(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0025488
(43) 공개일자 2025년02월21일

(51) 국제특허분류(Int. Cl.)
G06F 9/30 (2018.01)
(52) CPC특허분류
G06F 9/30018 (2023.08)
G06F 9/30032 (2013.01)
(21) 출원번호 10-2025-7002564
(22) 출원일자(국제) 2023년06월14일
심사청구일자 없음
(85) 번역문제출일자 2025년01월23일
(86) 국제출원번호 PCT/GB2023/051544
(87) 국제공개번호 WO 2024/003526
국제공개일자 2024년01월04일
(30) 우선권주장
2209637.4 2022년06월30일 영국(GB)

(71) 출원인
에이알엠 리미티드
영국 캠브리지 씨비1 9엔제이 체리힌톤 폴번로드 110
(72) 발명자
그로쿠트, 토마스 크리스토퍼
영국 씨비1 9엔제이 캠브릿지셔 캠브릿지 폴번 로드 110 씨/오 에이알엠 리미티드
(74) 대리인
이화익

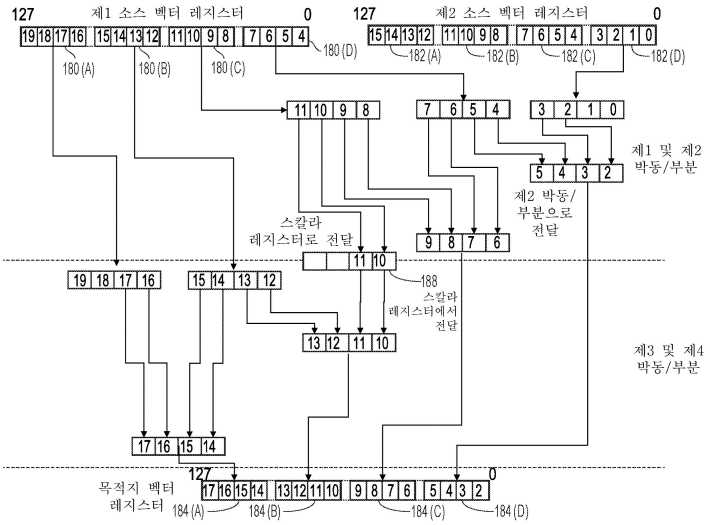
전체 청구항 수 : 총 24 항

(54) 발명의 명칭 **벡터 추출 및 병합 명령어**

(57) 요 약

장치, 방법 및 매체가 제공된다. 장치는 제어 파라미터, 제1 벡터 레지스터, 제2 벡터 레지스터, 및 목적지 벡터 레지스터를 명시하는 벡터 추출 및 병합 명령어에 응답하여 제어 신호를 생성하는 디코더 회로부를 포함한다. 장치는 제어 신호에 응답하여, 복수의 박동 프로세싱을 수행하기 위한 프로세싱 회로부를 포함하고, 각각의 박동은 적어도 제1 벡터 레지스터 및 목적지 벡터 레지스터의 부분에 해당하는 프로세싱을 포함한다. 명시된 레지스터의 각각의 K번째 부분에 해당하는 K번째 박동에 대한 프로세싱은, 제어 파라미터에 의해 명시된 비트를 제1 벡터 레지스터의 K번째 부분으로부터 추출하는 것, 비트를 추가 비트와 연결하는 것, 및 결과를 목적지 레지스터의 K번째 부분에 저장하는 것을 포함한다. 추가 비트는, 제1 부분의 경우, 제2 벡터 레지스터의 제1 부분으로부터 추출되고, 그렇지 않으면, 제1 벡터 레지스터의 (K-1)번째 부분으로부터 추출된다.

대 표 도 - 도16



(52) CPC특허분류
G06F 9/30036 (2023.08)
G06F 9/30098 (2013.01)

명 세 서

청구범위

청구항 1

장치로서,

복수의 벡터 레지스터;

제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 회로부; 및

상기 제어 신호에 응답하여 복수의 박동 프로세싱을 수행하는 프로세싱 회로부를 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 회로부는 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

상기 제어 파라미터에 의해 명시된 바와 같이, 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 비트를 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 상기 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달되는, 장치.

청구항 2

제1항에 있어서,

상기 디코더 회로부는 스칼라 레지스터를 명시하는 상기 벡터 추출 및 병합 명령어에 응답하며;

상기 복수의 박동은 현재 실행 중인 하나 이상의 박동의 서브세트를 포함하되, 상기 현재 실행 중인 박동의 서브세트는 상기 완료된 박동을 제외하고;

상기 프로세싱 회로부는 상기 제어 신호에 응답하여, 상기 스칼라 레지스터에 캐리 데이터의 적어도 하나의 아이템을 저장하고, 캐리 데이터의 상기 적어도 하나의 아이템은 상기 현재 실행 중인 하나 이상의 박동의 서브세트와 상기 복수의 박동 중 하나 이상의 박동의 추가 서브세트 사이에서 전달될 하나 이상의 비트를 포함하는, 장치.

청구항 3

제2항에 있어서, 상기 프로세싱 회로부는 상기 현재 실행 중인 하나 이상의 박동의 세트의 제1 박동에 대해 상기 제어 신호에 응답하고, 상기 벡터 추출 및 병합 명령어의 실행 전의 상기 박동 상태 정보가 적어도 하나의 박동이 억제되어야 한다는 것을 표시할 때, 상기 스칼라 레지스터로부터 상기 하나 이상의 추가 비트를 검색하는, 장치.

청구항 4

제3항에 있어서,

전달될 상기 하나 이상의 비트는 상기 제1 소스 벡터 레지스터의 부분의 모든 비트를 포함하고;

상기 스칼라 레지스터로부터 상기 하나 이상의 추가 비트를 검색하는 것은 상기 스칼라 레지스터로부터 마지막 비트의 서브세트를 검색하는 것을 포함하는, 장치.

청구항 5

제3항에 있어서,

전달될 상기 하나 이상의 비트는 상기 스칼라 레지스터의 임시 비트 위치의 세트에 저장된 상기 제1 소스 벡터 레지스터의 부분으로부터의 마지막 M 비트의 세트를 포함하고;

상기 스칼라 레지스터로부터 상기 하나 이상의 추가 비트를 검색하는 것은 상기 스칼라 레지스터의 상기 임시 비트 위치의 세트로부터 비트를 검색하는 것을 포함하는, 장치.

청구항 6

제1항 내지 제5항 중 어느 한 항에 있어서, 상기 추출된 비트를 연결하는 것은 상기 목적지 레지스터의 상기 K번째 부분의 제1 연속적인 비트 위치의 세트에 상기 추출된 비트를 저장하고 상기 목적지 레지스터의 상기 K번째 부분의 제2 연속적인 비트 위치의 세트에 상기 하나 이상의 추가 비트를 저장하는 것을 포함하는, 장치.

청구항 7

제6항에 있어서, 상기 제1 연속적인 비트 위치의 세트 및 상기 제2 연속적인 비트 위치의 세트는 비-중첩 비트 위치인, 장치.

청구항 8

제6항 또는 제7항에 있어서, 상기 제1 연속적인 비트 위치의 세트는 상기 목적지 레지스터의 상기 K번째 부분의 최상위 비트 위치의 세트이고 상기 제2 연속적인 비트 위치의 세트는 상기 목적지 레지스터의 상기 K번째 부분의 최하위 비트 위치의 세트인, 장치.

청구항 9

제6항 또는 제7항에 있어서, 상기 제1 연속적인 비트 위치의 세트는 상기 목적지 레지스터의 상기 K번째 부분의 최하위 비트 위치의 세트이고 상기 제2 연속적인 비트 위치의 세트는 상기 목적지 레지스터의 상기 K번째 부분의 최상위 비트 위치의 세트인, 장치.

청구항 10

제1항 내지 제9항 중 어느 한 항에 있어서, 상기 추출된 비트는 상기 제1 소스 벡터 레지스터의 상기 K번째 부분의 연속적인 비트 위치로부터 추출되는, 장치.

청구항 11

제10항에 있어서, 상기 연속적인 비트 위치는 상기 제1 소스 벡터 레지스터의 상기 K번째 부분의 최하위 연속적인 비트 위치의 세트인, 장치.

청구항 12

제1항 내지 제11항 중 어느 한 항에 있어서,

각각의 상기 명시된 레지스터의 각 부분은 N-비트 부분이고;

상기 제어 파라미터는 비트의 수를 명시하는 시프트 거리(shift distance) M을 나타내고;

상기 하나 이상의 추가 비트는 M 비트를 포함하고;

상기 제1 소스 벡터 레지스터의 상기 K번째 부분으로부터 상기 추출된 비트는 N에서 M을 뺀 비트를 포함하는, 장치.

청구항 13

제12항에 있어서,

각각의 N-비트 부분은 복수의 요소로 분할되고;

상기 시프트 거리는 정수 개의 요소에 해당하고;

상기 명시된 레지스터의 상기 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 상기 제1 부분의 요소의 최하위 서브세트를 포함하며 최하위 요소는 제외하는, 장치.

청구항 14

제12항에 있어서,

각각의 N-비트 부분은 복수의 요소로 분할되고;

상기 시프트 거리는 정수 개의 요소에 해당하고;

상기 명시된 레지스터의 상기 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 상기 제1 부분의 요소의 최상위 서브세트를 포함하는, 장치.

청구항 15

제1항 내지 제14항 중 어느 한 항에 있어서, 상기 목적지 벡터 레지스터는 상기 제2 소스 벡터 레지스터인, 장치.

청구항 16

제1항 내지 제15항 중 어느 한 항에 있어서, 상기 프로세싱 회로부는 상기 복수의 박동 중 적어도 2개를 병렬로 처리하도록 구성되는, 장치.

청구항 17

제1항 내지 제16항 중 어느 한 항에 있어서, 상기 프로세싱 회로부는 상기 주어진 벡터 명령어의 상기 복수의 박동 모두를 병렬로 수행하기에 불충분한 하드웨어를 포함하는, 장치.

청구항 18

제1항 내지 제16항 중 어느 한 항에 있어서, 상기 프로세싱 회로부는 상기 주어진 벡터 명령어의 상기 복수의 박동 모두를 병렬로 처리하도록 구성되는, 장치.

청구항 19

제1항 내지 제18항 중 어느 한 항에 있어서,

상기 디코더 회로부는 프로그램 카운터 순서로 상기 벡터 추출 및 병합 명령어에 인접한 메모리 데이터 전송 명령어에 응답하여 메모리 어드레스 및 상기 복수의 벡터 레지스터의 전송 레지스터를 명시하여 데이터 전송 제어 신호를 생성하고;

상기 장치는 복수의 박동의 메모리 데이터 전송 프로세싱을 수행하기 위해 상기 데이터 전송 제어 신호에 응답하는 데이터 제어 회로부를 더 포함하고, 각각의 박동은 상기 전송 레지스터의 해당 부분으로 데이터 전송을 수행하고, 상기 데이터 전송 명령어의 어느 박동이 완료되었는지를 나타내는 상기 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 메모리 데이터 전송 명령어의 완료된 박동을 억제하는 것을 포함하고;

상기 장치는, 상기 전송 레지스터가 상기 명시된 레지스터 중 하나일 때, 상기 벡터 추출 및 병합 명령어에 응답하여, 상기 전송 레지스터의 부분의 제2 서브세트에 해당하는 상기 복수의 박동 프로세싱의 제2 서브세트를 상기 프로세싱 회로부가 수행하는 것과 병렬로 상기 전송 레지스터의 부분의 제1 서브세트에 해당하는 상기 복수의 박동의 메모리 데이터 전송 프로세싱의 제1 서브세트를 수행하도록 구성되는, 장치.

