1410, 1415)를 포함할 수 있으며, 이들 프로세서는 제어기 허브(1420)에 연결된다. 일 실시예에서, 제어기 허브(1420)는 그래픽스 메모리 제어기 허브(graphics memory controller hub, GMCH)(1490) 및 입력/출력 허브(Input/Output Hub, IOH)(1450)(별개의 칩 상에 있을 수 있음)를 포함하고, GMCH(1490)는 메모리(1440) 및 코프로세서(1445)에 연결된 메모리 및 그래픽스 제어기를 포함하며, IOH(1450)는 입력/출력(input/output, I/O) 디바이스(1460)를 GMCH(1490)에 연결한다. 대안으로, 메모리 및 그래픽스 제어기 중 하나

또는 둘 다는 (본 명세서에 기술된 바와 같이) 프로세서 내에 통합되어 있고, 메모리(1440) 및 코프로세서(1445)는 프로세서(1410), 및 IOH(1450)를 갖는 단일 칩 내의 제어기 허브(1420)에 직접 연결된다.

[0083] 부가적인 프로세서(1415)의 선택적인 속성은 도 14에서 점선으로 표시되어 있다. 각 프로세서(1410, 1415)는 본 명세서에 설명된 하나 이상의 프로세싱 코어를 포함할 수 있으며 프로세서(1500)의 일부 버전일 수 있다.

[0084] 메모리(1440)는, 예를 들면, 다이나믹 랜덤 액세스 메모리(dynamic random access memory, DRAM), 상변화 메모리(phase change memory, PCM), 또는 이 둘의 조합일 수 있다. 적어도 일 실시예에서, 제어기 허브(1420)는 전면 버스(frontside bus, FSB)와 같은 멀티-드롭 버스(multi-drop bus), QPI(QuickPath Interconnect)와 같은 점대점(point-to-point) 인터페이스, 또는 유사한 커넥션(1495)을 통해 프로세서(들)(1410, 1415)와 통신한다.

[0085] 일 실시예에서, 코프로세서(1445)는, 예를 들어, 고 처리량의 MIC 프로세서, 네트워크 또는 통신 프로세서, 압축 엔진, 그래픽스 프로세서, GPGPU, 임베디드 프로세서 등과 같은 특수 목적 프로세서이다. 일 실시예에서, 제어기 허브(1420)는 통합된 그래픽스 가속기를 포함할 수 있다.

[0086] 아키텍처적, 마이크로아키텍처적, 열적(thermal), 및 전력 소비 특성 등을 비롯한 다양한 장점 지표 측면에서 물리 자원(1410, 1415) 사이에는 여러 가지 차이가 있을 수 있다.

[0087] 일 실시예에서, 프로세서(1410)는 일반적인 유형의 데이터 프로세싱 동작을 제어하는 명령어를 실행한다. 명령어 내에는 코프로세서 명령어가 내장될 수 있다. 프로세서(1410)는 이러한 코프로세서 명령어를 소속된 코프로세서(1445)에 의해 실행되어야만 하는 유형이라고 인식한다. 이에 따라, 프로세서(1410)는 이러한 코프로세서 명령어(또는 코프로세서 명령어를 나타내는 제어 신호)를 코프로세서 버스 또는 다른 인터커넥트를 통해 코프로세서(1445)로 발행한다. 코프로세서(들)(1445)는 수신된 코프로세서 명령어를 접수하고 실행한다.

[0088] 이제 도 15를 참조하면, 본 발명의 일 실시예에 따른 보다 구체적이고 예시적인 제 1 시스템(1500)의 블록도가 도시된다. 도 15에 도시된 바와 같이, 멀티프로세서 시스템(1500)은 점대점 인터커넥트 시스템(point-to-point interconnect system)이고, 점대점 인터커넥트(1550)를 통해 연결되는 제 1 프로세서(1570) 및 제 2 프로세서(1580)를 포함한다. 각 프로세서(1570 및 1580)는 프로세서(1500)의 일부 버전일 수 있다. 본 발명의 일 실시예에서, 프로세서(1570 및 1580)은 각기 프로세서(1410 및 1415)인 반면, 코프로세서(1538)는 코프로세서(1445)이다. 다른 실시예에서, 프로세서(1570 및 1580)는 각기 프로세서(1410) 및 코프로세서(1445)이다.

[0089] 프로세서(1570 및 1580)는 각기 통합된 메모리 제어기(integrated memory controller, IMC) 유닛(1572 및 1582)을 포함하는 것으로 도시되어 있다. 프로세서(1570)는 또한 그의 버스 제어기 유닛의 일부로서, 점대점(point-to-point, P-P) 인터페이스(1576 및 1578)를 포함하며, 이와 유사하게, 제 2 프로세서(1580)는 P-P 인터페이스(1586 및 1588)를 포함한다. 프로세서(1570, 1580)는 P-P 인터페이스 회로(1578, 1588)를 사용하여 점대점(point-to-point, P-P) 인터페이스(1550)를 통해 정보를 교환할 수 있다. 도 15에 도시된 바와 같이, IMC(1572 및 1582)는 프로세서를 각자의 메모리(1532 및 1534), 즉, 각자의 프로세서에 국부적으로 소속된 메인 메모리의 일부분일 수 있는 메모리(1532) 및 메모리(1534)에 연결한다.

[0090] 프로세서(1570, 1580)는 각기 점대점 인터페이스 회로(1576, 1594, 1586, 1598)를 사용하여 개개의 P-P 인터페이스(1552, 1554)를 통해 칩셋(1590)과 정보를 교환할 수 있다. 칩셋(1590)은 선택적으로 고성능 인터페이스(1539)를 통해 코프로세서(1538)와 정보를 교환할 수 있다. 일 실시예에서, 코프로세서(1538)는, 예를 들면, 고 처리량의 MIC 프로세서, 네트워크 또는 통신 프로세서, 압축 엔진, 그래픽스 프로세서, GPGPU, 임베디드 프로세서 등과 같은 특수 목적 프로세서이다.

[0091] 공유된 캐시(도시되지 않음)가 어느 하나의 프로세서에 또는 양자 모두의 프로세서의 외부에 포함될 수 있지만, P-P 인터커넥트를 통해 프로세서와 접속되어, 프로세서가 저 전력 모드에 놓이면, 어느 하나의 프로세서 또는 양자 모두의 프로세서의 로컬 캐시 정보가 공유 캐시에 저장될 수 있도록 한다.

[0092] 칩셋(1590)은 인터페이스(1596)를 통해 제 1 버스(1516)에 연결될 수 있다. 일 실시예에서, 제 1 버스(1516)는 주변 컴포넌트 인터커넥트(Peripheral Component Interconnect, PCI) 버스일 수 있거나, 또는 PCI 익스프레스(Express) 버스 또는 다른 제 3 세대 I/O 인터커넥트 버스와 같은 버스일 수 있지만, 본 발명의 범위는 이렇게 제한되지 않는다.

[0093] 도 15에 도시된 바와 같이, 제 1 버스(1516)를 제2 버스(1520)에 연결하는 버스 브리지(1518)와 함께, 다양한 I/O 디바이스(1514)가 제 1 버스(1516)에 연결될 수 있다. 일 실시예에서, 코프로세서, 고 처리량의 MIC 프로세서, GPGPU, (예를 들면, 그래픽스 가속기 또는 디지털 신호 프로세싱(digital signal processing, DSP) 유닛

등과 같은) 가속기, 필드 프로그래머블 게이트 어레이, 또는 임의의 다른 프로세서와 같은 하나 이상의 부가적인 프로세서(들)(1515)가 제 1 버스(1516)에 연결된다. 일 실시예에서, 제 2 버스(1520)는 로우 핀 카운트(low pin count, LPC) 버스일 수 있다. 일 실시예에서, 예를 들면, 키보드 및/또는 마우스(1522), 통신 디바이스(1527), 및 명령어/코드 및 데이터(1530)를 포함할 수 있는 디스크 드라이브 또는 기타 대용량 저장 디바이스와 같은 저장 유닛(1528)을 비롯한 각종 디바이스가 제 2 버스(1520)에 연결될 수 있다. 또한, 오디오 I/O(1524)가 제2 버스(1520)에 연결될 수 있다. 다른 아키텍처도 가능하다는 것에 주목한다. 예를 들면, 도 15의 점대점 아키텍처 대신에, 시스템은 멀티드롭 버스 또는 다른 이러한 아키텍처를 구현할 수 있다.

[0094] 이제 도 16을 참조하면, 본 발명의 일 실시예에 따른 보다 구체적이고 예시적인 제 2 시스템(1600)의 블록도가 도시된다. 도 15 및 도 16에서의 같은 요소는 같은 참조 번호를 가지며, 도 16의 다른 양태를 불명료하게 하는 것을 피하기 위해 도 15의 특정 양태가 도 16에서 생략되었다.

[0095] 도 16은 프로세서(1570, 1580)가 각기 통합된 메모리 및 I/O 제어 로직("control logic, CL")(1572 및 1582)을 포함할 수 있다는 것을 예시한다. 그러므로 CL(1572, 1582)은 통합된 메모리 제어기 유닛을 포함하며 I/O 제어 로직을 포함한다. 도 16은 메모리(1532, 1534)가 CL(1572, 1582)에 연결되어 있을 뿐만 아니라 I/O 디바이스(1614)도 또한 제어 로직(1572, 1582)에 연결되어 있는 것을 예시한다. 레거시 I/O 디바이스(1615)는 칩셋(1590)에 연결된다.

[0096] 이제 도 17을 참조하면, 본 발명의 일 실시예에 따른 SoC(1700)의 블록도가 도시된다. 도 13에 있는 유사한 요소는 같은 참조 부호를 갖는다. 또한, 점선 박스는 더 진보된 SoC에 대한 선택적인 특징이다. 도 17에서, 인터커넥트 유닛(들)(1702)은 한 세트의 하나 이상의 코어(202A-N) 및 공유된 캐시 유닛(들)(1306)을 포함하는 애플리케이션 프로세서(1710); 시스템 에이전트 유닛(1310); 버스 제어기 유닛(들)(1316); 통합된 메모리 제어기 유닛(들)(1314); 통합된 그래픽스 로직, 이미지 프로세서, 오디오 프로세서, 및 비디오 프로세서를 포함할 수 있는 한 세트의 하나 이상의 코프로세서(1720); 스태틱 랜덤 액세스 메모리(static random access memory, SRAM) 유닛(1730); 직접 메모리 액세스(direct memory access, DMA) 유닛(1732); 및 하나 이상의 외부 디스플레이에 연결하기 위한 디스플레이 유닛(1740)에 연결된다. 일 실시예에서, 코프로세서(들)(1720)는, 예를 들면, 네트워크 또는 통신 프로세서, 압축 엔진, GPGPU, 고처리량 MIC 프로세서, 임베디드 프로세서 등과 같은 특수 목적 프로세서를 포함한다.