청구항 20

제1항 내지 제19항 중 어느 한 항에 있어서, 상기 제어 파라미터는 상기 벡터 추출 및 병합 명령어에서 즉시 값으로서 명시되는, 장치.

청구항 21

제1항 내지 제20항 중 어느 한 항에 있어서, 상기 명시된 레지스터의 상기 제1 부분은 상기 명시된 레지스터의 최하위 부분이고, 상기 명시된 레지스터의 상기 마지막 부분은 상기 명시된 레지스터의 최상위 부분인, 장치.

청구항 22

복수의 벡터 레지스터, 디코더 회로부 및 프로세싱 회로부를 포함하는 장치를 작동시키는 방법으로서, 상기 방법은:

상기 디코더 회로부를 사용하고 벡터 추출 및 병합 명령어에 응답하여, 제어 신호, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터를 생성하는 단계; 및

상기 프로세싱 회로부를 사용하고 상기 제어 신호에 응답하여, 복수의 박동 프로세싱을 수행하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하는 단계, 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하는 단계, 및 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하는 단계를 포함하며,

상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 상기 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달되는, 방법.

청구항 23

장치의 제조를 위한 컴퓨터 판독가능 코드를 저장하기 위한 컴퓨터 판독가능 매체로서, 상기 컴퓨터 판독가능 저장 매체는:

복수의 벡터 레지스터;

제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 회로부; 및

상기 제어 신호에 응답하여 복수의 박동 프로세싱을 수행하는 프로세싱 회로부를 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 회로부는 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것,

상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 상기 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달되는, 컴퓨터 판독가능 매체.

청구항 24

명령어 실행 환경을 제공하도록 호스트 데이터 프로세싱 장치를 제어하기 위한 컴퓨터 프로그램으로서, 상기 컴퓨터 프로그램은:

복수의 벡터 레지스터를 포함하는 레지스터 로직;

제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 로직; 및

상기 제어 신호에 응답하여 복수의 박동 프로세싱을 수행하는 프로세싱 로직을 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 로직은 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 상기 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달되는, 컴퓨터 프로그램.

발명의 설명

기 술 분 야

[0001] 본 기술은 장치, 장치를 작동시키는 방법 및 장치의 제조를 위한 컴퓨터 판독가능 코드를 저장하기 위한 컴퓨터 판독가능 매체에 관한 것이다.

[0002] 일부 데이터 프로세싱 시스템은, 명령어의 소스 피연산자 또는 결과 값이 다수의 부분을 포함하는 벡터인 벡터 명령어의 프로세싱을 지원한다. 단일 명령어에 응답하여 벡터의 다수의 별개의 부분의 프로세싱을 지원함으로써, 코드 밀도가 개선되고 명령어의 페칭 및 디코딩의 오버헤드가 감소될 수 있다. 때때로, 벡터의 부분이 서로 의존하는 벡터 명령어가 수행되는 것이 바람직하다.

[0003] 일부 구성에 따르면 다음의 장치가 제공되며, 상기 장치는:

[0004] 복수의 벡터 레지스터;

[0005] 제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 회로부; 및

[0006] 상기 제어 신호에 응답하여 복수의 박동(beat) 프로세싱을 수행하는 프로세싱 회로부를 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 회로부는 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

[0007] 상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

[0008] 상기 제어 파라미터에 의해 명시된 바와 같이, 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 비트를 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

[0009] 상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 상기 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

[0010] 상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

[0011] 상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달된다.

[0012] 일부 구성에 따르면 복수의 벡터 레지스터, 디코더 회로부 및 프로세싱 회로부를 포함하는 장치를 작동시키는 방법이 제공되며, 상기 방법은:

[0013] 상기 디코더 회로부를 사용하고 벡터 추출 및 병합 명령어에 응답하여, 제어 신호, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터를 생성하는 단계; 및

[0014] 상기 프로세싱 회로부를 사용하고 상기 제어 신호에 응답하여, 복수의 박동 프로세싱을 수행하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하는 단계, 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하는 단계, 및 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하는 단계를 포함하며,

[0015] 상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

[0016] 상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

[0017] 상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

[0018] 상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

[0019] 상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달된다.

[0020] 일부 구성에 따르면 장치의 제조를 위한 컴퓨터 판독가능 코드를 저장하기 위한 컴퓨터 판독가능 매체가 제공되며, 상기 컴퓨터 판독가능 매체는:

[0021] 복수의 벡터 레지스터;

[0022] 제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 회로부; 및

[0023] 상기 제어 신호에 응답하여 복수의 박동 프로세싱을 수행하는 프로세싱 회로부를 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 회로부는 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

[0024] 상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

[0025] 상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

[0026] 상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

[0027] 상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

[0028] 상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달된다.

[0029] 일부 구성에서 컴퓨터 판독가능 매체는 비일시적 컴퓨터 판독가능 매체이다.

[0030] 일부 구성에 따르면 명령어 실행 환경을 제공하기 위해 호스트 데이터 프로세싱 장치를 제어하기 위한 컴퓨터 프로그램이 제공되며, 상기 컴퓨터 프로그램은:

[0031] 복수의 벡터 레지스터를 포함하는 레지스터 로직;

[0032] 제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어, 제어 파라미터를 명시하는 상기 벡터 추출 및 병합 명령어 및, 상기 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터에 응답하는 디코더 로직; 및

[0033] 상기 제어 신호에 응답하여 복수의 박동 프로세싱을 수행하는 프로세싱 로직을 포함하되, 각각의 박동은 적어도 상기 제1 소스 벡터 레지스터 및 상기 목적지 벡터 레지스터의 부분에 해당하는 조합 프로세싱을 포함하며, 상기 프로세싱 로직은 상기 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 상기 박동 상태 정보에 의해 완료된 것으로 표시된 상기 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성되고,

[0034] 상기 명시된 레지스터 각각의 K번째 부분에 해당하는 K번째 박동에 대한 상기 조합 프로세싱은:

[0035] 상기 제어 파라미터에 의해 명시된 비트를 상기 제1 소스 벡터 레지스터의 상기 K번째 부분에서 추출하는 것, 상기 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 상기 연결의 결과를 상기 목적지 레지스터의 상기 K번째 부분에 저장하는 것;

[0036] 상기 K번째 부분이 상기 명시된 레지스터의 마지막 부분이 아닐 때, 상기 목적지 레지스터에 저장되지 않은 상기 제1 소스 벡터 레지스터의 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함하며;

[0037] 상기 명시된 레지스터의 제1 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고;

[0038] 상기 명시된 레지스터의 상기 제1 부분 이외의 각각의 부분에 대해, 상기 하나 이상의 추가 비트는 상기 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달된다.

[0039] 일부 구성에서 상기 컴퓨터 프로그램은 비일시적 컴퓨터 판독가능 매체에 기록된다.

[0040] 본 기술은 첨부 도면에 도시된 바와 같이 그의 구성을 참조하여 예로서만 더 설명될 것이다.

[0041] 도 1은 본 기술의 다양한 구성에 따른 벡터 명령어의 프로세싱을 지원하는 데이터 프로세싱 장치를 개략적으로 도시한다;

[0042] 도 2는 본 기술의 다양한 구성에 따른 벡터 명령어의 중첩 실행의 예를 개략적으로 도시한다;

[0043] 도 3은 본 기술의 다양한 구성에 따른 상이한 프로세서 구현 사이에서 또는 명령어 실행의 상이한 인스턴스 사이에 런타임에서 연속적인 벡터 명령어 간의 중첩의 양을 스케일링하는 세 가지 예를 개략적으로 도시한다;

[0044] 도 4는 본 기술의 다양한 구성에 따른 다수의 벡터 명령어의 블록의 어느 박동이 완료되었는지를 나타내기 위한 박동 상태 정보에 대한 예시적인 인코딩을 개략적으로 도시한다;

[0045] 도 5는 본 기술의 다양한 구성에 따른 디버그 이벤트 또는 예외의 발생 시 박동 상태 정보를 기록하는 2개의 예를 개략적으로 도시한다;

[0046] 도 6은 본 기술의 다양한 구성에 따른 디버그 이벤트 또는 예외로부터 복귀한 후 프로세싱을 재개하기 위해 박동 상태 정보를 사용하는 예를 개략적으로 도시한다;

[0047] 도 7은 본 기술의 다양한 구성에 따른 데이터 프로세싱 장치를 개략적으로 도시한다;

[0048] 도 8은 본 기술의 다양한 구성에 따른 데이터 프로세싱 장치를 개략적으로 도시한다;

[0049] 도 9는 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0050] 도 10은 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0051] 도 11은 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0052] 도 12는 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0053] 도 13은 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0054] 도 14는 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0055] 도 15는 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0056] 도 16은 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0057] 도 17은 본 기술의 다양한 구성에 따른 벡터 추출 및 병합 명령어의 예를 개략적으로 도시한다;

[0058] 도 18은 본 기술의 다양한 구성에 따른 장치에 의해 수행되는 단계의 시퀀스를 개략적으로 도시한다;

[0059] 도 19는 본 기술의 다양한 구성에 따른 장치를 개략적으로 도시한다;

[0060] 도 20은 일부 예시적인 구성에 따라 사용될 수 있는 시뮬레이터를 개략적으로 도시한다.

[0061] 주어진 명령어 세트 아키텍처에 따라 기입된 소프트웨어는 상이한 하드웨어 구현을 갖는 상이한 데이터 프로세싱 장치의 범위에서 실행될 수 있다. 주어진 명령어 세트가 실행될 때 아키텍처에 의해 기대하는 결과를 제공하는 한, 특정 구현은 이러한 아키텍처 준수를 달성하는 어떤 방식으로든 마이크로 아키텍처 설계를 자유롭게 변경할 수 있다. 예를 들어, 일부 애플리케이션의 경우, 에너지 효율은 성능보다 더 중요할 수 있고, 따라서 명령어 세트 아키텍처로부터 명령어를 실행하기 위해 제공되는 프로세싱 회로부의 마이크로-아키텍처 설계는 이것이 성능을 희생하는 경우라도 가능한 적은 에너지를 소비하도록 설계될 수 있다. 다른 애플리케이션은 성능을 에너지 효율보다 더 중요한 기준으로서 볼 수 있고, 따라서 명령어의 더 큰 처리량을 가능하게 하지만 더 많은 전력을 소비할 수 있는 더 복잡한 하드웨어 구조를 포함할 수 있다. 따라서, 상이한 에너지 또는 성능 포인트의 범위에 걸쳐 스케일링을 지원하도록 명령어 세트 아키텍처를 설계하는 것이 바람직할 수 있다.