[0097] 본 명세서에 개시된 메커니즘의 실시예는 하드웨어, 소프트웨어, 펌웨어, 또는 이와 같은 구현 접근법의 조합으로 구현될 수 있다. 본 발명의 실시예는 적어도 하나의 프로세서, (휘발성 및 비휘발성 메모리 및/또는 저장 요소를 포함하는) 저장 시스템, 적어도 하나의 입력 디바이스, 및 적어도 하나의 출력 디바이스를 포함하는 프로그래머블 시스템상에서 실행되는 컴퓨터 프로그램 또는 프로그램 코드로서 구현될 수 있다.

[0098] 도 15에 예시된 코드(1530)와 같은 프로그램 코드는 본 명세서에서 기술된 기능을 수행하기 위해 명령어를 입력하며 출력 정보를 생성하기 위해 적용될 수 있다. 출력 정보는 공지된 방식으로 하나 이상의 출력 디바이스에 제공될 수 있다. 이러한 응용을 위해, 프로세싱 시스템은, 예를 들어, 디지털 신호 프로세서(DSP), 마이크로컨트롤러, 주문형 집적 회로(application specific integrated circuit, ASIC), 또는 마이크로프로세서와 같은 프로세서를 갖는 임의의 시스템을 포함한다.

[0099] 프로그램 코드는 프로세싱 시스템과 통신하기 위해 고급의 절차적 또는 객체 지향형 프로그래밍 언어로 구현될 수 있다. 요구된다면, 프로그램 코드는 또한 어셈블리 또는 기계 언어로 구현될 수 있다. 실제로, 본 명세서에 설명된 메커니즘은 임의의 특정 프로그래밍 언어로 범위가 제한되지는 않는다. 어떤 경우든, 언어는 컴파일링되거나 해석된 언어일 수 있다.

[0100] 적어도 일 실시예의 하나 이상의 양태는, 머신에 의해 판독될 때 이 머신으로 하여금 본 명세서에 설명된 기술을 수행하는 로직을 제작하게 하는, 프로세서 내의 다양한 로직을 나타내는 머신 판독가능한 매체상에 저장된 대표적인 명령어에 의해 구현될 수 있다. "IP 코어"로 알려진 그러한 표현은 유형의 머신 판독 가능한 매체에 저장될 수 있으며 실제로 다양한 고객에게 공급되거나 로직 또는 프로세서를 만드는 제조 기계에다 적재시키는 제조 설비에 공급될 수 있다.

[0101] 이러한 머신 판독가능한 저장 매체는 하드 디스크와, 플로피 디스크, 광디스크, 콤팩트 디스크 판독 전용 메모리(compact disk read-only memory, CD-ROM), 콤팩트 디스크 리라이터블(compact disk rewritable, CDRW) 및 광자기 디스크를 포함하는 임의의 다른 유형의 디스크와, 판독 전용 메모리(read-only memory, ROM)와, 다이나믹 랜덤 액세스 메모리(dynamic random access memory, DRAM), 스태틱 랜덤 액세스 메모리(static random

access memory, SRAM)와 같은 랜덤 액세스 메모리(Random Access Memory, RAM), 소거가능한 프로그래머블 판독 전용 메모리(erasable programmable read-only memory, EPROM), 플래시 메모리, 전기적으로 소거가능한 프로그래머블 판독 전용 메모리(electrically erasable programmable read-only memory, EEPROM), 상변화 메모리(phase change memory, PCM)와 같은 반도체 디바이스, 자기 또는 광 카드, 또는 전자 명령어를 저장하는 데 적합한 임의의 다른 종류의 매체와 같은 저장 매체를 포함하는, 머신 또는 디바이스에 의해 제조 또는 형성되는 물품의 비일시적 유형의(tangible) 구성을 포함할 수 있지만, 이들로 제한되지 않는다.

[0102] 따라서, 본 발명의 실시예는 명령어를 포함하거나, 또는 본 명세서에 설명된 구조, 회로, 장치, 프로세서 및/또는 시스템 특징을 정의하는 하드웨어 서술 언어(Hardware Description Language, HDL)와 같은 설계 데이터를 포함하는 비일시적인 유형의 머신 판독가능한 매체를 또한 포함한다. 이러한 실시예는 프로그램 제품이라고도 또한 지칭될 수 있다.

[0103] (이진 변환, 코드 모핑(code morphing) 등을 비롯한) 에뮬레이션

[0104] 일부 경우에 있어서, 명령어 변환기는 소스 명령어 집합으로부터 타겟 명령어 집합으로 명령어를 변환하는데 사용될 수 있다. 예를 들면, 명령어 변환기는 명령어를 코어에 의해 처리될 하나 이상의 다른 명령어로 (예를 들어, 정적 이진 변환, 동적 컴파일을 비롯한 동적 이진 변환을 이용하여) 번역하거나, 모핑하거나, 에뮬레이트하거나, 또는 다른 방식으로 변환할 수 있다. 명령어 변환기는 소프트웨어, 하드웨어, 펌웨어 또는 이들의 조합으로 구현될 수 있다. 명령어 변환기는 온 프로세서(on processor), 오프 프로세서(off processor), 또는 부분 온 및 부분 오프 프로세서(part on and part off processor)일 수 있다.

[0105] 도 18은 본 발명의 실시예에 따라 소프트웨어 명령어 변환기를 사용하여 소스 명령어 집합 내의 이진 명령어를 변환하는 것과 타겟 명령어 집합 내의 이진 명령어로 변환하는 것을 대비하는 블록도이다. 예시된 실시예에서, 명령어 변환기는 소프트웨어 명령어 변환기이지만, 대안으로 명령어 변환기는 소프트웨어, 펌웨어, 하드웨어 또는 이들의 다양한 조합으로 구현될 수 있다. 도 18은 고급 언어(1802)로 된 프로그램이 x86 컴파일러(1804)를 사용하여 컴파일되어, 적어도 하나의 x86 명령어 집합 코어(1816)를 갖춘 프로세서에 의해 기본적으로 실행될 수 있는 x86 이진 코드(1806)를 생성할 수 있다는 것을 보여준다. 적어도 하나의 x86 명령어 집합 코어(1816)를 갖춘 프로세서는, 적어도 하나의 x86 명령어 집합 코어를 갖춘 Intel 프로세서와 실질적으로 동일한 결과를 달성하기 위해, (1) Intel x86 명령어 집합 코어의 명령어 집합의 상당 부분, 또는 (2) 적어도 하나의 x86 명령어 집합 코어를 갖춘 Intel 프로세서상에서 실행하도록 정해져 있는 애플리케이션 또는 다른 소프트웨어의 오브젝트 코드 버전을, 호환 가능하게 실행하거나 다른 방식으로 처리함으로써 적어도 하나의 x86 명령어 집합 코어를 갖춘 Intel 프로세서와 실질적으로 동일한 기능을 수행할 수 있는 임의의 프로세서를 나타낸다. x86 컴파일러(1804)는 부수적인 연결 프로세싱에 의해 또는 부수적인 연결 프로세싱 없이, 적어도 하나의 x86 명령어 집합 코어를 갖춘 프로세서(1816) 상에서 실행될 수 있는 x86 이진 코드(1806)(예컨대, 오브젝트 코드)를 생성하도록 동작 가능한 컴파일러를 나타낸다. 이와 유사하게, 도 18은 고급 언어(1802)로 된 프로그램이 대안의 명령어 집합 컴파일러(1808)를 사용하여 컴파일되어, 적어도 하나의 x86 명령어 집합 코어가 없는 프로세서(1814)(예를 들면, 캘리포니아 서니베일 소재의 MIP Technologies의 MIPS 명령어 집합을 실행하는 및/또는 캘리포니아 서니베일 소재의 ARM Holdings의 ARM 명령어 집합을 실행하는 코어를 갖춘 프로세서)에 의해 기본적으로 실행될 수 있는 대안의 명령어 집합 이진 코드(1810)를 생성할 수 있다는 것을 나타낸 것이다. 명령어 변환기(1812)는 x86 이진수 코드(1806)를 x86 명령어 집합 코어가 없는 프로세서(1814)에 의해 기본적으로 실행될 수 있는 코드로 변환하는 데 사용된다. 이것을 할 수 있는 명령어 변환기를 만들기 어렵기 때문에 이와 같이 변환된 코드가 대안의 명령어 집합 이진 코드(1810)와 동일하지 않을 가능성이 있지만, 변환된 코드는 일반적인 연산을 달성할 것이고 대안의 명령어 집합으로부터의 명령어로 이루어져 있을 것이다. 그러므로 명령어 변환기(1812)는 에뮬레이션, 시뮬레이션 또는 임의의 다른 프로세스를 통해, x86 명령어 집합 프로세서 또는 코어를 갖지 않은 프로세서 또는 다른 전자 디바이스가 x86 이진 코드(1806)를 실행할 수 있게 하는 소프트웨어, 펌웨어, 하드웨어, 또는 이들의 조합을 나타낸다.

[0106] 도 3 내지 도 10 중 임의의 도면에 대해 기술되는 컴포넌트, 특징 및 세부 사항은 또한 도 1 내지 도 2 중 임의의 도면에 선택적으로 적용될 수 있다. 또한, 장치 중 임의의 장치에 대해 기술된 컴포넌트, 특징, 및 세부 사항은 실시예에서 그러한 장치에 의해 및/또는 그러한 장치를 가지고 수행될 수 있는 임의의 방법에도 선택적으로 적용될 수 있다. 본 명세서에 기술된 프로세서 중 임의의 프로세서는 본 명세서에 기술된 컴퓨터 시스템 중 임의의 컴퓨터 시스템에 포함될 수 있다. 일 양태에서, 시스템은 패킷 처리 관련 작동을 위해 본 명세서에 개시된 명령어를 이용할 수 있는, 예를 들면, 스위치, 라우터, 다른 네트워크 가전기기(예를 들어, 방화벽, 스니퍼(sniffer) 등)와 같은 네트워킹 장비로서 채용될 수 있겠지만, 본 발명의 범위는 그렇게 제한되지는 않는다. 일

부 실시예에서, 필수 사항은 아니지만, 명령어는 본 명세서에서 개시된 명령어 포맷의 특징 또는 세부사항을 가질 수 있다.

[0107] 설명 및 청구항에서, "연결된(coupled)" 및/또는 "접속된(connected)"이라는 용어는 이들의 파생어와 함께 사용될 수 있다. 이들 용어는 서로에 대한 동의어로서 의도되는 것은 아니다. 오히려, 실시예에서, "접속된"은 2개 이상의 요소가 서로 직접적인 물리적 및/또는 전기적 접촉을 이루고 있는 것을 표시하는데 사용될 수 있다. "연결된"은 2개 이상의 요소가 서로 직접적인 물리적 및/또는 전기적 접촉을 이루고 있는 것을 의미할 수 있다. 그러나 "연결된"은 또한 2개 이상의 요소가 서로 직접 접촉하지는 않지만, 아직도 여전히 서로 협력하거나 상호작용하는 것을 의미할 수 있다. 예를 들면, 실행 유닛은 하나 이상의 중개 컴포넌트를 통해 레지스터 또는 디코드 유닛과 연결될 수 있다. 도면에서, 화살표는 접속 및 연결을 보여주기 위해 사용된다.

[0108] "및/또는"이라는 용어가 사용된다. 본 명세서에서 사용되는 바와 같이, "및/또는"이라는 용어는 하나 또는 다른 것 또는 모두 다를 의미한다(예를 들면, A 및/또는 B는 A 또는 B 또는 A와 B 모두 다를 의미한다).

[0109] 전술한 설명에서는 실시예의 충분한 이해를 제공하기 위해 특정의 세부 사항이 설명되었다. 그러나 다른 실시예는 이러한 특정 세부사항 중 일부가 없어도 구현될 수 있다. 본 발명의 범위는 위에서 제공된 특정 예에 의해서가 아니라 아래의 청구범위에 의해서만 결정되어야 한다. 다른 예로, 잘 알려진 회로, 구조, 디바이스, 및 동작은 설명의 이해를 불명료하게 하는 것을 피하기 위해 블록도의 형태로 및/또는 세부 사항 없이 도시되었다. 적절하다고 간주하는 경우, 달리 명시되지 않거나 명명백백하지 않다면, 유사하거나 동일한 특징을 선택적으로 가질 수 있는 대응하거나 유사한 요소를 표시하기 위해, 참조 부호 또는 참조 부호의 끝 부분이 도면에서 반복된다.

[0110] 특정 동작이 하드웨어 컴포넌트에 의해 수행될 수 있거나, 머신, 회로, 또는 하드웨어 컴포넌트(예를 들어, 프로세서, 프로세서의 부분, 회로 등)로 하여금 동작을 수행하는 명령어로 프로그래밍되도록 하거나 그리고/또는 그렇게 되도록 초래하는데 사용될 수 있는 머신 실행 가능한 또는 회로 실행 가능한 명령어로 구현될 수 있다. 동작은 또한 선택적으로 하드웨어와 소프트웨어의 조합에 의해 수행될 수 있다. 프로세서, 머신, 회로, 또는 하드웨어는 특유하거나 특정한 회로를 포함할 수 있고, 또는 다른 로직(예를 들어, 펌웨어 및/또는 소프트웨어와 잠재적으로 조합된 하드웨어)은 명령어를 실행 및/또는 처리하고 그 명령어에 응답하여 결과를 저장하도록 동작 가능하다.

[0111] 일부 실시예는 머신 판독가능한 매체를 포함하는 제조 물품(예를 들면, 컴퓨터 프로그램 제품)을 포함한다. 매체는 머신에 의해 판독가능한 형태의 정보를 제공하는, 예를 들어 저장하는 메커니즘을 포함할 수 있다. 머신 판독가능한 매체는, 머신에 의해 실행되는 경우 및/또는 머신에 의해 실행될 때 머신으로 하여금 본 명세서에 개시된 하나 이상의 동작, 방법 또는 기술을 수행하게 하고 그리고/또는 머신이 결과적으로 하나 이상의 동작, 방법 또는 기술을 수행하게 하는 동작 가능한 명령어 또는 명령어의 시퀀스를 제공하거나 저장할 수 있다.

[0112] 일부 실시예에서, 머신 판독가능한 매체는 비일시적인 머신 판독가능한 저장 매체를 포함할 수 있다. 예를 들면, 비일시적인 머신 판독가능한 저장 매체는 플로피 디스켓, 광 저장 매체, 광디스크, 광 데이터 저장 디바이스, CD-ROM, 자기 디스크, 광자기 디스크, 판독 전용 메모리(read-only memory, ROM), 소거 가능한 프로그래머블 ROM(erasable-and-programmable ROM), 전기적으로 소거 가능한 프로그래머블 ROM EEPROM(electrically-erasable-and-programmable ROM), 랜덤 액세스 메모리(dynamic random access memory, RAM), 스태틱 RAM(SRAM), 다이나믹 RAM(DRAM), 플래시 메모리, 상변화 메모리, 상변화 데이터 저장 재료, 비휘발성 메모리, 비휘발성 데이터 저장 디바이스, 비일시적 메모리, 비일시적 데이터 저장 디바이스 등을 포함할 수 있다. 비일시적 머신 판독가능한 저장 매체는 일시 전파 신호(transitory propagated signal)로 구성되지 않는다. 일부 실시예에서, 저장 매체는 고형물을 포함하는 유형의 매체를 포함할 수 있다.

[0113] 적합한 머신의 예는 이것으로 제한되는 것은 아니지만, 범용 프로세서, 특수 목적 프로세서, 디지털 로직 회로, 또는 집적 회로 등을 포함한다. 적합한 머신의 다른 예는 프로세서, 디지털 로직 회로, 또는 집적 회로를 포함하는 컴퓨터 시스템 또는 다른 전자 디바이스를 포함한다. 이러한 컴퓨터 시스템 또는 전자 디바이스의 예는 이것으로 제한되는 것은 아니지만, 데스크톱 컴퓨터, 랩톱 컴퓨터, 노트북 컴퓨터, 태블릿 컴퓨터, 넷북, 스마트 폰, 셀룰러 폰, 서버, 네트워크 디바이스(예를 들어, 라우터 및 스위치), 모바일 인터넷 디바이스(Mobile Internet device, MID), 미디어 플레이어, 스마트 텔레비전, 넷톱(nettop), 셋톱 박스, 및 비디오 게임 제어기를 포함한다.

[0114] 본 명세서 전체에서 "일 실시예", "실시예", 또는 "하나 이상의 실시예", "일부 실시예"라고 언급하는 것은 예

를 들면, 특별한 특징이 본 발명의 실시에 포함될 수 있지만, 반드시 그렇게 요구되는 것은 아니라는 것을 의미한다. 유사하게, 본 개시를 간소화하고 다양한 본 발명의 양태의 이해를 도울 목적으로, 설명에서는 각종 특징이 때때로 단일 실시예, 도면, 또는 그의 설명에서 함께 그룹화된다. 그러나 본 개시의 이러한 방법은 본 발명이 각 청구항에서 명백하게 인용되는 것보다 더 많은 특징을 요구한다는 의도를 반영하는 것으로서 해석되지 않아야 된다. 오히려, 다음의 청구항이 반영하는 바와 같이, 본 발명의 양태는 개시된 단일 실시예의 모든 특징보다 적다. 따라서, 상세한 설명 다음에 이어지는 청구 범위는 이에 의해 명백히 상세한 설명에 포함되는 것이며, 각각의 청구항은 본 발명의 별개의 실시예로서 자체를 주장한다.

[0115] 예시적인 실시예

[0116] 다음의 예는 추가의 실시예에 관련된다. 예에서의 세부 내용은 하나 이상의 실시예의 어디에서도 사용될 수 있다.

[0117] 예 1은 패킹된 복수의 패킹된 데이터 레지스터와, 데이터 요소 선택 및 통합 명령어를 디코딩하는 디코드 유닛을 포함하는 프로세서이다. 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자를 갖고, 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는다. 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응한다. 프로세서는 또한 디코드 유닛과 연결된 실행 유닛을 포함한다. 실행 유닛은 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자를 데이터 요소 선택 및 통합 명령어에 의해 표시되는 목적지 저장 위치에 저장한다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함한다.

[0118] 예 2는 예 1의 프로세서를 포함하며, 이 예에서 디코드 유닛은 마스크 요소를 갖는 제 2 소스 피연산자로서 이미디어트를 갖는 명령어를 디코딩하며, 이 예에서 마스크 요소는 마스크 비트이다.

[0119] 예 3는 예 1의 프로세서를 포함하며, 이 예에서 디코드 유닛은 프로세서의 한 세트의 패킹된 데이터 연산 마스크 레지스터 중의 패킹된 데이터 연산 마스크 레지스터가 되는 제 2 소스 피연산자를 갖는 명령어를 디코딩한다. 또한 선택적으로 이 예에서, 프로세서의 명령어 집합 중 복수의 다른 명령어는 예측 피연산자를 제공하기 위해 패킹된 데이터 연산 마스크 레지스터 세트 중의 레지스터를 명시한다.

[0120] 예 4는 예 1의 프로세서를 포함하며, 이 예에서 디코드 유닛은 마스크 비트인 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는 명령어를 디코딩한다.

[0121] 예 5는 예 1의 프로세서를 포함하며, 이 예에서 디코드 유닛은 패킹된 데이터 피연산자가 되는 제 2 소스 피연산자를 갖는 명령어를 디코딩하며, 이 예에서 마스크 요소는 마스크 데이터 요소가 된다.

[0122] 예 6은 예 1의 프로세서를 포함하며, 이 예에서 실행 유닛은, 명령어에 응답하여, 결과 패킹된 데이터 피연산자의 최하위 부분에서 함께, 제 1 패킹된 데이터 피연산자에서와 동일한 순서로 통합되는 모든 데이터 요소를 포함하는 결과 패킹된 데이터 피연산자를 저장한다.

[0123] 예 7은 예 1의 프로세서를 포함하며, 이 예에서 실행 유닛은, 명령어에 응답하여, 결과 패킹된 데이터 피연산자의 최상위 부분에서 함께, 제 1 패킹된 데이터 피연산자에서와 동일한 순서로 통합되는 모든 데이터 요소를 포함하는 결과 패킹된 데이터 피연산자를 저장한다.

[0124] 예 8은 예 1의 프로세서를 포함하며, 이 예에서 실행 유닛은, 명령어에 응답하여, 제 2 소스 피연산자의 마스킹하는 마스크 요소에 대응하는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 제외하는 통합된 데이터 요소를 갖는 결과 패킹된 데이터 피연산자를 저장한다. 또한, 선택적으로 이 예에서, 제 1 소스 패킹된 데이터 피연산자는 마스킹하지 않는 마스크 요소에 대응하는 데이터 요소 사이에서 마스킹하는 마스크 요소에 대응하는 적어도 하나의 데이터 요소를 갖는다.

[0125] 예 9는 예 1 내지 예 8 중 어느 한 예의 프로세서를 포함하며, 이 예에서 디코드 유닛은 제 1 소스 패킹된 데이터 피연산자의 데이터 요소의 크기를 표시하는 하나 이상의 비트를 갖는 명령어를 디코딩한다.

[0126] 예 10은 예 1 내지 예 8 중 어느 한 예의 프로세서를 포함하며, 이 예에서 디코드 유닛은 적어도 128-비트의 비트 폭을 갖고 복수의 8-비트 데이터 요소 및 복수의 16-비트 데이터 요소로부터 선택된 복수의 데이터 요소를 갖는 명령어를 디코딩한다. 또한 선택적으로 이 예에서, 목적지 저장 위치는 프로세서의 패킹된 데이터 레지스

터를 포함한다.

[0127] 예 11은 예 1 내지 예 8 중 어느 한 예의 프로세서를 포함하며, 이 예에서 디코드 유닛은 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소가, 제 1 소스 패킹된 데이터 피연산자 내의 데이터 요소가 임의로 특정하게 배열되는 것과 상관없이 그리고 제 2 소스 피연산자 내의 마스크 요소가 임의로 특정하게 배열되는 것과 상관없이, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는 것을 표시하는 오피코드를 갖는 명령어를 디코딩한다.

[0128] 예 12는 예 1 내지 예 8 중 어느 한 예의 프로세서를 포함한다. 또한 선택적으로 이 예에서, 프로세서는 범용 프로세서를 포함하며, 목적지 저장 위치는 프로세서의 패킹된 데이터 레지스터를 포함한다.