[0062] 일부 구성에서, 제어 신호를 생성하기 위한 벡터 추출 및 병합 명령어에 응답하는 복수의 벡터 레지스터 및 디코더 회로부를 포함하는 장치가 제공된다. 벡터 추출 및 병합 명령어는 제어 파라미터를 명시하고, 복수의 벡터 레지스터의 명시된 레지스터로서, 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 및 목적지 벡터 레지스터를 명시한다. 장치는 또한 복수의 박동 프로세싱을 수행하기 위해 제어 신호에 응답하는 프로세싱 회로부를 포함한다. 각각의 박동은 적어도 제1 소스 벡터 레지스터 및 목적지 벡터 레지스터의 부분에 해당하는 조합 프

로세싱을 포함한다. 프로세싱 회로부는 벡터 추출 및 병합 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 박동 상태 정보에 의해 완료된 것으로 표시된 벡터 추출 및 병합 명령어의 완료된 박동을 억제하도록 구성된다. 명시된 레지스터의 각각의 K번째 부분에 해당하는 K번째 박동에 대한 조합 프로세싱은, 제어 파라미터에 의해 명시된 바와 같이, 제1 소스 벡터 레지스터의 K번째 부분에서 비트를 추출하는 것, 추출된 비트를 하나 이상의 추가 비트와 연결하는 것, 및 연결의 결과를 목적지 레지스터의 K번째 부분에 저장하는 것을 포함한다. K번째 부분에 대한 조합 프로세싱은, K번째 부분이 명시된 레지스터의 마지막 부분이 아닐 때, 목적지 레지스터에 저장되지 않은 제1 소스 벡터 레지스터의 K번째 부분 중 적어도 하나의 비트가 상기 복수의 박동 중 (K+1)번째 박동에서 처리되도록 전달하는 것을 포함한다. 명시된 레지스터의 제1 부분에 대해, 하나 이상의 추가 비트는 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되고, 명시된 레지스터의 제1 부분 이외의 각각의 부분에 대해, 하나 이상의 추가 비트는 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터 전달된다.

[0063] 이러한 배열은 벡터 명령어를 지원하는 마이크로 아키텍처가 더 효율적으로 상이한 성능 및 에너지 포인트로 스케일링될 수 있게 한다. 2개 이상의 벡터 명령어의 완료된 박동을 추적하는 박동 상태 정보를 제공함으로써, 이는 상이한 벡터 명령어의 실행이 중첩되는 양을 변경시키는 특정 마이크로-아키텍처 구현에 대한 자유를 제공하여, 각각 부분적으로 실행된 명령어의 진행을 여전히 추적하는 동시에 상이한 벡터 명령어의 각자의 박동을 서로 병렬로 수행하는 것이 가능하다. 일부 마이크로-아키텍처 구현은, 하나의 벡터 명령어의 모든 박동이 다음 명령어가 시작되기 전에 완료되도록, 각자의 벡터 명령어의 실행을 전혀 중첩되지 않도록 선택할 수 있다. 다른 마이크로-아키텍처는 제2 벡터 명령어의 제1 박동의 서브세트가 제1 벡터 명령어로부터 제2 박동의 서브세트와 병렬로 수행되도록 연속적인 벡터 명령어의 실행을 스태거링할 수 있다.

[0064] 벡터 추출 및 병합 명령어는 디코더 회로부에 의해 해석되는 명령어 세트 아키텍처의 명령어이다. 명령어 세트 아키텍처는 연산을 수행하도록 프로세싱 회로부에 지시하기 위해 프로그래머 또는 컴파일러에 의해 사용될 수 있는 완전한 명령어 세트를 형성한다. 논의된 바와 같이, 프로세싱 회로부가 명령어 세트 아키텍처를 준수하는 한, 마이크로 아키텍처의 실제 구현, 즉 프로세싱 회로부를 구성하는 회로 및 로직 블록의 물리적 배열은 구현마다 다를 수 있다. 일부 마이크로-아키텍처 구현은 벡터의 모든 부분을 병렬로 프로세싱할 수 있는 반면, 다른 구현은 한 번에 벡터의 하나 이상의 부분을 프로세싱할 수 있다. 일부 벡터 명령어는 이러한 유연성에 잘 맞을 수 있다. 예를 들어, 2개의 소스 벡터의 복수의 요소에 대한 요소별 덧셈을 지원하는 벡터 명령어는 각각 벡터의 요소에 해당하는 복수의 스칼라 덧셈으로 분할될 수 있다. 그러나, 데이터가 상이한 요소 사이에서 또는 상이한 부분(벡터의 복수의 요소를 포함할 수 있음) 사이에서 전파되는 명령어, 즉 상이한 부분이 서로 의존하는 명령어는 마이크로-아키텍처 구현의 그러한 유연성에 그렇게 쉽게 적응되지 않을 수 있다.

[0065] 벡터 추출 및 병합 명령어는 그러한 명령어 중 하나이다. 벡터 추출 및 병합 명령어에서, 제1 소스 벡터 레지스터로부터의 하나 이상의 비트는 제2 소스 벡터 레지스터로부터의 하나 이상의 비트와 연결된다. 본 발명자는 이러한 마이크로-아키텍처 유연성을 제공하는 벡터 추출 및 병합 명령어가 하나 이상의 박동(각각의 명시된 벡터 레지스터의 하나 이상의 부분에 해당함)을 병렬로 또는 스태거링된 방식으로 처리하고, 박동 프로세싱 사이에(즉, 하나의 부분에서 다른 부분으로) 적어도 하나의 비트를 전달하도록 배열된 프로세싱 회로부를 제공함으로써 구현될 수 있다는 것을 깨달았다. 결과적으로, 프로세싱 회로부는 각각의 박동을 서로 다른 박동과 진정으로 독립적인 것으로 간주하지 않는다. 대신, 특정 정보는 하나의 처리된 박동으로부터 다른 처리된 박동으로 전파될 수 있다. 특히, 벡터 추출 및 병합 명령어는 제어 파라미터 및 복수의 벡터 레지스터를 입력으로서 명시한다. 복수의 벡터 레지스터는 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터 및 목적지 벡터 레지스터를 포함한다. 제어 파라미터는 각각의 박동 프로세싱 동안 제1 소스 벡터 레지스터로부터 추출될 비트의 수를 나타내며, 디코더 회로부에 전달되는 파라미터로서 명령어에서 명시적으로 명시될 수 있거나 고정된 값을 갖는 것으로서 명령어에서 암시적으로 명시될 수 있다. 예를 들어, 명령어 세트 아키텍처는 하나 이상의 벡터 추출 및 병합 명령어를 정의할 수 있고, 이들 각각은 고정된 제어 파라미터를 암시적으로 정의한다. 제어 파라미터는 표시 값일 수 있고, 따라서 추출하려는 비트의 개수를 간접적으로 명시할 수 있다.

[0066] 이러한 방식으로 정의된 조합 프로세싱은 제2 소스 벡터 레지스터의 하나 이상의 추가 비트가 제1 소스 벡터 레지스터로부터 추출된 하나 이상의 비트(제어 파라미터에 의해 명시된 바와 같은)와 연결되는 제1 박동(제1 부분)으로부터의 비트의 전파를 야기한다. 그 후, 제1 박동(K=1)의 제1 소스 벡터 레지스터로부터의 비트는 제2 박동(K=2)으로 전달(전파)되고, 후속 박동의 프로세싱 시점에 후속 박동에서 제1 소스 벡터 레지스터의 하나 이상의 비트와 연결된다. 그 후, K번째 박동의 하나 이상의 비트가 (K+1)번째 박동으로 전달되면서 프로세스가 반복된다. 캐리는 K번째 부분이 명시된 레지스터의 마지막 부분이 아닐 때 생성된다. 일부 구성에서, 명시된

벡터 레지스터의 마지막 부분에 대해 캐리가 생성되지 않는다. 일부 대안적인 구성에서, 제1 소스 벡터 레지스터의 마지막 부분의 적어도 하나의 비트의 캐리가 생성된다. 박동의 순서는 벡터 레지스터 내의 비트의 순서와 독립적일 수 있다는 것이 인식될 것이다. 한 구성에서 제1 박동(K=1)은 벡터 레지스터의 최하위 비트 세트에 해당할 수 있고, 마지막 박동은 벡터 레지스터의 최상위 비트 세트에 해당할 수 있다. 그러나 일부 대안적인 구성에서 제1 박동(K=1)은 벡터 레지스터의 최상위 비트 세트에 해당할 수 있고, 마지막 박동은 벡터 레지스터의 최하위 비트 세트에 해당할 수 있다.

[0067] 이러한 방식으로, 장치는 마이크로-아키텍처 구현이 여전히 명령어 세트 아키텍처에 준수를 허용하면서 변경될 수 있는 벡터 추출 및 병합 명령어의 구현을 가능하게 하는 프로세싱 회로부를 제공함으로써, 전력 제약 및 회로 크기 요건에 기초하여 적응될 수 있는 유연한 구현을 가능하게 한다.

[0068] 일부 구성에서, 디코더 회로부는 스칼라 레지스터를 명시하는 벡터 추출 및 병합 명령어에 응답하며; 복수의 박동은 현재 실행 중인 하나 이상의 박동의 서브세트를 포함하며, 여기서 현재 실행 중인 박동의 서브세트는 완료된 박동을 제외하고; 프로세싱 회로부는 제어 신호에 응답하여, 스칼라 레지스터에 캐리 데이터의 적어도 하나의 아이템을 저장하고, 캐리 데이터의 적어도 하나의 아이템은 현재 실행 중인 하나 이상의 박동의 서브세트와 복수의 박동 중 하나 이상의 박동의 추가 서브세트 사이에서 전달될 하나 이상의 비트를 포함한다. 현재 실행 중인 박동의 서브세트는 복수의 박동 중 하나 이상의 박동을 포함하고, 복수의 박동 중 적어도 하나의 박동의 추가 서브세트를 제외한다. 이러한 구성에서, 스칼라 레지스터는 현재 실행 중인 박동의 서브세트와 하나 이상의 박동의 하나 이상의 추가 서브세트 사이에서 캐리 데이터의 적어도 하나의 아이템을 전달하는 데 사용된다. 스칼라 레지스터는 복수의 스칼라 레지스터 중 하나로서, 예를 들어 벡터 추출 및 병합 명령어에서의 파라미터로서 명시적으로 명시될 수 있다. 대안적으로, 프로세싱 회로부는 벡터 추출 및 병합 명령어에서 암시적으로 정의되는 특정 캐리 레지스터를 포함할 수 있다.

[0069] 캐리 레지스터는 캐리 데이터를 현재 실행 중인 박동의 서브세트 내로 또는 현재 실행 중인 박동의 서브세트 밖으로 전파하는 데 사용될 수 있다. 일부 구성에서, 프로세싱 회로부는 현재 실행 중인 하나 이상의 박동의 세트의 제1 박동에 대해 제어 신호에 응답하고, 벡터 추출 및 병합 명령어의 실행 전의 박동 상태 정보가 적어도 하나의 박동이 억제되어야 한다는 것을 표시할 때, 스칼라 레지스터로부터 하나 이상의 추가 비트를 검색한다. 하나 이상의 박동 프로세싱의 서브세트는 제1 박동의 서브세트로부터 다음 박동의 서브세트로 하나 이상의 정보 비트가 전파되는 순서대로 실행된다. 실행 동안, 프로세싱 회로부는 제어 정보를 판독하여 어느 박동이 현재 실행 중인 하나 이상의 박동의 서브세트의 제1 박동을 포함하는지를 결정한다. 이전에 하나 이상의 박동 프로세싱이 실행된 경우, 제어 정보는 이들 하나 이상의 박동이 억제되어야 함을 나타낸다. 따라서, 프로세싱 회로부는 캐리 데이터가 스칼라 레지스터에서 이용가능함을 추론하고 스칼라 레지스터에서의 캐리 데이터로부터 하나 이상의 추가 비트를 추출할 수 있다.

[0070] 캐리 데이터에 포함되는 데이터는 다양한 형태를 가질 수 있다. 일부 구성에서 전달될 하나 이상의 비트는 제1 소스 벡터 레지스터의 부분의 모든 비트를 포함하고; 스칼라 레지스터로부터 하나 이상의 추가 비트를 검색하는 것은 스칼라 레지스터로부터 마지막 비트의 서브세트를 검색하는 것을 포함한다. 그 결과, 하나 이상의 추가 비트의 추출은 추출이 스칼라 레지스터로부터 또는 제2 소스 벡터 레지스터로부터 이루어지는지와 독립적인 동일한 패턴을 따르므로, 결과적으로 구현이 더 간단해지고, 이에 의해 구현이 간단해진다. 일부 구성에서 전달될 하나 이상의 비트는 스칼라 레지스터의 임시 비트 위치의 세트에 저장된 제1 소스 벡터 레지스터의 부분으로부터의 마지막 M 비트의 세트를 포함하고; 스칼라 레지스터로부터 하나 이상의 추가 비트를 검색하는 것은 스칼라 레지스터의 임시 비트 위치의 세트로부터 비트를 검색하는 것을 포함한다. 결과적으로, 스칼라 레지스터에서 전달되기 위해 더 적은 비트가 요구된다. 일부 구성에서, 마지막 비트의 서브세트는 최상위 비트의 서브세트로, 벡터 레지스터의 (K-1)번째 부분의 최상위 비트의 서브세트로부터 K번째 부분으로 데이터가 전파된다. 대안적인 구현에서 데이터는 반대 방향으로 전파될 수 있고, 이러한 구성에서, 마지막 비트의 서브세트는 최하위 비트의 서브세트이다.

[0071] 일부 구성에서 추출된 비트를 연결하는 것은 목적지 레지스터의 K번째 부분의 제1 연속적인 비트 위치의 세트에 추출된 비트를 저장하고 목적지 레지스터의 K번째 부분의 제2 연속적인 비트 위치의 세트에 하나 이상의 추가 비트를 저장하는 것을 포함한다. 일부 구성에서 제1 비트 위치의 서브세트 및 제2 비트 위치의 서브세트의 조합은 목적지 레지스터의 K번째 부분의 모든 비트 위치를 포함한다. 일부 구성에서 제1 연속적인 비트 위치의 세트 및 제2 연속적인 비트 위치의 세트는 비-중첩 비트 위치이다. 결과적으로, 목적지 레지스터의 K번째 부분의 모든 비트 위치는 하나 이상의 추가 비트 중 하나 또는 추출된 비트 중 하나로서 정의된다.

[0072] 제1 연속적인 비트 위치의 세트 및 제2 비트 위치의 세트의 순서는 구현에 따라 달라질 수 있다. 일부 구성에서 제1 연속적인 비트 위치의 세트는 목적지 레지스터의 K번째 부분의 최상위 비트 위치의 세트이고 제2 연속적인 비트 위치의 세트는 목적지 레지스터의 K번째 부분의 최하위 비트 위치의 세트이다. 대안적으로, 명시된 벡터의 프로세싱 순서는 반대로 될 수 있다. 따라서, 일부 구성에서 제1 연속적인 비트 위치의 세트는 목적지 레지스터의 K번째 부분의 최하위 비트 위치의 세트이고 제2 연속적인 비트 위치의 세트는 목적지 레지스터의 K번째 부분의 최상위 비트 위치의 세트이다.

[0073] 일부 구성에서 추출된 비트는 제1 소스 벡터 레지스터의 K번째 부분의 연속적인 비트 위치로부터 추출된다. 연속적인 비트 위치는 제어 파라미터에 의해 명시되고, 예를 들어, 제1 비트 위치 및 제2 비트 위치에 기초하여, 또는 제1 비트 위치 및 추출할 비트의 수에 기초하여 정의될 수 있다.

[0074] 일부 구성에서 연속적인 비트 위치는 제1 소스 벡터 레지스터의 K번째 부분의 최하위 연속적인 비트 위치의 세트이다. 이러한 구성에서, 제어 파라미터는 추출될 연속적인 비트 위치의 수를 명시하기 위해서만 요구된다. 추출될 연속적인 비트 위치의 수는 즉시 값으로 명시되거나 벡터 추출 및 병합 명령어에서 명시된 레지스터 내에 포함될 수 있다. 대안적인 구성에서 연속적인 비트 위치는 제1 소스 벡터 레지스터의 K번째 부분의 최상위 연속적인 비트 위치의 세트이다. 일부 구성에서 추출될 연속적인 비트 위치의 가능한 수의 서브세트만이 지원될 수 있다. 예를 들어, 일부 구성은 길이가 8, 16, 또는 24비트인 연속적인 비트 위치만을 지원할 수 있다. 따라서 이러한 구성에서 제어 파라미터는 지원되는 길이 중 하나를 선택함으로써 추출될 연속적인 비트 위치의 수를 간접적으로 명시할 수 있다. 이러한 구성은 제어 파라미터를 표현하는 데 필요한 비트의 수를 감소시킨다.

[0075] 일부 구성에서 각각의 명시된 레지스터의 각 부분은 N-비트 부분이고; 제어 파라미터는 비트의 수를 명시하는 시프트 거리(shift distance) M을 나타내고; 하나 이상의 추가 비트는 M 비트를 포함하고; 제1 소스 벡터 레지스터의 K번째 부분으로부터 추출된 비트는 N에서 M을 뺀 비트를 포함한다. 결과적으로, 벡터 추출 및 병합 명령어는 제2 소스 벡터 레지스터의 제1 부분으로부터의 M 비트를 제1 소스 벡터 레지스터의 제1 부분의 N에서 M을 뺀 비트와 결합하여 목적지 레지스터의 제1 부분을 형성한다. 또한, 벡터 추출 및 병합 명령어는 제1 소스 벡터 레지스터의 (K-1)번째 부분으로부터의 M 비트를 제1 소스 벡터 레지스터의 K번째 부분의 N에서 M을 뺀 비트와 결합한다. 다시 말해서, 제1 소스 벡터 레지스터의 각 부분의 M 비트는 목적지 벡터 레지스터의 다음 부분에 저장되도록 시프트된다.

[0076] 명시된 레지스터의 제1 부분에 대해, 하나 이상의 추가 요소는 다양한 방식으로 선택될 수 있다. 일부 구성에서, 각각의 N-비트 부분은 복수의 요소로 분할되고; 시프트 거리는 정수 개의 요소에 해당하고; 명시된 레지스터의 제1 부분에 대해, 하나 이상의 추가 비트는 제2 소스 벡터 레지스터의 제1 부분의 요소의 최상위 서브세트를 포함한다. 결과적으로, 시프트 및 병합 명령어는 결과 벡터 레지스터를 생성하기 위해 제1 소스 벡터 레지스터의 비트와 연결되는 제2 소스 벡터 레지스터의 최상위 서브세트를 취한다.

[0077] 대안적으로, 일부 구성에서 각각의 N-비트 부분은 복수의 요소로 분할되고; 시프트 거리는 정수 개의 요소에 해당하고; 명시된 레지스터의 제1 부분에 대해, 하나 이상의 추가 비트는 제2 소스 벡터 레지스터의 제1 부분의 요소의 최하위 서브세트를 포함하며 최하위 요소는 제외한다. 비트 수(또는 요소 수)만큼 시프트되는 시프트된 벡터를 순차적으로 생성하기 위해 벡터 추출 및 병합 명령어를 반복적으로 적용하는 것이 유익할 수 있는 일부 사용 사례 시나리오가 있다. 예를 들어, 유한 임펄스 응답 필터를 구현할 때, 시퀀스의 이전 벡터로부터 단일 요소만큼 시프트되는 벡터를 순차적으로 생성하는 것이 필요할 수 있다. 벡터 추출 및 병합 명령어는 초기 벡터, 예를 들어, 제2 소스 벡터 레지스터를 취하고, 하나의 요소만큼 시프트된 벡터의 시퀀스를 생성함으로써 시프트 벡터의 시퀀스가 생성될 수 있게 한다. 이러한 경우에, 제1 및 제2 소스 벡터 레지스터를 유지하는 것보다는, 이전의 목적지 레지스터가 제2 소스 벡터 레지스터로서 사용될 수 있다. 이러한 상황에서, 하나 이상의 추가 비트에 포함될 필요한 비트의 위치는 이미 최상위 요소로부터 하나 이상의 비트 위치만큼 시프트되었다. 따라서, 명시된 레지스터의 제1 부분의 경우, 하나 이상의 추가 비트로서 최하위 요소를 제외한 최하위 요소의 서브세트를 선택함으로써, 벡터 추출 및 병합 명령어는 제2 소스 벡터 레지스터가 선행 벡터 추출 및 병합 명령어의 결과를 포함하는 경우에 대해 적응될 수 있다. 일부 구성에서, 요소 폭(width)은 벡터 추출 및 병합 명령어의 폭 파라미터에 의해 제어될 수 있다. 일부 구성에서 제어 파라미터는 추출할 비트와 요소 폭 둘 다를 나타낼 수 있다. 이러한 구성에서, 파라미터를 인코딩하는 데 필요한 비트의 수는, 요소 폭과 추출할 비트의 수 및 비트의 위치의 제한된 수의 조합만이 지원되는 상황에서 감소된다.

[0078] 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터 및 목적지 벡터 레지스터 각각에 대해 별개의 벡터 레지스터

를 명시하기 보다는, 일부 구성에서는 목적지 벡터 레지스터가 제2 소스 벡터 레지스터이다. 제2 소스 벡터 레지스터를 목적지 레지스터로서 용도 변경하는 것은 레지스터 요건과 벡터 추출 및 병합 명령어에 필요한 인코딩 공간을 감소시킨다.