[0129] 예 13은 데이터 요소 선택 및 통합 명령어를 수신하는 단계를 포함하는 프로세서에서의 방법이다. 데이터 요소 선택 및 통합 명령어는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자와, 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는다. 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응한다. 방법은 또한 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자를 목적지 저장 위치에 저장하는 단계를 포함한다. 목적지 저장 위치는 데이터 요소 선택 및 통합 명령어에 의해 표시된다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함한다.

[0130] 예 14는 예 13의 방법을 포함하며, 이 예에서 수신하는 단계는 마스크 요소를 갖는 제 2 소스 피연산자로서 이미디어트를 갖는 명령어를 수신하는 단계를 포함한다.

[0131] 예 15는 예 13의 방법을 포함하며, 이 예에서 수신하는 단계는 예측에 사용되는 한 세트의 전용의 패킹된 데이터 연산 마스크 레지스터 중의 패킹된 데이터 연산 마스크 레지스터인 제 2 소스 피연산자를 갖는 명령어를 수신하는 단계를 포함한다.

[0132] 예 16은 예 13의 방법을 포함하며, 이 예에서 마스크 요소로서 마스크 비트를 갖는 제 2 소스 피연산자에 액세스하는 단계를 더 포함한다.

[0133] 예 17은 예 13의 방법을 포함하며, 이 예에서 수신하는 단계는 제 1 패킹된 데이터 피연산자의 데이터 요소의 크기를 표시하는 하나 이상의 비트를 갖는 명령어를 수신하는 단계를 포함한다.

[0134] 예 18은 예 13의 방법을 포함하며, 이 예에서 수신하는 단계는 적어도 128-비트를 갖고 8-비트 데이터 요소 및 16-비트 데이터 요소 중 하나인 데이터 요소를 포함하는 제 1 소스 패킹된 데이터 피연산자를 표시하는 명령어를 수신하는 단계를 포함한다. 또한 선택적으로 이 예에서, 저장하는 단계는 결과 패킹된 데이터 피연산자의 최상위 부분에서 함께 통합된 모든 데이터 요소를 갖는 결과 패킹된 데이터 피연산자를 모든 데이터 요소가 제 1 패킹된 데이터 피연산자에서 출현하는 것과 동일한 순서로 저장하는 단계를 포함한다.

[0135] 예 19는 예 13의 방법을 포함하며, 네트워크로부터 패킷을 수신하는 단계와, 패킷의 일부분을 제 1 소스 패킹된 데이터 피연산자로서 저장하는 단계를 더 포함한다. 방법은 또한 패킷의 프로토콜을 디코딩하는 단계와, 선택적으로 패킷의 프로토콜을 디코딩한 것에 기초하여 패킷의 일부분에 있는 플로우 바이트의 위치를 결정하는 단계를 선택적으로 포함할 수 있다. 방법은 또한 플로우 바이트 각각에 대해 제 2 소스 피연산자 내의 마스킹하지 않는 마스크 요소를 저장하고 패킷의 일부분 내의 다른 바이트에 대해 제 2 소스 피연산자 내의 마스킹하는 요소를 저장하는 단계를 선택적으로 포함할 수 있다.

[0136] 예 20은 예 13의 방법을 포함하며, 이 예에서 제 1 소스 패킹된 데이터 피연산자는 네트워크로부터 수신된 패킷의 데이터 요소를 갖는다. 선택적으로 방법은 또한 결과 패킹된 데이터 피연산자의 통합된 데이터 요소에 대해 암호화 동작을 수행하는 단계를 포함할 수 있다.

[0137] 예 21은 인터커넥트와, 인터커넥트와 연결된 프로세서를 포함하는 명령어를 처리하는 시스템이다. 프로세서는 복수의 데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자를 갖는 데이터 요소 선택 및 통합 명령어를 수신한다. 명령어는 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는다. 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응한다. 명령어는 또한 목적지 저장 위치를 표시한다. 프로세서는 데이터 요소 선택 및 통합 명령어에 응답하여, 결과 패킹된 데이터 피연산자를 목적지 저장 위치에 저장한다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹

된 데이터 피연산자의 모든 데이터 요소를 포함한다. 시스템은 또한 인터커넥트와 연결된 다이나믹 랜덤 액세스 메모리(DRAM)을 포함한다. DRAM은 한 세트의 프로토콜 디코드 명령어를 저장하며, 명령어 세트는 프로세서에 의해 실행될 때, 프로세서로 하여금, 제 1 네트워크 패킷의 프로토콜을 디코딩하고, 제 2 소스 피연산자를 생성하여 제 1 소스 패킹된 데이터 피연산자에 저장된 제 2 네트워크 패킷의 헤더의 적어도 일부분 내의 플로우 요소를 마스킹하지 않고 비 플로우 요소를 마스킹하는 동작을 수행하게 한다.

[0138] 예 22는 예 21의 시스템을 포함하며, 이 예에서 제 2 소스 피연산자는 이미디어트를 포함한다.

[0139] 예 23은 비일시적 머신 판독가능한 저장 매체를 포함하는 제조 물품이다. 비일시적 머신 판독가능한 저장 매체는 데이터 요소 선택 및 통합 명령어를 포함한다. 명령어는 복수의 데이터를 갖는 제 1 소스 패킹된 데이터 피연산자를 갖고, 복수의 마스크 요소를 갖는 제 2 소스 피연산자를 갖는다. 제 2 소스 피연산자의 각 마스크 요소는 동일한 상대적 위치에 있는 제 1 소스 패킹된 데이터 피연산자의 상이한 데이터 요소에 대응한다. 명령어는 목적지 저장 위치를 표시한다. 명령어는 머신에 의해 실행되는 경우 머신으로 하여금 결과 패킹된 데이터 피연산자를 목적지 저장 위치에 저장하는 것을 포함하는 동작을 수행하게 한다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않는 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함한다.

[0140] 예 24는 예 23의 제조 물품을 포함하며, 이 예에서 제 2 소스 피연산자는 예측에 사용되는 한 세트의 전용의 패킹된 데이터 연산 마스크 레지스터 중의 패킹된 데이터 연산 마스크 레지스터이다.

[0141] 예 25는 예 13 내지 예20 중 어느 한 예의 방법을 수행하도록 동작하는 프로세서 또는 다른 장치이다.

[0142] 예 26은 예 13 내지 예20 중 어느 한 예의 방법을 수행하기 위한 수단을 포함하는 프로세서 또는 다른 장치이다.

[0143] 예 27은 예 13 내지 예20 중 어느 한 예의 방법을 수행하는 모듈을 포함하는 프로세서 또는 다른 장치이다.

[0144] 예 28은 예 13 내지 예20 중 어느 한 예의 방법을 수행하기 위한 모듈 및/또는 유닛 및/또는 로직 및/또는 회로 및/또는 수단의 임의의 조합을 포함하는 프로세서이다.

[0145] 예 29는 명령어를 선택적으로 저장하거나 그렇지 않으면 제공하는 선택적으로 비일시적 머신 판독 가능한 매체를 포함하는 제조 물품이며, 명령어는 프로세서, 컴퓨터 시스템, 전자 디바이스, 또는 다른 머신에 의해 실행되면 그리고/또는 실행될 때, 머신으로 하여금 예 13 내지 예 20 중 어느 한 예의 방법을 수행하게 하도록 작용한다.

[0146] 예 30은 버스 또는 다른 인터커넥트를 포함하는 컴퓨터 시스템, 다른 전자 디바이스, 또는 다른 장치이며, 예 1 내지 예 12 중 어느 한 예의 프로세서는 인터커넥트에 연결되고, 인터커넥트에는 다이나믹 랜덤 액세스 메모리(DRAM), 네트워크 인터페이스, 그래픽스 칩, 무선 통신 칩, 세계 이동통신 시스템(Global System for Mobile Communications, GSM) 안테나, 상변화 메모리, 및 비디오 카메라로부터 선택되는 적어도 하나의 컴포넌트가 연결된다.

[0147] 예 31은 실질적으로 본 명세서에서 기술된 바와 같은 프로세서 또는 다른 장치이다.

[0148] 예 32는 실질적으로 본 명세서에서 기술된 바와 같은 임의의 방법을 수행하도록 동작하는 프로세서 또는 다른 장치이다.

[0149] 예 33은 실질적으로 본 명세서에서 기술된 바와 같은 임의의 데이터 요소 선택 및 압축 명령어를 수행하도록 동작하는 프로세서 또는 다른 장치이다.

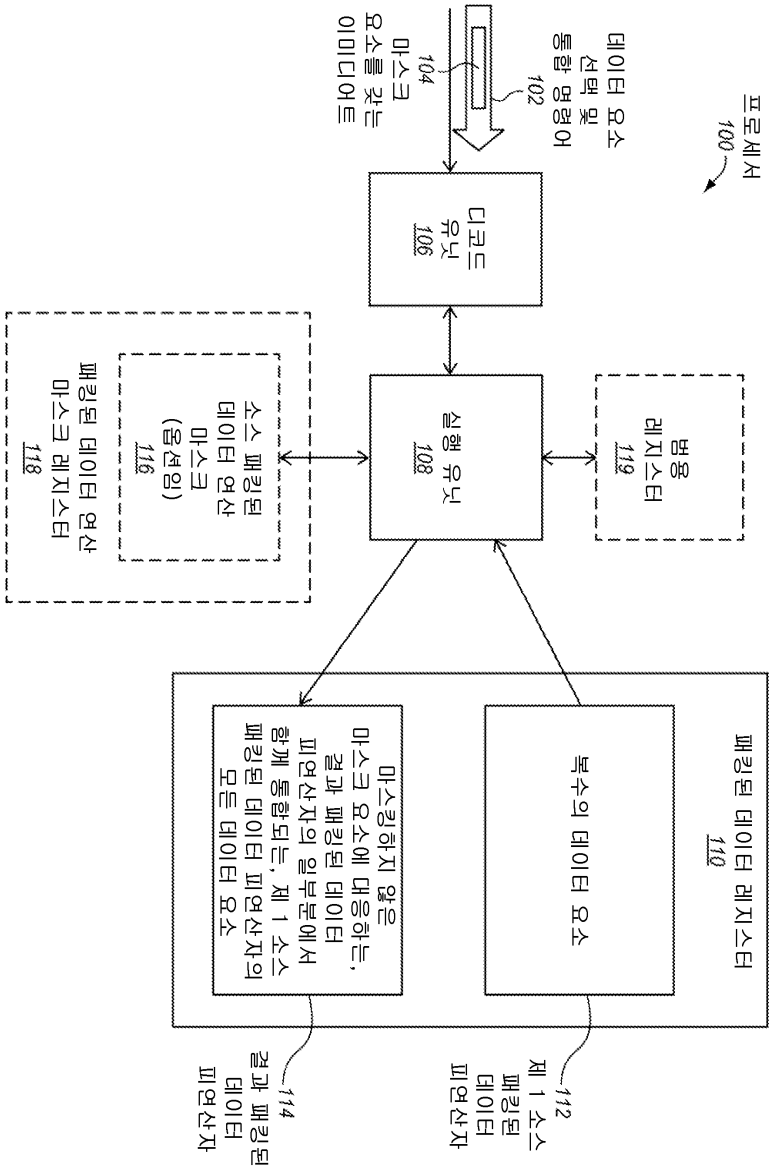
[0150] 예 34는 제 1 명령어 집합의 명령어를 디코딩하는 디코드 유닛을 포함하는 프로세서 또는 다른 장치이다. 디코드 유닛은 제 1 명령어를 에뮬레이트하는 제 1 명령어 집합 중의 하나 이상의 명령어를 수신한다. 제 1 명령어는 실질적으로 본 명세서에서 개시된 바와 같은 임의의 데이터 요소 선택 및 압축 명령어일 수 있으며, 상이한 제 2 명령어 집합을 갖는다. 프로세서 또는 다른 장치는 또한 디코드 유닛과 연결되어 제 1 명령어 집합 중의 하나 이상의 명령어를 실행하는 하나 이상의 실행 유닛을 포함한다. 하나 이상의 실행 유닛은 제 1 명령어 집합 중의 하나 이상의 명령어에 응답하여, 결과를 목적지에 저장한다. 결과는 실질적으로 본 명세서에서 개시된 바와 같은 데이터 요소 선택 및 압축 명령어의 임의의 결과일 수 있다.