[0079] 논의된 바와 같이, 벡터 추출 및 병합 명령어는 주어진 사이클에서 복수의 박동 프로세싱 중 하나 이상을 수행할 수 있는 하드웨어를 사용하여 유연하게 구현될 수 있다. 일부 구성에서 프로세싱 회로부는 복수의 박동 중 적어도 2개를 병렬로 처리하도록 구성된다. 이러한 프로세싱 회로부의 하드웨어 제공은 단지 적어도 2개의 박동을 처리하는 데만 충분할 수 있고, 프로세싱 회로부는 복수의 박동 중 적어도 2개를 프로세싱하는 것에 병렬로 인접 명령어의 박동을 처리하도록 구성될 수 있다. 대안적으로, 프로세싱 회로부는 복수의 박동의 모든 박동을 병렬로 처리하기에 충분할 수 있다.

[0080] 일부 구성에서 프로세싱 회로부는 주어진 벡터 명령어의 복수의 박동 모두를 병렬로 수행하기에 불충분한 하드웨어를 포함한다. 따라서, 프로세싱 회로부는 제1 서브세트를 완료한 후에 주어진 벡터 명령어의 제2 박동의 서브세트를 수행할 수 있다. 제1 및 제2 서브세트는 프로세서 구현에 따라 단일 박동을 포함할 수 있거나 다중 박동을 포함할 수도 있다.

[0081] 일부 구성에서 프로세싱 회로부는 주어진 벡터 명령어의 복수의 박동 모두를 병렬로 처리하도록 구성된다. 이러한 하드웨어를 갖는 프로세싱 회로부는 여전히 위에서 명시된 바와 같은 박동 상태 정보를 생성하고 사용할 수 있지만, 박동 상태 정보는 일반적으로 완료된 박동이 없었음을 나타낼 것이다. 따라서, 박동 상태 정보를 정의함으로써, 아키텍처는 상이한 구현의 범위를 지원할 수 있다.

[0082] 일부 구성에서 디코더 회로부는 프로그램 카운터 순서로 벡터 추출 및 병합 명령어에 인접한 메모리 데이터 전송 명령어에 응답하여 메모리 어드레스 및 복수의 벡터 레지스터의 전송 레지스터를 명시하여 데이터 전송 제어 신호를 생성한다. 장치는 복수의 박동의 메모리 데이터 전송 프로세싱을 수행하기 위해 데이터 전송 제어 신호에 응답하는 데이터 제어 회로부를 더 포함하고, 각각의 박동은 전송 레지스터의 해당 부분으로 데이터 전송을 수행하고, 데이터 전송 명령어의 어느 박동이 완료되었는지를 나타내는 박동 상태 정보를 설정하고, 박동 상태 정보에 의해 완료된 것으로 표시된 메모리 데이터 전송 명령어의 완료된 박동을 억제하는 것을 포함하고; 장치는 전송 레지스터가 명시된 레지스터 중 하나일 때, 벡터 추출 및 병합 명령어에 응답하여, 전송 레지스터의 부분의 제2 서브세트에 해당하는 복수의 박동 프로세싱의 제2 서브세트를 프로세싱 회로부가 수행하는 것과 병렬로 전송 레지스터의 부분의 제1 서브세트에 해당하는 복수의 박동의 메모리 데이터 전송 프로세싱의 제1 서브세트를 수행하도록 구성된다. 제1 박동의 서브세트 및 제2 박동의 서브세트는 각각 동일한 수의 박동 또는 상이한 수의 박동을 포함할 수 있다. 예를 들어, 일부 구성에서, 장치는 복수의 부분(데이터 전송 프로세싱의 복수의 박동에 해당)에 대한 메모리 데이터 전송 동작을 수행하기에 충분한 하드웨어를 제공받을 수 있지만, 벡터 추출 및 병합 명령어에 대한 단일 박동 프로세싱을 수행하기에 충분한 하드웨어만을 제공받을 수 있다. 대안적으로, 장치에는 부분의 절반에 대한 메모리 데이터 전송 동작을 수행하기에 충분한 하드웨어 및 벡터 길이의 부분의 절반에 대한 벡터 추출 및 병합 명령어에 대한 박동 프로세싱을 수행하기에 충분한 하드웨어가 제공될 수 있다. 이러한 상황 각각에서, 복수의 박동 프로세싱의 제1 서브세트와 복수의 박동 프로세싱의 제2 서브세트에 대해 사용되고 있는 데이터 및 하드웨어 사이에 중첩이 없다. 따라서, 제1 박동의 서브세트 및 제2 박동의 서브세트를 병렬화할 수 있는 프로세싱 장치를 제공함으로써, 더 큰 명령어 처리량이 달성될 수 있다.

[0083] 일부 구성에서, 제어 파라미터는 벡터 추출 및 병합 명령어에서 즉시 값으로서 명시된다. 일부 대안적인 구성에서, 제어 파라미터는 제어 파라미터가 정의되는 레지스터로서 명시될 수 있다.

[0084] 일부 구성에서, 명시된 레지스터의 제1 부분은 명시된 레지스터의 최하위 부분이고, 명시된 레지스터의 마지막 부분은 명시된 레지스터의 최상위 부분이다. 대안적인 구성에서, 명시된 레지스터의 제1 부분은 명시된 레지스터의 최상위 부분이고, 명시된 레지스터의 마지막 부분은 명시된 레지스터의 최하위 부분이다. 이러한 방식으로, 프로세싱 장치에는 제2 소스 벡터 레지스터로부터 추출된 하나 이상의 추가 비트를 특정 구현 선택에 의존하는 최하위 단부 또는 최상위 단부로부터 목적지 레지스터로 시프트함으로써 벡터 추출 및 병합 명령어를 수행하는 회로가 제공될 수 있다.

[0085] 본원에 설명된 개념은 설명된 개념을 구현하는 장치의 제조를 위한 컴퓨터-판독가능 코드로 구현될 수 있다. 예를 들어, 컴퓨터-판독가능 코드는 개념을 구현하는 장치를 포함하는 집적 회로를 제조하기 위해 전자 설계 자동화(electronic design automation; EDA) 스테이지를 포함하는 반도체 설계 및 제조 프로세스의 하나 이상의 스테이지에서 사용될 수 있다. 위의 컴퓨터-판독가능 코드는 추가적으로 또는 대안적으로 본원에 설명된 개념을 구현하는 장치의 정의, 모델링, 시뮬레이션, 검증 및/또는 테스트를 가능하게 할 수 있다.

[0086] 예를 들어, 본원에 설명된 개념을 구현하는 장치의 제조를 위한 컴퓨터-판독가능 코드는 개념의 하드웨어 기술 언어(hardware description language; HDL) 표현을 정의하는 코드로 구현될 수 있다. 예를 들어, 코드는 개념을 구현하는 장치를 정의하기 위한 하나 이상의 로직 회로의 레지스터-전송-레벨(register-transfer-level; RTL) 개요(abstraction)를 정의할 수 있다. 코드는 FIRRTL과 같은 중간 표현뿐만 아니라 Verilog, SystemVerilog, Chisel, 또는 VHDL(Very High-Speed Integrated Circuit Hardware Description Language)로 장치를 구현하는 하나 이상의 로직 회로의 HDL 표현을 정의할 수 있다. 컴퓨터-판독가능 코드는 SystemC 및 SystemVerilog와 같은 시스템 레벨 모델링 언어를 사용하여 개념을 구현하는 정의 또는 개념의 시뮬레이션, 기능적 및/또는 형식적 검증, 및 테스트를 가능하게 하기 위해 컴퓨터에 의해 해석될 수 있는 개념의 다른 거동 표현을 제공할 수 있다.

[0087] 추가적으로 또는 대안적으로, 컴퓨터 판독가능 코드는 GDSII와 같은 표현을 포함하는 하나 이상의 넷리스트 또는 집적 회로 레이아웃 정의와 같이 본원에 설명된 개념을 구현하는 집적 회로 컴포넌트의 하위 레벨 설명을 정의할 수 있다. 집적 회로 컴포넌트의 하나 이상의 넷리스트 또는 다른 컴퓨터 판독가능 표현은 본 발명을 구현하는 장치의 제조에 사용하기 위한 정의를 생성하기 위해 하나 이상의 로직 합성 프로세스를 RTL 표현에 적용함으로써 생성될 수 있다. 대안적으로 또는 추가적으로, 하나 이상의 로직 합성 프로세스는 설명된 개념을 구현하도록 FPGA를 구성하기 위해 필드 프로그래밍 가능 게이트 어레이(FPGA) 내에 로딩될 비트스트림을 컴퓨터-판독가능 코드로부터 생성할 수 있다. FPGA는 집적 회로에서 제조하기 전에 개념의 검증 및 테스트 목적으로 배치되거나 FPGA가 제품에 직접적으로 배치될 수 있다.

[0088] 컴퓨터-판독가능 코드는, 예를 들어, 본 발명을 구현하는 장치를 제조하기 위해 반도체 설계 및 제조 프로세스에서 사용될 RTL 표현, 넷리스트 표현, 또는 또 다른 컴퓨터-판독가능 정의 중 하나 이상의 혼합을 포함하는, 장치의 제조를 위한 코드 표현의 혼합을 포함할 수 있다. 대안적으로 또는 추가적으로, 개념은 반도체 설계 및 장치를 제조하기 위한 제조 프로세스에서 사용될 컴퓨터-판독가능 정의와, 일단 제조되면 정의된 장치에 의해 실행될 명령어를 정의하는 컴퓨터-판독가능 코드의 조합으로 정의될 수 있다.

[0089] 그러한 컴퓨터-판독가능 코드는 임의의 알려진 일시적 컴퓨터-판독가능 매체(예컨대, 네트워크를 거치는 코드의 유선 또는 무선 송신) 또는 반도체, 자기 디스크 또는 광학 디스크와 같은 비일시적 컴퓨터-판독가능 매체에 배치될 수 있다. 컴퓨터-판독가능 코드를 사용하여 제조된 집적 회로는 중앙 프로세싱 유닛, 그래픽 프로세싱 유닛, 신경 프로세싱 유닛, 디지털 신호 프로세서, 또는 개별적으로 또는 집합적으로 개념을 구현하는 다른 컴포넌트 중 하나 이상과 같은 컴포넌트를 포함할 수 있다.

[0090] 본 발명의 특정 구성은 이제 첨부 도면을 참조하여 설명될 것이다.

[0091] 도 1은 벡터 명령어의 프로세싱을 지원하는 데이터 프로세싱 장치(2)의 예를 개략적으로 도시한다. 이것은 설명의 용이성을 위해 단순화된 다이어그램이며, 실제로 장치는 간결성을 위해 도 1에 도시되지 않은 많은 요소를 가질 수 있다는 것을 이해할 것이다. 장치(2)는 명령어 디코더(6)에 의해 디코딩된 명령어에 응답하여 데이터 프로세스를 수행하기 위한 프로세싱 회로부(4)를 포함한다. 프로그램 명령어는 메모리 시스템(8)으로부터 인출되고, 명령어 디코더에 의해 디코딩되어 프로세싱 회로부(4)가 아키텍처에 의해 정의된 방식으로 명령어를 프로세싱하도록 제어하는 제어 신호를 생성한다. 예를 들어, 디코더(6)는 디코딩된 명령어의 연산 코드 및 명령어의 임의의 추가적인 제어 필드를 해석하여 프로세싱 회로부(4)로 하여금 산술 연산, 로드/저장 연산 또는 로직 연산과 같은 연산을 수행하기 위해 적절한 하드웨어 유닛을 활성화하게 하는 제어 신호를 생성할 수 있다. 장치는 프로세싱 회로부(4)에 의해 처리될 데이터 값 및 프로세싱 회로부의 동작을 구성하기 위한 제어 정보를 저장하기 위한 레지스터(10)의 세트를 갖는다. 산술 또는 로직 명령어에 응답하여, 프로세싱 회로부(4)는 레지스터(10)로부터 피연산자를 판독하고, 명령어의 결과를 레지스터(10)에 다시 기입한다. 로드/저장 명령어에 응답하여, 데이터 값이 프로세싱 회로부를 통해 레지스터(10)와 메모리 시스템(8) 사이에서 전송된다. 메모리 시스템(8)은 메인 메모리뿐만 아니라 하나 이상의 캐시 레벨을 포함할 수 있다.

[0092] 레지스터(10)는 단일 데이터 요소를 포함하는 스칼라 값을 저장하기 위한 다수의 스칼라 레지스터를 포함하는 스칼라 레지스터(12)를 포함한다. 명령어 디코더(6) 및 프로세싱 회로부(4)에 의해 지원되는 일부 명령어는 스칼라 레지스터(12)로부터 판독된 스칼라 피연산자를 처리하여 스칼라 레지스터에 다시 기입되는 스칼라 결과를 생성하는 스칼라 명령어이다.

[0093] 레지스터(10)는 또한 다수의 데이터 요소를 포함하는 벡터 값을 각각 저장하기 위한 다수의 벡터 레지스터를 포함하는 벡터 레지스터 파일(14)을 포함한다. 벡터 명령어에 응답하여, 명령어 디코더(6)는 프로세싱 회로부(4)를 제어하여, 벡터 레지스터(14) 중 하나로부터 판독된 벡터 피연산자의 각자의 요소에 대해 벡터 프로세싱

의 다수의 레인을 수행하여, 스칼라 레지스터(12)에 기입될 스칼라 결과 또는 벡터 레지스터(14)에 기입될 추가적인 벡터 결과를 생성한다. 일부 벡터 명령어는 하나 이상의 스칼라 피연산자로부터 벡터 결과를 생성할 수 있거나, 스칼라 레지스터 파일 내의 스칼라 피연산자에 대해 추가적인 스칼라 연산을 수행할 수 있을 뿐만 아니라, 벡터 레지스터 파일(14)로부터 판독된 벡터 피연산자에 대한 벡터 프로세싱 레인을 수행할 수 있다. 따라서, 일부 명령어는 하나 이상의 소스 레지스터 및 명령어의 목적지 레지스터 중 적어도 하나가 벡터 레지스터(14)이고 하나 이상의 소스 레지스터 및 목적지 레지스터 중 다른 하나가 스칼라 레지스터(12)인 혼합-스칼라-벡터 명령어일 수 있다. 벡터 명령어는 또한 벡터 로드/저장 명령어를 포함할 수 있으며, 이는 벡터 레지스터(14)와 메모리 시스템(8) 내의 위치 사이에서 데이터 값이 전송되게 한다. 로드/저장 명령어는 메모리 내의 위치가 어드레스의 연속적인 범위에 해당하는 연속적인 벡터 로드/저장 명령어를 포함할 수 있거나, 다수의 불연속 어드레스를 지정하고, 프로세싱 회로부(4)를 제어하여 이들 어드레스 각각으로부터 벡터 레지스터의 각자의 요소로 데이터를 로드하거나 벡터 레지스터의 각자의 요소로부터 불연속 어드레스로 데이터를 저장하도록 하는 분산/수집 유형 벡터 로드/저장 명령어를 포함할 수 있다.

[0094] 프로세싱 회로부(4)는 상이한 데이터 요소 크기의 범위를 갖는 벡터의 프로세싱을 지원할 수 있다. 예를 들어, 128-비트 벡터 레지스터(14)는 16개의 8-비트 데이터 요소, 8개의 16-비트 데이터 요소, 4개의 32-비트 데이터 요소 또는 2개의 64-비트 데이터 요소로 분할될 수 있다. 레지스터 뱅크(10) 내의 제어 레지스터는 현재 데이터 요소 크기를 명시하는 데 사용될 수 있거나, 대안적으로 이는 실행될 주어진 벡터 명령어의 파라미터일 수 있다.

[0095] 레지스터(10)는 또한 프로세싱 회로부(4)의 프로세싱을 제어하기 위한 다수의 제어 레지스터를 포함한다. 예를 들어, 이들은 처리되는 현재 실행 포인트에 해당하는 명령어의 어드레스를 나타내는 프로그램 카운터 어드레스를 저장하기 위한 프로그램 카운터 레지스터(16), 함수 호출의 처리에 후속하여 프로세싱이 지시될 복귀 어드레스를 저장하기 위한 링크 레지스터(18), 스택 데이터 구조의 메모리 시스템(8) 내의 위치를 나타내는 스택 포인터 레지스터(20), 및 아래에서 더 상세히 설명되는 박동 상태 정보를 저장하기 위한 박동 상태 레지스터(22)를 포함할 수 있다. 이들은 단지 저장될 수 있는 제어 정보의 유형 중 일부이고, 실제로 주어진 아키텍처의 명령어 세트는 아키텍처에 의해 정의된 바와 같은 많은 다른 제어 파라미터를 저장할 수 있다는 것이 이해될 것이다. 예를 들어, 제어 레지스터는 벡터 레지스터의 전체 폭, 또는 주어진 벡터 프로세싱의 인스턴스에 사용되는 현재 데이터 요소 크기를 명시할 수 있다.

[0096] 프로세싱 회로부(4)는 상이한 명령어 클래스를 프로세싱하기 위한 다수의 별개의 하드웨어 블록을 포함할 수 있다. 예를 들어, 메모리 시스템(8)과 상호작용하는 로드/저장 명령어는 전용 로드/저장 유닛에 의해 처리될 수 있는 반면, 산술 또는 로직 명령어는 산술 로직 유닛(ALU)에 의해 처리될 수 있다. ALU 자체는 곱셈을 수반하는 연산을 수행하기 위한 곱셈-누산 유닛(MAC) 및 다른 종류의 ALU 연산을 프로세싱하기 위한 추가 유닛으로 추가로 분할될 수 있다. 부동 소수점 명령어를 처리하기 위해 부동 소수점 유닛도 제공될 수 있다. 어떠한 벡터 프로세싱도 수반하지 않는 순수 스칼라 명령어는 또한 벡터 명령어와 비교하여 별개의 하드웨어 블록에 의해 처리되거나 동일한 하드웨어 블록을 재사용할 수 있다.

[0097] 디지털 신호 처리(DSP)와 같은 일부 응용에서는 거의 동일한 수의 ALU 및 로드/저장 명령어가 있을 수 있으므로, MAC과 같은 일부 대형 블록은 상당한 양의 시간 동안 유휴 상태로 남겨질 수 있다. 이러한 비효율성은 벡터 아키텍처에서 악화될 수 있는데, 이는 더 높은 성능을 얻기 위해 실행 자원이 벡터 레인의 수에 따라 스케일링되기 때문이다. 더 작은 프로세서(예를 들어, 단일 발행, 순차 코어)에서는 완전히 스케일링된 벡터 파이프라인의 면적 오버헤드가 엄청나게 클 수 있다. 이용 가능한 실행 자원을 더 잘 사용하면서 면적 영향을 최소화하는 한 가지 접근법은 도 2에 도시된 바와 같이 명령어의 실행을 중첩시키는 것이다. 이 예에서, 세 개의 벡터 명령어는 로드 명령어 VLDR, 곱셈 명령어 VMUL 및 시프트 명령어 VSHR을 포함하며, 이러한 모든 명령어는 이들 간에 데이터 종속성이 있더라도 동시에 실행될 수 있다. 이는 VMUL의 요소 1이 Q1 레지스터 전체가 아니라 Q1의 요소 1에만 의존하므로 VLDR의 실행이 완료되기 전에 VMUL의 실행이 시작될 수 있기 때문이다. 명령어가 중첩되는 것을 허용함으로써, 곱셈기와 같은 고가의 블록을 더 많은 시간 동안 활성 상태로 유지할 수 있다.

[0098] 따라서, 마이크로아키텍처 구현이 벡터 명령어의 실행을 중첩시킬 수 있게 하는 것이 바람직할 수 있다. 그러나 아키텍처가 고정된 양의 명령어 중첩이 있다고 가정하면, 마이크로아키텍처 구현이 실제로 아키텍처에 의해 가정되는 명령어 중첩의 양과 매칭되는 경우에는 높은 효율성을 제공할 수 있지만, 상이한 중첩을 사용하거나 전혀 중첩시키지 않는 상이한 마이크로아키텍처로 스케일링되는 경우에는 문제가 발생할 수 있다.

[0099] 대신, 아키텍처는 도 3의 예에 도시된 바와 같이 상이한 중첩의 범위를 지원할 수 있다. 벡터 명령어의 실행은 "박동(beat)"으로 지칭되는 부분으로 분할되며, 각각의 박동은 미리 결정된 크기의 벡터의 일부의 처리에 해당한다. 박동은 완전히 실행되거나 전혀 실행되지 않는 벡터 명령어의 원자 부분이며, 부분적으로 실행될 수 없다. 하나의 박동에서 처리되는 벡터 부분의 크기는 아키텍처에 의해 정의되며 벡터의 임의의 분수일 수 있다. 도 3의 예에서 박동은 벡터 폭의 1/4에 해당하는 처리로서 정의되므로 벡터 명령어당 4개의 박동이 있다. 분명히, 이것은 단지 하나의 예에 불과하며, 다른 아키텍처는 2 또는 8과 같은 상이한 수의 박동을 사용할 수 있다. 하나의 박동에 해당하는 벡터의 부분은 처리 중인 벡터의 데이터 요소 크기와 동일한 크기이거나 그보다 크거나 작을 수 있다. 따라서, 요소 크기가 구현마다 다르거나 상이한 명령어 간에 런타임에서 다르더라도, 박동은 벡터 프로세싱의 특정 고정 폭이다. 하나의 박동에서 처리되는 벡터 부분이 여러 데이터 요소를 포함하는 경우, 각자의 요소가 독립적으로 처리되도록 각각의 요소 사이의 경계에서 캐리 신호가 비활성화될 수 있다. 하나의 박동에서 처리되는 벡터 부분이 요소의 일부에만 해당하고 하드웨어가 여러 박동을 병렬로 계산하기에 불충분한 경우, 하나의 박동 프로세싱 동안 생성된 캐리 출력은 2개의 박동의 결과가 함께 데이터 요소를 형성하도록 다음 박동 프로세싱에 캐리 입력으로 입력될 수 있다.