[0151] 예 35는 제 1 명령어 집합의 명령어를 디코딩하는 디코드 유닛을 갖는 프로세서를 포함하는 컴퓨터 시스템 또는 다른 전자 디바이스이다. 프로세서는 또한 하나 이상의 실행 유닛을 갖는다. 전자 디바이스는 또한 프로세서와

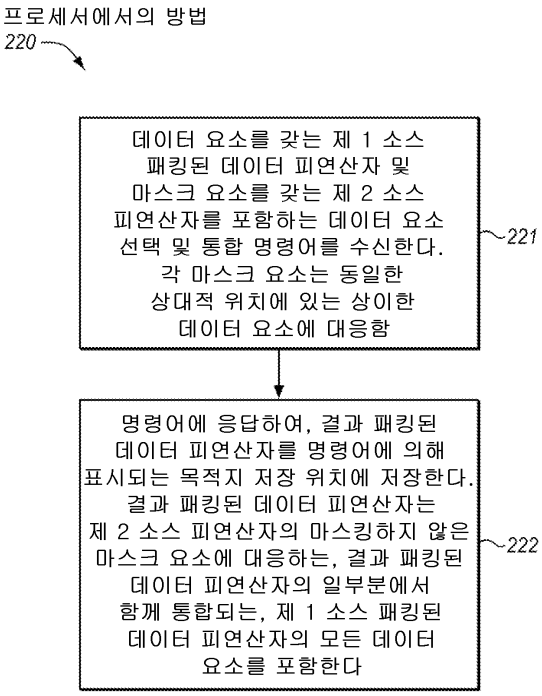
연결된 저장 디바이스를 포함한다. 저장 디바이스는 실질적으로 본 명세서에서 개시된 바와 같은 임의의 데이터 요소 선택 및 압축 명령어일 수 있으며, 상이한 제 2 명령어 집합을 갖는 제 1 명령어를 저장한다. 저장 디바이스는 또한 제 1 명령어를 제 1 명령어 집합 중의 하나 이상의 명령어로 변환하는 명령어를 저장한다. 제 1 명령어 집합 중의 하나 이상의 명령어는 프로세서에 의해 수행될 때, 프로세서로 하여금 결과를 목적지에 저장하게 한다. 결과는 실질적으로 본 명세서에서 개시된 바와 같은 데이터 요소 선택 및 압축 명령어의 임의의 결과를 포함할 수 있다.

도면

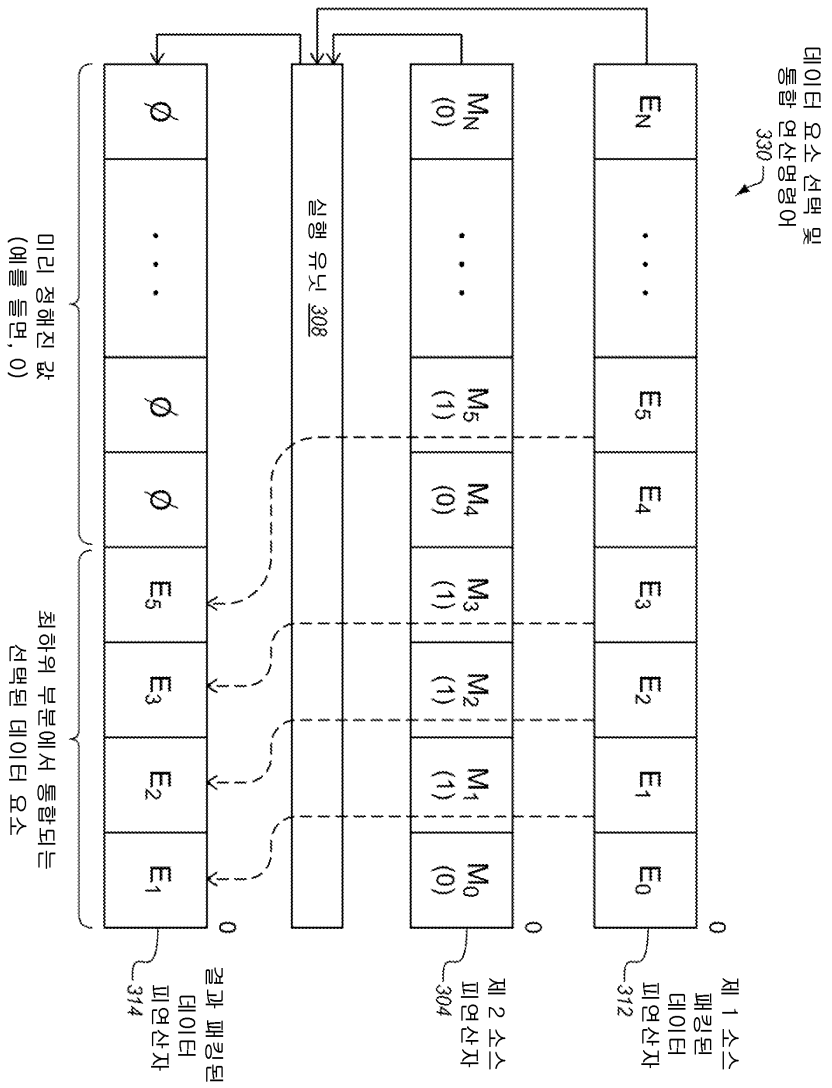
도면1



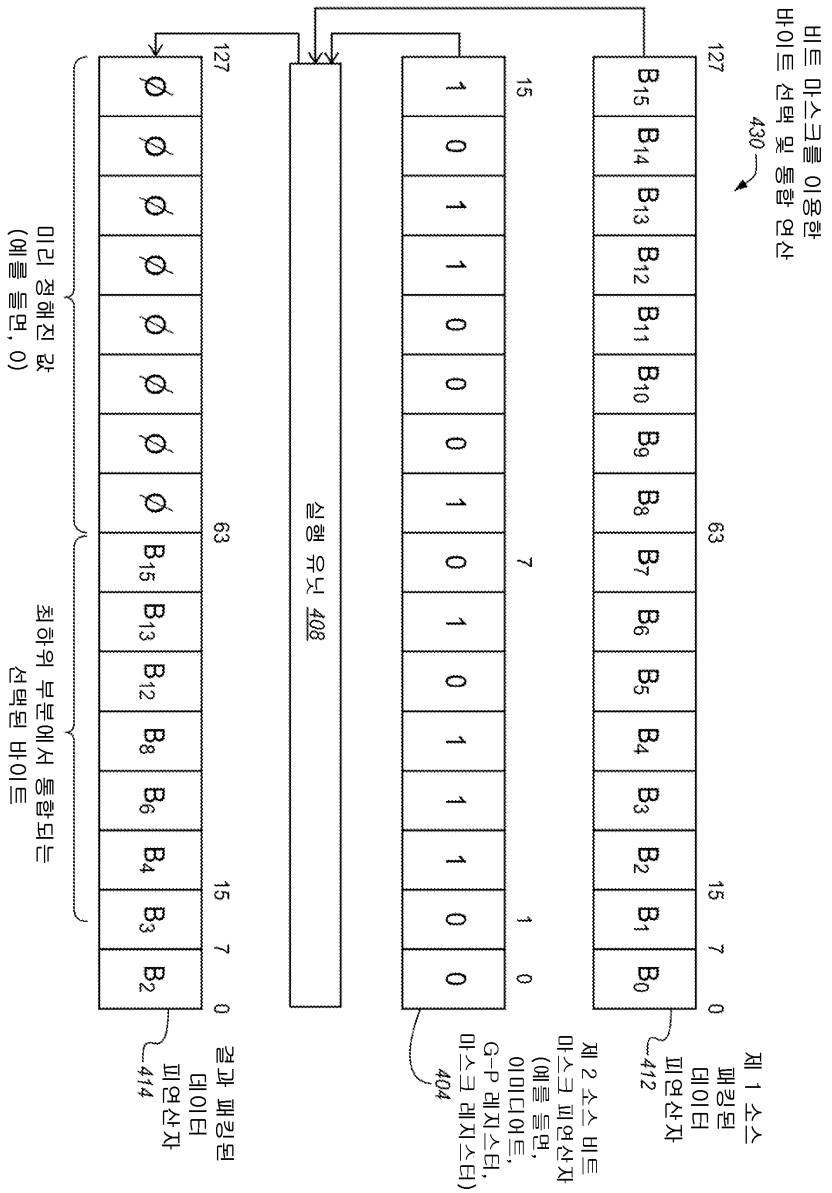
도면2


프로세서에서의 방법
220
데이터 요소를 갖는 제 1 소스 패킹된 데이터 피연산자 및 마스크 요소를 갖는 제 2 소스 피연산자를 포함하는 데이터 요소 선택 및 통합 명령어를 수신한다. 각 마스크 요소는 동일한 상대적 위치에 있는 상이한 데이터 요소에 대응함
221
명령어에 응답하여, 결과 패킹된 데이터 피연산자를 명령어에 의해 표시되는 목적지 저장 위치에 저장한다. 결과 패킹된 데이터 피연산자는 제 2 소스 피연산자의 마스킹하지 않은 마스크 요소에 대응하는, 결과 패킹된 데이터 피연산자의 일부분에서 함께 통합되는, 제 1 소스 패킹된 데이터 피연산자의 모든 데이터 요소를 포함한다
222