[0100] 도 3에 도시된 바와 같이, 프로세싱 회로(4)의 상이한 마이크로아키텍처 구현은 추상적 아키텍처 클록의 하나의 "틱(tick)"에서 상이한 수의 박동을 실행할 수 있다. 여기서, "틱"은 아키텍처 상태 진전의 단위에 해당한다(예를 들어, 간단한 아키텍처에서 각각의 틱은 다음 명령어를 가리키도록 프로그램 카운터를 업데이트하는 것을 포함하여, 명령어 실행과 관련된 모든 아키텍처 상태를 업데이트하는 경우에 해당할 수 있음). 이 분야의 통상의 기술자는, 파이프라이닝과 같은 공지된 마이크로아키텍처 기술은 단일 틱이 하드웨어 레벨에서 수행하기 위해 다수의 클록 사이클을 필요로 할 수 있다는 것을 의미할 수 있고, 실제로 하드웨어 레벨에서의 단일 클록 사이클은 다수의 명령어의 다수의 부분을 처리할 수 있다는 것을 이해할 것이다. 그러나, 이러한 마이크로아키텍처 기술은 소프트웨어에 대해 가시적이지 않은데, 이는 틱이 아키텍처 레벨에서 원자적이기 때문이다. 간결성을 위해, 마이크로아키텍처는 본 개시에 대한 추가 설명 동안 무시된다.

[0101] 도 3의 하부 예에 도시된 바와 같이, 일부 구현은 하나의 틱 내에서 모든 박동을 병렬로 처리하기 위한 충분한 하드웨어 자원을 제공함으로써 동일한 틱에서 벡터 명령어의 4개의 박동 모두를 스케줄링할 수 있다. 이는 더 높은 성능 구현에 적합할 수 있다. 이 경우, 전체 명령어를 하나의 틱에서 완료할 수 있으므로 아키텍처 레벨에서 명령어 간의 어떠한 중첩도 필요하지 않다.

[0102] 반면에, 더 면적 효율적인 구현은 틱당 2개의 박동만을 처리할 수 있는 더 좁은 처리 유닛을 제공할 수 있으며, 도 3의 중간 예에 도시된 바와 같이, 명령어 실행은 제1 명령어의 제3 또는 제4 박동과 병렬로 수행되는 제2 벡터 명령어의 제1 및 제2 박동과 중첩될 수 있으며, 이러한 명령어는 프로세싱 회로부 내의 상이한 실행 유닛에서 실행된다(예를 들어, 도 3에서, 제1 명령어는 로드/저장 유닛을 사용하여 실행되는 로드 명령어이고 제2 명령어는 MAC을 사용하여 실행되는 곱셈 누산 명령어임).

[0103] 훨씬 더 에너지/면적 효율적인 구현은 더 좁고 한 번에 하나의 박동만을 처리할 수 있는 하드웨어 유닛을 제공할 수 있으며, 이 경우, 도 3의 상부 예에 도시된 바와 같이 명령어 실행이 하나의 박동만큼 중첩되고 스태거링하면서 틱당 하나의 박동이 처리될 수 있다(이는 위의 도 2에 도시된 예와 동일함).

[0104] 도 3에 도시된 중첩은 일부 예에 불과하며, 다른 구현도 가능하다는 것을 이해할 것이다. 예를 들어, 프로세싱 회로부(4)의 일부 구현은 동일한 틱에서 병렬로 여러 명령어를 이중 발행하는 것을 지원할 수 있으므로, 명령어의 더 큰 처리량이 존재한다. 이 경우, 한 사이클에서 함께 시작되는 두 개 이상의 벡터 명령어는 다음 사이클에서 시작되는 두 개 이상의 벡터 명령어와 일부 박동이 중첩될 수 있다.

[0105] 상이한 성능 포인트로 스케일링하기 위해 구현마다 중첩의 양을 변경할 뿐만 아니라, 벡터 명령어 간의 중첩의 양도 프로그램 내의 벡터 명령어의 실행의 상이한 인스턴스 간에 런타임에 변경될 수 있다. 따라서, 프로세싱 회로부(4)는 이전 명령어에 대한 주어진 명령어의 실행 타이밍을 제어하기 위해 도 1에 도시된 바와 같이 박동 제어 회로부(30)가 제공될 수 있다. 이는 구현하기 더 어렵거나 명령어에 의해 이용 가능한 자원에 의존하는 특정 코너 사례에서 명령어를 중첩시키지 않도록 선택할 수 있는 자유를 마이크로아키텍처에 제공한다. 예를 들어, 동일한 자원을 필요로 하는 주어진 유형(예를 들어, 곱셈 누산)의 연속 명령어가 있고, 이용 가능한 모든 MAC 또는 ALU 자원이 이미 다른 명령어에서 사용되고 있다면, 다음 명령어 실행을 시작하기에 충분한 자유 자원이 없을 수 있으므로, 중첩시키는 것이 아니라, 제1 명령어가 완료될 때까지 제2 명령어의 발행이 대기될 수 있다.

[0106] 실행 벡터 명령어의 상이한 중첩의 범위를 허용하는 것은 성능 포인트의 범위에 걸쳐 하드웨어 자원의 더 효율

적인 이용을 허용할 수 있지만, 이는 예외나 디버그 이벤트 또는 현재 실행 스레드를 중단을 트리거하는 다른 이벤트를 처리하는 데 약간의 복잡성이 발생할 수 있다. 예를 들어, 도 2에 도시된 예에서 네 번째 틱에서 예외가 발생하면 레지스터 파일에 여러 명령어의 부분 업데이트가 포함될 것이다. 이를 처리하는 한 가지 방법은, 예외가 발생하면 되돌아갈 수 있는 추측 상태로 부분 업데이트를 처리하는 것이지만, 이는 데이터가 커밋될 때까지 메모리 시스템(8)에 데이터를 저장하기 위한 저장 요청을 버퍼링하고 추측 상태를 추적하기 위해 하드웨어에 추가 레지스터를 제공할 필요가 있을 수 있기 때문에 필요한 하드웨어의 양을 증가시킬 수 있다. 또 다른 접근법은 벡터 명령어 도중에 예외가 발생하는 것을 아예 비활성화하고, 가장 오래된 미완료된 명령어가 완료될 때까지 예외 발생을 지연시키는 것이지만, 예외 처리 지연 시간을 늘리는 것은 바람직하지 않을 수 있으며, 예외가 정확한 오류인 경우 이러한 동작은 오류와 연관된 아키텍처 보장을 깨뜨릴 수 있다.

[0107] 대신에, 도 4에 도시된 바와 같이, 박동 상태 레지스터(22)는 현재의 스레드의 중단을 유도하는 예외, 디버그 이벤트 또는 다른 이벤트의 포인트에서 인접한 명령어의 그룹의 어느 박동이 완료되었는지를 추적하는 박동 상태 값을 기록하는데 사용될 수 있다. 아키텍처에 실행의 중첩 특성을 노출시킴으로써, 마이크로아키텍처 복잡성을 감소시키고 전력 및 면적 효율성을 증가시키는데 도움이 될 수 있다.

[0108] 도 4의 예에서, 박동 상태 정보는 3개의 벡터 명령어 A, B, C의 그룹의 완료된 박동을 추적하며, 여기서 명령어 A는 가장 오래된 미완료된 벡터 명령어에 해당하고, 명령어 B는 명령어 A 이후의 다음 벡터 명령어이고, 명령어 C는 명령어 B 이후의 다음 벡터 명령어이다. 표기법 Ax는 명령어 A의 x번째 박동을 나타내며, 여기서 x는 4-박동 벡터 구현의 경우 1~4 사이이며, 예를 들어, A2는 명령어 A의 제2 박동이다. 도 4는 박동 상태 정보를 사용하여 3개의 명령어가 추적되는 예를 도시하지만, 주어진 포인트에서 더 많은 수의 명령어가 부분적으로 완료될 수 있게 하는 다른 예에서, 박동 상태 정보는 더 많은 수의 명령어를 추적할 수 있다. 예를 들어, 이중 발행이 지원되는 경우, 3개 이상의 명령어에 대한 박동 진행 상황을 나타내는 것이 바람직할 수 있다. 박동 상태 필드의 각 값은 완료된 박동의 주어진 조합에 할당된다. 예를 들어, 박동 상태 값 0011은 명령어 A의 제1 및 제2 박동과 명령어 B의 제1 박동이 완료되었음을 나타낸다. 명령어의 각자의 그룹의 박동의 특정 세트에 대한 박동 상태 정보의 특정 인코딩된 값의 특정 매핑은 임의적이고 변경될 수 있다. 이 예에서 박동 상태 값 0000은 불완전한 명령어가 없고, 따라서 불완전한 명령어의 완료된 박동이 없다는 것을 나타낸다. 이것은 예를 들어, 프로세서가 스칼라 명령어를 실행했을 때 발생할 수 있다.

[0109] 도 5는 현재 실행의 스레드가 중단된 시점에서 기록된 박동 상태 정보의 일부 예를 보여준다. 도 5의 상단 예시에서 벡터 명령어는 틱당 1개의 박동과 함께 실행되고, 제4 틱에서 디버그 이벤트 또는 예외가 발생한다. 따라서, 이 시점에서, 명령어 A의 제1 세 박동, 명령어 B의 제1 두 박동 및 명령어 C의 제1 박동은 이미 완료되었지만, 박동 A4, B3, C2, D1은 아직 수행되어야 한다. 따라서, 박동 상태 정보는 도 4의 예에 따라 박동 A1, A2, A3, B1, B2 및 C1이 이미 완료되었음을 나타내는 값 0111을 가질 것이다.

[0110] 마찬가지로, 도 5의 예의 하단에서, 실행되는 명령어는 명령어 B 및 C가 (예를 들어, 동일한 하드웨어 유닛의 사용을 필요로 하기 때문에) 중첩될 수 없고, 따라서 이번에는 명령어 C 및 D가 디버그 이벤트 또는 예외 발생 시점에 아직 시작되지 않았다. 이번에는 틱 4에서 발생하는 예외가 박동 A1, A2, A3, B1 및 B2는 이미 완료되었지만 C1은 완료되지 않았음을 나타내는 박동 상태 정보 0110의 기록을 트리거할 것이다.

[0111] 마찬가지로, 도 3의 틱당 2개의 박동을 예로 들면, 틱 2에서 예외가 발생하면 박동 A1과 박동 A2만 완료되었을 것이며 박동 상태 값은 0010이 된다. 박동 상태 정보의 값 0001 및 0010은 단지 하나의 명령어 A가 예외 발생 시에 부분적으로 완료되었음을 나타내지만, 박동 상태 정보는 다수의 명령어의 그룹 중 어느 박동이 완료되었는지를 여전히 나타내는데, 이는 다음 2개의 명령어 B, C의 박동 중 어느 것도 완료되지 않았음을 식별하기 때문이다.

[0112] 도 3의 틱당 4개의 박동 예시에서, 박동 상태 값은 각 명령어가 하나의 틱 내에서 완료되므로 예외 시점에 부분적으로 완료된 명령어가 없기 때문에 예외가 언제 발생하는지에 관계없이 0000이 될 것이다.

[0113] 디버그 이벤트 또는 예외가 발생하면, 복귀 어드레스는 가장 오래된 미완료된 명령어의 어드레스를 나타내는 프로그램 카운터(16)의 현재 값으로 설정된다. 따라서 도 5의 두 예에서 복귀 어드레스는 명령어 A의 어드레스로 설정된다. 복귀 어드레스는 스택 포인터 레지스터의 값에 대한 스택 상의 위치에 또는 복귀 어드레스 레지스터에 포함하는 다양한 장소에 저장될 수 있다.