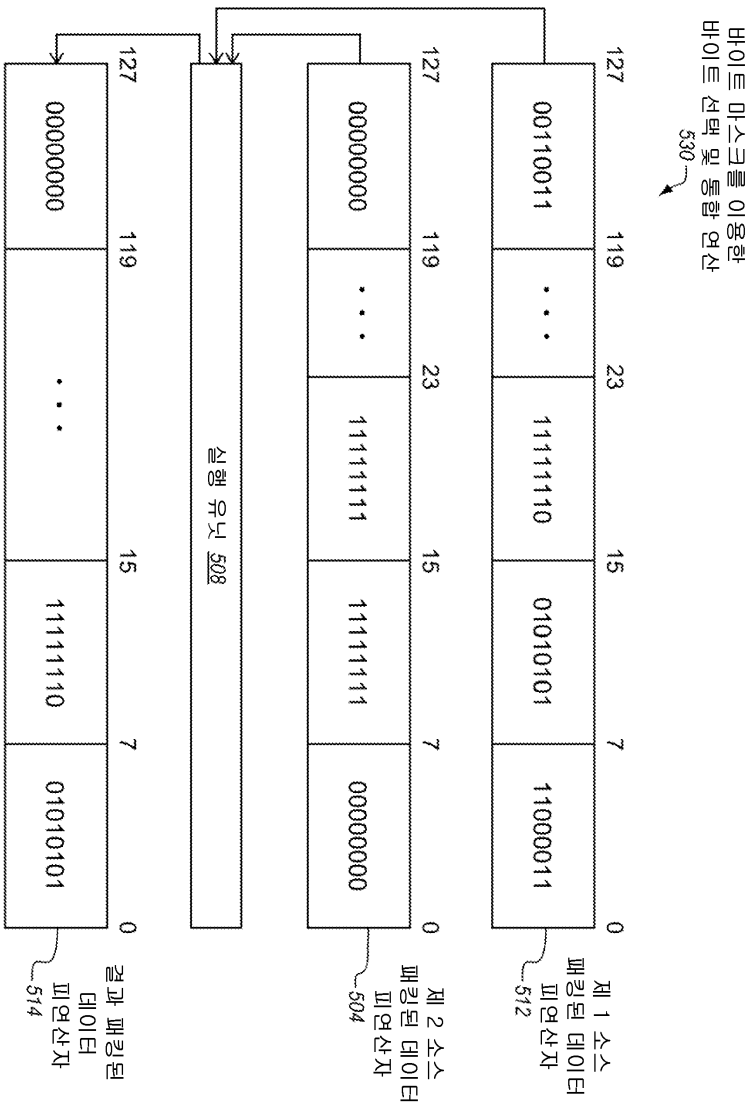
도면3


데이터 요소 선택 및 통합 연산명령어
330
실행 유닛 308
미리 정해진 값 (예를 들면, 0)
최하위 부분에서 통합되는 선택된 데이터 요소
제 1 소스 패킹된 데이터 피연산자
312
제 2 소스 피연산자
304
결과 패킹된 데이터 피연산자
314
0
0
0

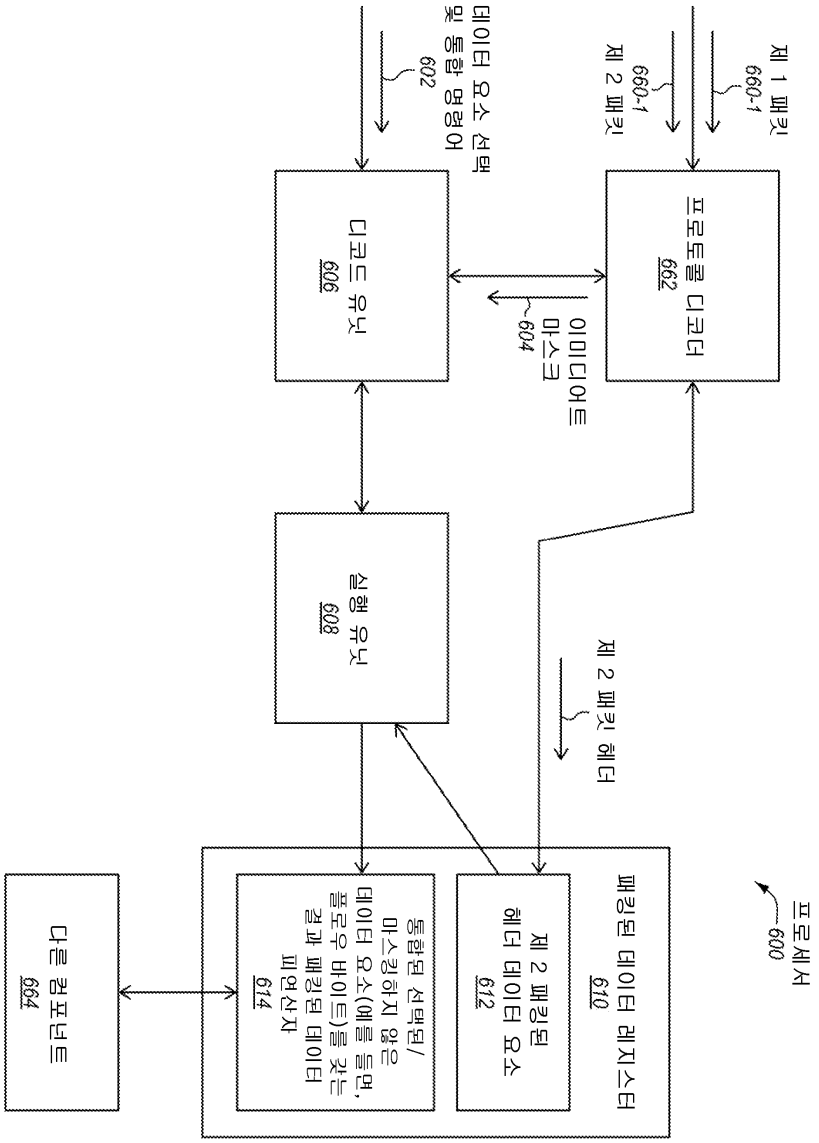
도면4


비트 마스크를 이용한
바이트 선택 및 통합 연산
430
127
63
15
7
0
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
제 1 소스
패킹된
데이터
피연산자
412
1 0 1 1 0 0 0 1 0 1 0 1 1 1 0 0
제 2 소스 비트
마스크 피연산자
(예를 들면,
이미디어트,
G-P 레지스터,
마스크 레지스터)
404
실행 유닛 408
Ø Ø Ø Ø Ø Ø Ø Ø B15 B13 B12 B8 B6 B4 B3 B2
미리 정해진 값
(예를 들면, 0)
최하위 부분에서 통합되는
선택된 바이트
결과 패킹된
데이터
피연산자
414

도면5


바이트 마스크를 이용한 바이트 선택 및 통합 연산
530
127
119
23
15
7
0
00110011
• • •
11111110
01010101
11000011
제 1 소스 패킹된 데이터 피연산자
512
00000000
• • •
11111111
11111111
00000000
제 2 소스 패킹된 데이터 피연산자
504
실행 유닛 508
00000000
• • •
11111110
01010101
결과 패킹된 데이터 피연산자
514

도면6


제 1 패킷
660-1
제 2 패킷
660-1
데이터 요소 선택
및 동작 명령어
602
프로토콜 디코더
662
디코드 유닛
606
이미디어트
마스크
604
실행 유닛
608
제 2 패킷 헤더
패킹된 데이터 레지스터
610
제 2 패킹된
헤더 데이터 요소
612
동작된 선택된/
마스킹하지 않은
데이터 요소(예를 들면,
플로우 바이트)를 갖는
결과 패킹된 데이터
피연산자
614
다른 컴포넌트
664
프로세서
600

도면7

데이터 요소 선택 및 통합 명령어 702A

오피코드 740A	제 1 소스 패킹된 데이터 피연산자 지정자 742A	목적지 지정자 (옵션임) 744A	이미디어트 704	데이터 요소 크기 지정자 (옵션임) 746A

(a)

데이터 요소 선택 및 통합 명령어 702B

오피코드 740B	제 1 소스 패킹된 데이터 피연산자 지정자 742B	목적지 지정자 (옵션임) 744B	패킹된 데이터 연산 마스크 지정자 748	데이터 요소 크기 지정자 (옵션임) 746B

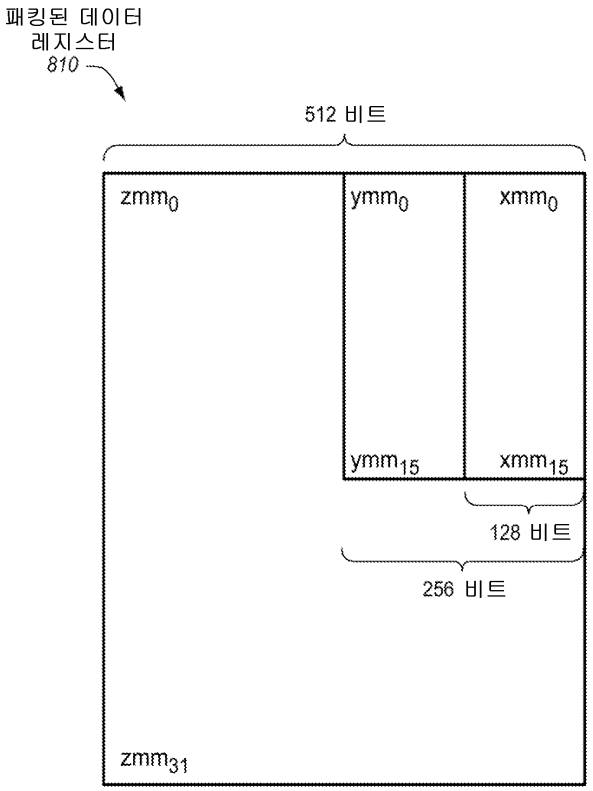
(b)

데이터 요소 선택 및 통합 명령어 702C

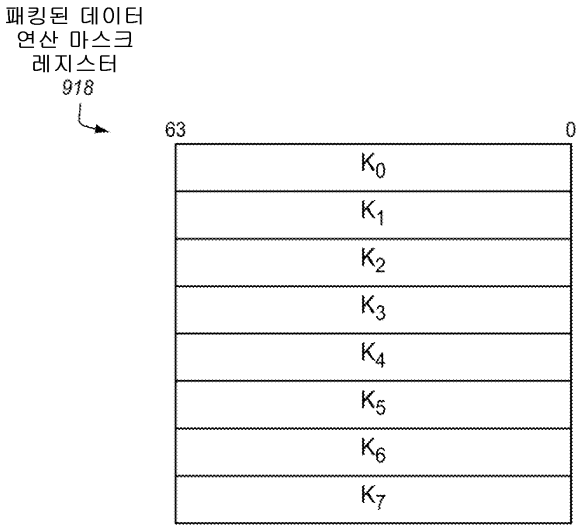
오피코드 740C	제 1 소스 패킹된 데이터 피연산자 지정자 742C	목적지 지정자 (옵션임) 744C	제 2 소스 패킹된 데이터 피연산자 지정자 750	데이터 요소 크기 지정자 (옵션임) 746C

(c)

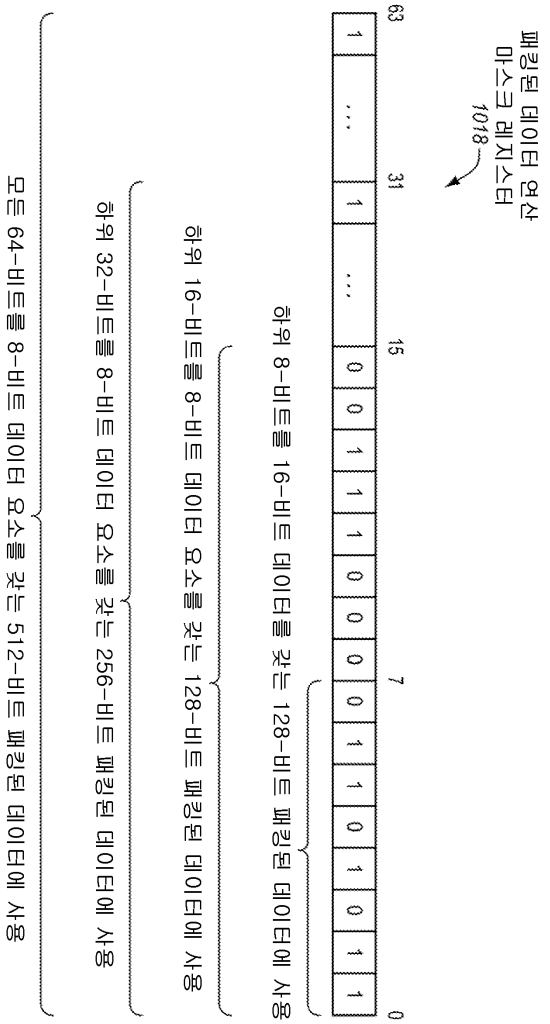
도면8


패킹된 데이터
레지스터
810
512 비트
zmm0
ymm0
xmm0
ymm15
xmm15
128 비트
256 비트
zmm31

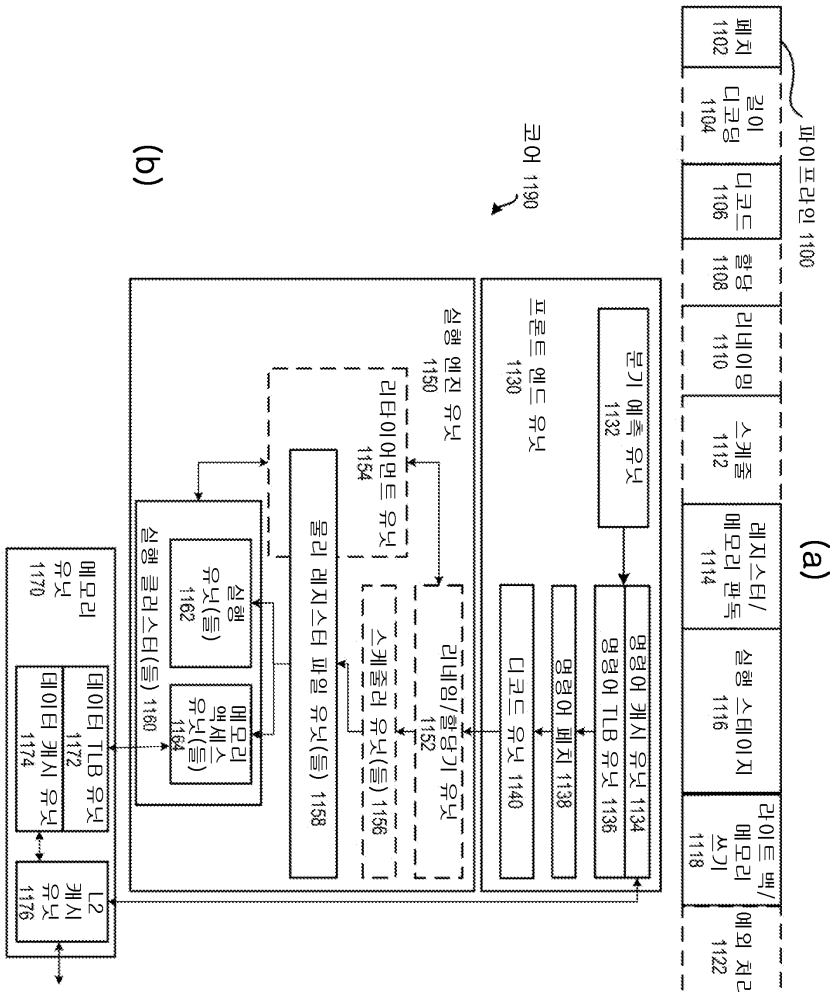
도면9


패킹된 데이터
연산 마스크
레지스터
918
63
0
K0
K1
K2
K3
K4
K5
K6
K7

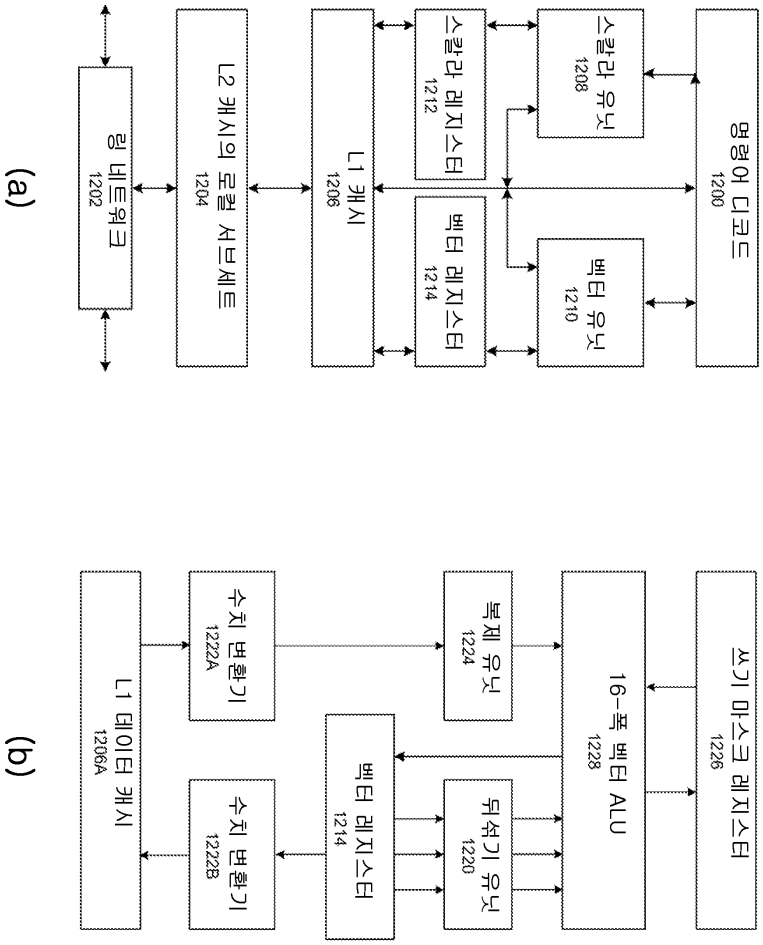
도면10


패킹된 데이터 연산
마스크 레지스터
1018
63
31
15
7
0
1
...
1
...
0
0
1
1
1
0
0
0
0
1
1
1
0
1
0
1
1
하위 8-비트를 16-비트 데이터를 갖는 128-비트 패킹된 데이터에 사용
하위 16-비트를 8-비트 데이터 요소를 갖는 128-비트 패킹된 데이터에 사용
하위 32-비트를 8-비트 데이터 요소를 갖는 256-비트 패킹된 데이터에 사용
모든 64-비트를 8-비트 데이터 요소를 갖는 512-비트 패킹된 데이터에 사용

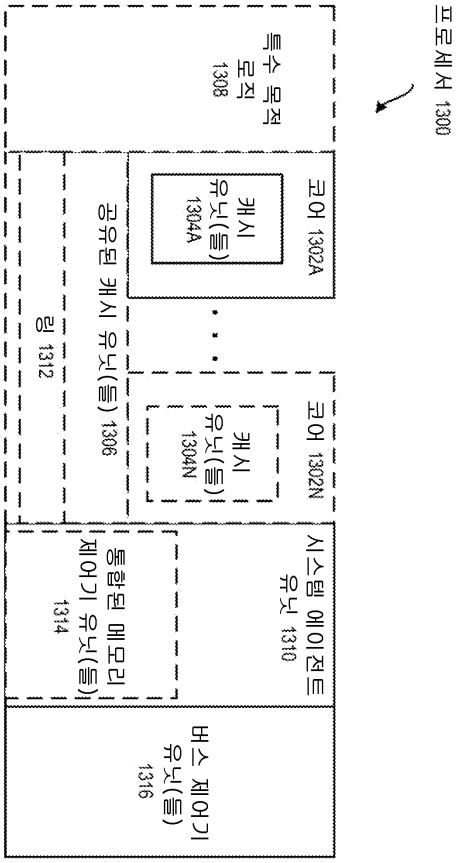
도면11


(a)
파이프라인 1100
페치 1102
길이 디코딩 1104
디코드 1106
할당 1108
리네이밍 1110
스케줄 1112
레지스터/메모리 판독 1114
실행 스테이지 1116
라이트 백/메모리 쓰기 1118
예외 처리 1122
커밋 1124
코어 1190
분기 예측 유닛 1132
명령어 캐시 유닛 1134
명령어 TLB 유닛 1136
명령어 페치 1138
디코드 유닛 1140
프론트 엔드 유닛 1130
(b)
실행 엔진 유닛 1150
리네임/할당기 유닛 1152
리타이어먼트 유닛 1154
스케줄러 유닛(들) 1156
물리 레지스터 파일 유닛(들) 1158
실행 클러스터(들) 1160
실행 유닛(들) 1162
메모리 엑세스 유닛(들) 1164
메모리 유닛 1170
데이터 TLB 유닛 1172
데이터 캐시 유닛 1174
L2 캐시 유닛 1176

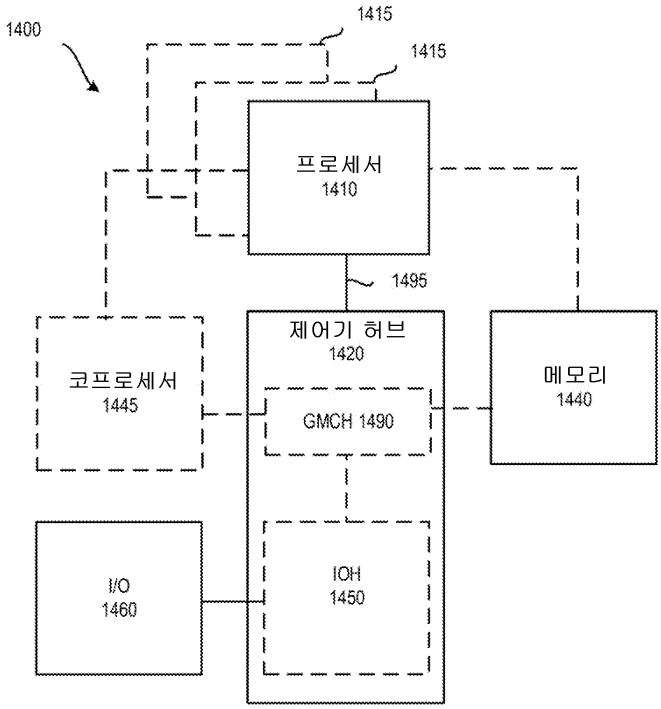
도면12


(a)
링 네트워크
1202
L2 캐시의 로컬 서브세트
1204
L1 캐시
1206
스칼라 레지스터
1212
벡터 레지스터
1214
스칼라 유닛
1208
벡터 유닛
1210
명령어 디코드
1200
(b)
L1 데이터 캐시
1206A
수치 변환기
1222A
수치 변환기
1222B
벡터 레지스터
1214
복제 유닛
1224
뒤섞기 유닛
1220
16-폭 벡터 ALU
1228
쓰기 마스크 레지스터
1226

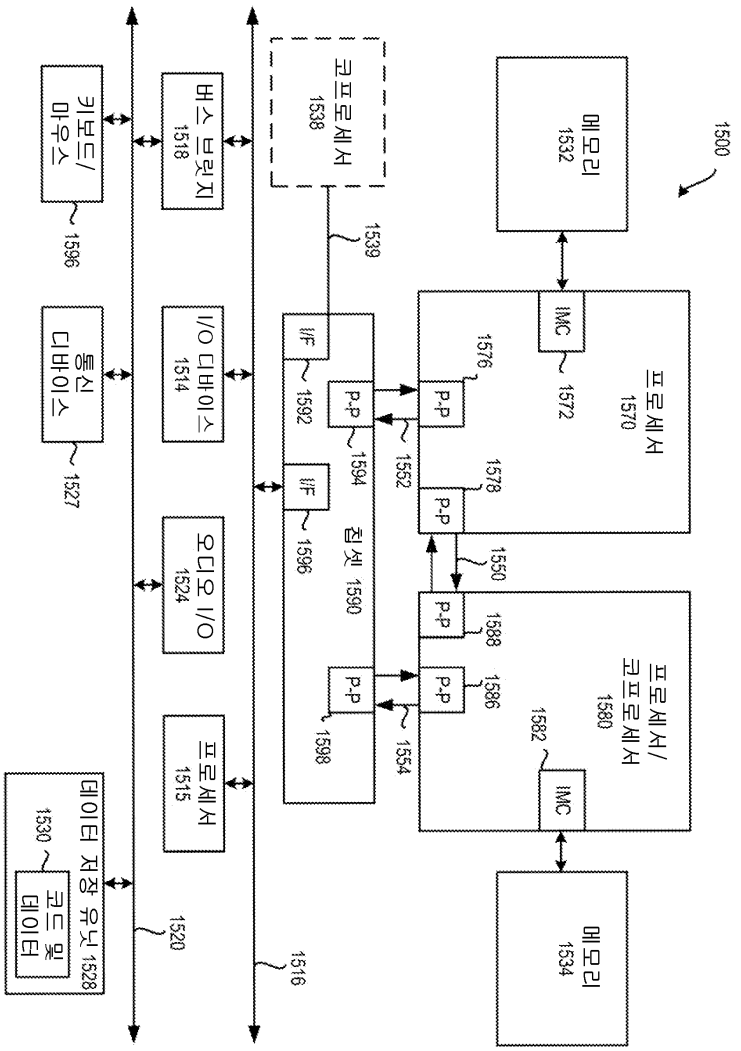
도면13


프로세서 1300
특수 목적 로직 1308
코어 1302A
캐시 유닛(들) 1304A
코어 1302N
캐시 유닛(들) 1304N
공유된 캐시 유닛(들) 1306
링 1312
시스템 에이전트 유닛 1310
통합된 메모리 제어기 유닛(들) 1314
버스 제어기 유닛(들) 1316

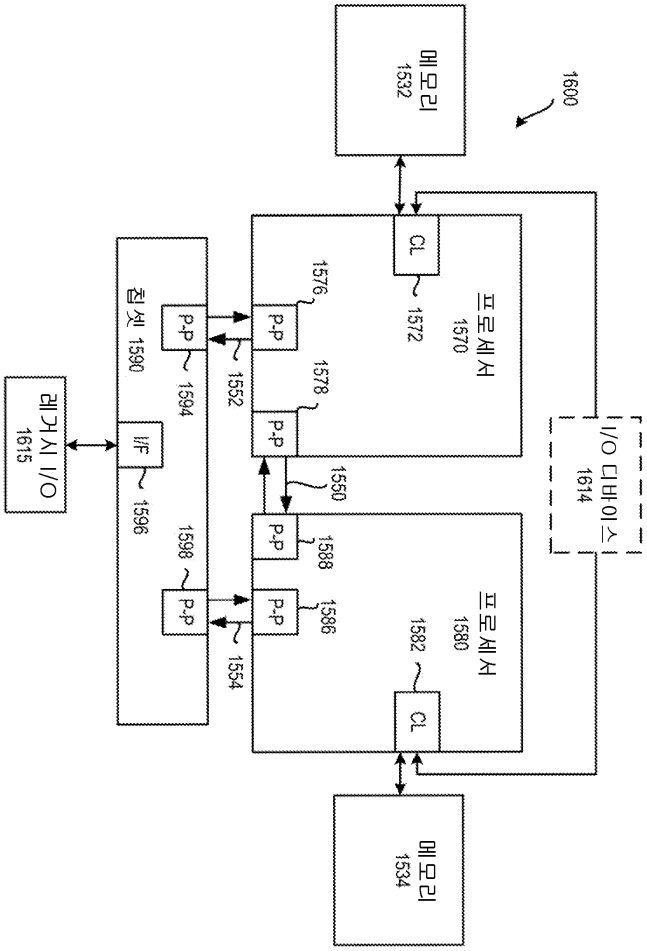
도면14


1400
1415
1415
프로세서 1410
1495
제어기 허브 1420
GMCH 1490
IOH 1450
코프로세서 1445
메모리 1440
I/O 1460

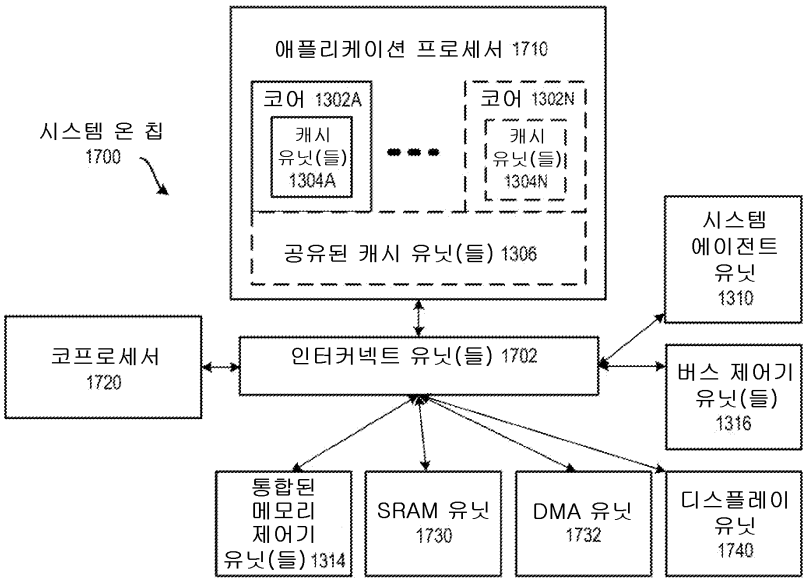
도면15


1500
메모리
1532
IMC
1572
프로세서
1570
P-P
1576
P-P
1578
1552
1550
P-P
1588
P-P
1586
프로세서/
코프로세서
1580
IMC
1582
메모리
1534
1554
I/F
1592
P-P
1594
I/F
1596
칩셋 1590
P-P
1598
1539
코프로세서
1538
키보드/
마우스
1596
버스 브릿지
1518
통신
디바이스
1527
I/O 디바이스
1514
오디오 I/O
1524
프로세서
1515
데이터 저장 유닛 1528
코드 및
데이터
1530
1520
1516

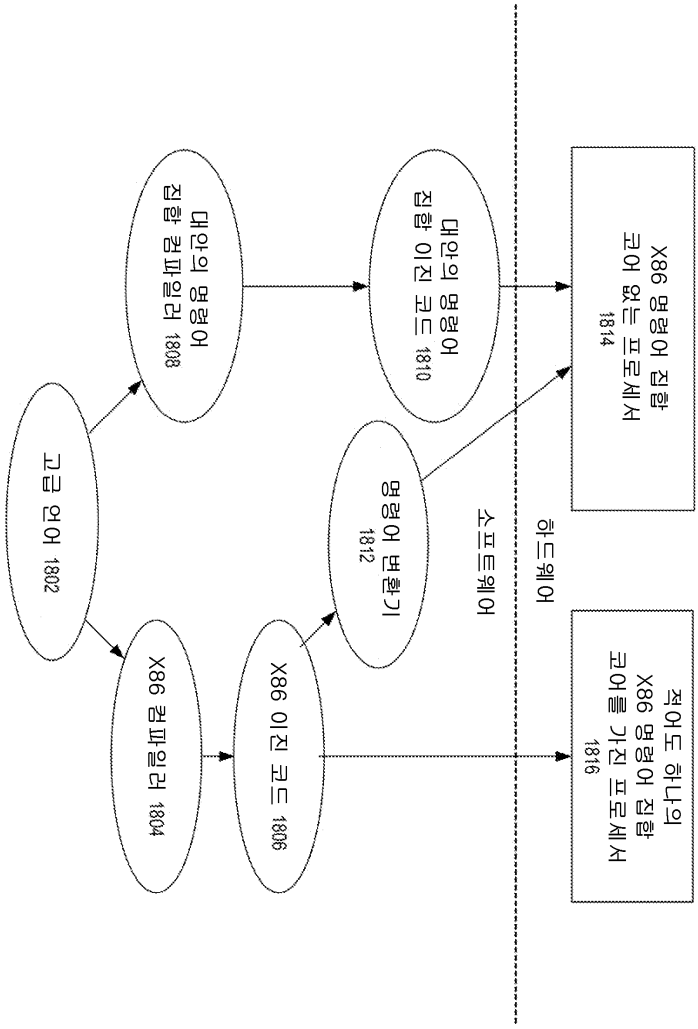
도면16


메모리
1532
1600
CL
1572
프로세서
1570
1576
P-P
1578
P-P
1552
P-P
1594
칩셋 1590
I/F
1596
1598
P-P
레거시 I/O
1615
1550
P-P
1588
P-P
1586
1554
프로세서
1580
1582
CL
I/O 디바이스
1614
메모리
1534

도면17


애플리케이션 프로세서 1710
코어 1302A
캐시 유닛(들) 1304A
코어 1302N
캐시 유닛(들) 1304N
공유된 캐시 유닛(들) 1306
시스템 온 칩 1700
시스템 에이전트 유닛 1310
코프로세서 1720
인터커넥트 유닛(들) 1702
버스 제어기 유닛(들) 1316
통합된 메모리 제어기 유닛(들) 1314
SRAM 유닛 1730
DMA 유닛 1732
디스플레이 유닛 1740

도면18


고급 언어 1802
X86 컴파일러 1804
대안의 명령어 집합 컴파일러 1808
X86 이진 코드 1806
대안의 명령어 집합 이진 코드 1810
명령어 변환기 1812
소프트웨어
하드웨어
X86 명령어 집합 코어 없는 프로세서 1814
적어도 하나의 X86 명령어 집합 코어를 가진 프로세서 1816