[0114] 도 6에 도시된 바와 같이, 이것은 이벤트로부터의 복귀 요청(예를 들어, 디버그 모드 또는 예외 핸들러로부터의 복귀)에 응답하여 프로세서가 박동 상태 레지스터(22)의 복귀 어드레스 및 박동 상태 정보에 기초하여 결정된

포인트로부터 프로세싱을 재개할 수 있게 한다. 이벤트로부터의 복귀 요청은 디버그 이벤트의 경우 디버거에서, 또는 예외 이벤트의 경우 예외 핸들러에서 수행할 수 있다. 이벤트로부터의 복귀 요청 이후, 처리될 명령어의 페칭은 이 경우 명령어 A에 해당하는 복귀 어드레스에 의해 표시된 어드레스로부터 재개된다. 명령어 B, C 및 D가 이어진다(이 예는 도 5의 상단 예에 해당됨). 그러나, 복귀 후 처음 몇 사이클 동안, 이미 완료된 것으로 박동 상태 정보에 의해 표시된 임의의 박동은 억제된다. 프로세서는 해당 프로세싱 동작이 전혀 수행되지 않도록 하여 이러한 박동을 억제할 수 있다(예를 들어, 데이터를 로드 또는 저장하기 위한 요청을 억제하거나 ALU 또는 MAC의 비활성화). 대안적으로, 연산은 ALU 연산의 경우에 여전히 수행될 수 있지만, 프로세서는 레지스터 상태에 영향을 미치지 않도록 연산의 결과의 기록을 억제(즉, 목적지 벡터 레지스터의 부분의 업데이트를 억제)할 수 있다. 제4 틱에 도달하면 파이프라인은 이전에 디버그 이벤트나 예외가 발생한 지점에 도달한 다음 프로세싱이 정상적으로 계속된다. 따라서, 예외 복귀 이후의 처음 몇 사이클 동안, 프로세서는 어떤 유용한 작업도 수행하지 않을 수 있고, 본질적으로 원래의 예외 또는 디버그 이벤트가 발생했을 때 진행 중이었던 다수의 명령어를 단지 재페칭하는 것이다. 그러나, 예외 복귀 레이턴시는 종종 일부 애플리케이션에 대해 중요하지 않기 때문에, 이는 예외를 취할 때의 레이턴시를 감소시키기 위한 양호한 트레이드 오프일 수 있고, 또한 미완료된 명령어의 결과를 추측적으로 저장할 필요가 없기 때문에 예외에 저장되어야 하는 아키텍처 상태의 양을 감소시키는 데 도움이 된다. 이 접근법은 또한 벡터 명령어의 박동에 의해 발생한 정확한 오류인 예외의 처리를 가능하게 한다.

[0115] 일부 경우에, 다수의 명령어의 그룹의 완료된 박동을 나타내는 박동 상태 정보는 디버그 이벤트 또는 예외 발생에 응답하여 설정될 수 있다. 그러나, 일부 구현에서, 예외가 발생했는지 여부에 관계없이, 명령어가 완료될 때마다 박동 상태 레지스터를 업데이트하는 것이 더 쉬울 수 있어서, 예외가 다음 틱에서 발생하면, 박동 상태 레지스터(22)는 이미 명령어의 그룹의 이미 완료된 박동을 표시한다.

[0116] 도 4는 박동 상태 정보의 인코딩의 하나의 예를 도시하지만, 또 다른 가능성은 박동 상태 정보를 명령어 그룹 A, B, C 등 중 하나의 박동에 각각 해당하는 다수의 비트를 포함하는 비트맵으로서 제공하는 것이며, 각각의 비트는 해당 박동이 완료되었으면 1로 설정되고, 해당 박동이 완료되지 않으면 0으로 설정된다(또는 그 반대). 그러나, 실제로, 주어진 명령어의 나중 박동은 더 이른 박동이 아직 완료되지 않은 경우 완료될 수 없기 때문에, 모든 박동에 대해 박동을 제공할 필요가 없고, 도 4의 예에서와 같이, 더 작은 비트 필드의 특정 인코딩을 완료된 박동의 특정 조합에 할당하는 것이 더 효율적일 수 있다.

[0117] 도 7은 본 기술의 다양한 구성에 따라 배열된 장치(30)의 세부 사항을 개략적으로 도시한다. 특히, 장치(30)에는 디코더 회로부(38), 프로세싱 회로부(40) 및 레지스터 세트(32)가 제공된다. 레지스터(32)는 하나 이상의 스칼라 레지스터(34) 및 하나 이상의 벡터 레지스터(36)를 포함한다. 디코더 회로부는 (예를 들어, 프로그래머 또는 컴파일러에 의해 생성된 프로그램 코드에 기초하여) 명령어를 수신하고 명령어 세트 아키텍처에 기초하여 명령어를 해석하도록 배열된다. 특히, 디코더 회로부는 제1 소스 벡터 레지스터(44), 제2 소스 벡터 레지스터(46), 목적지 레지스터(54) 및 제어 파라미터(43)를 명시하는 벡터 추출 및 병합 명령어를 해석하도록 배열된다. 디코더 회로부는, 벡터 추출 및 병합 명령어의 수신 시에, 프로세싱 회로부(40)가 벡터 추출 및 병합 프로세싱을 수행하도록 제어 신호를 생성한다. 프로세싱 회로부(40)는 복수의 박동 프로세싱 중 하나 이상의 박동(48)을 수행함으로써 벡터 추출 및 병합 프로세싱을 수행하도록 제어 신호에 응답한다. 각각의 박동 프로세싱은 적어도 제1 소스 벡터 레지스터(44) 및 목적지 벡터 레지스터(54) 각각의 부분에 해당한다. 프로세싱 회로부(40)는 제1 소스 벡터 레지스터(44)의 하나 이상의 부분(48), 제2 소스 벡터 레지스터의 하나 이상의 부분(49)에 해당하는 하나 이상의 박동 프로세싱을 수행하여 목적지 벡터 레지스터(50)에 저장될 하나 이상의 부분(50)을 생성하도록 배열된다. 프로세싱 회로부(40)는 복수의 박동 프로세싱 중 K번째 박동 프로세싱이 제1 소스 벡터 레지스터(48)의 K번째 부분으로부터 하나 이상의 비트를 추출하고 해당 비트를 하나 이상의 추가 비트와 연결하도록 배열된다. K번째 박동이 복수의 박동 중 제1 박동인 경우, 하나 이상의 추가 비트는 제2 소스 벡터 레지스터(49)의 제1 부분(K=1인 경우 K번째 부분)으로부터 추출된다. K번째 박동이 제1 박동(K>1) 이외의 박동인 경우, 하나 이상의 추가 비트는 제1 소스 벡터 레지스터(44)의 (K-1)번째 부분에 해당하는 (K-1)번째 박동으로부터 전달된 캐리 비트(52)이다. 또한, 프로세싱 회로부는 K번째 박동이 복수의 박동 중 마지막 박동이 아닌 경우, (K+1)번째 박동 프로세싱에서 사용될 캐리 데이터로서 하나 이상의 비트를 출력하도록 배열된다.

[0118] 도 8은 본 기술의 일부 구성에 따라 배열된 프로세싱 장치(60)의 세부 사항을 개략적으로 도시한다. 특히, 프로세싱 장치(60)에는 레지스터(62), 디코더 회로부(68), 프로세싱 회로부(70) 및 데이터 제어 회로부(72)가 제공된다. 레지스터(62)는 복수의 스칼라 레지스터(64) 및 복수의 벡터 레지스터(66)를 포함한다. 디코더 회로부(68)는 명령어 세트 아키텍처의 부분을 형성하는 명령어에 응답하여 제어 신호를 생성하도록 배열된다. 제어

신호는 프로세싱 회로부(70) 및 데이터 제어 회로부(72)로 전달(라우팅)된다. 프로세싱 회로부(70)는 벡터 추출 및 병합 명령어에 응답하여 복수의 박동 프로세싱을 수행하도록 배열된다. 프로세싱 회로부의 세부 사항은 도 7에서 언급된 프로세싱 회로부(40)의 세부 사항과 동일하다. 데이터 제어 회로부(72)는 데이터 전송 명령어에 응답하여 디코더 회로부(68)에 의해 생성되는 데이터 제어 신호에 응답하여, 복수의 박동의 메모리 전송 프로세싱을 수행한다. 주어진 틱의 경우, 장치(60)는 데이터 제어 회로부(72)에 의해 수행되는 복수의 박동의 메모리 전송 프로세싱의 제1 서브세트, 및 프로세싱 회로부(70)에 의해 수행되는 벡터 추출 및 병합 명령어에 응답하여 복수의 박동의 조합 프로세싱의 제2 서브세트를 포함하는 복수의 박동을 수행하도록 배열된다. 장치(60)는 동일한 벡터 레지스터(72)의 비-중첩 부분을 참조하면서 복수의 박동의 제1 서브세트 및 복수의 박동의 제2 서브세트를 수행하도록 배열된다.

[0119] 도 9는 본 기술의 일부 구성에 따른 벡터 추출 및 병합 명령어의 세부 사항을 개략적으로 도시한다. 벡터 추출 및 병합 명령어는 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 목적지 벡터 레지스터 및 제어 파라미터 M을 명시한다. 도시된 예에서, 프로세싱 회로부는 2개의 박동 프로세싱을 수행하며, 각각은 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터 및 목적지 레지스터의 N-비트 부분에 해당한다. 제1 소스 벡터 레지스터는 제1 N-비트 부분(82)을 포함한다. 제1 N-비트 부분(82)은 최상위 M 비트(84) 및 최하위 N-M 비트(86)를 포함한다. 프로세싱 회로부는, 제1 소스 벡터 레지스터(82)의 제1 부분, 제2 소스 벡터 레지스터(88)의 제1 부분, 및 목적지 벡터 레지스터(102)의 제1 부분에 해당하는 제1 박동 프로세싱에 대해, 제1 소스 벡터 레지스터(82)의 제1 부분의 N-M 비트(86)를 추출하고, 추출된 N-M 비트를 제2 소스 벡터 레지스터(88)의 제1 부분으로부터 추출되는 M-비트(하나 이상의 추가 비트)(90)와 연결하도록 배열된다. 구체적으로, 제1 소스 벡터 레지스터(82)의 제1 부분으로부터 추출된 N-M 비트(86)는 목적지 벡터 레지스터(102)의 제1 부분의 최상위 N-M 비트(98)로서 저장된다. 제2 소스 벡터 레지스터(88)의 제1 부분으로부터 추출된 M-비트(90)는 목적지 벡터 레지스터(102)의 제1 부분의 최하위 M-비트(100)로서 저장된다. 프로세싱 회로부는 또한 제1 소스 벡터 레지스터(82)의 제1 부분의 최상위 M-비트(84)를 캐리 비트(96)로서 전달하도록 구성된다. 캐리 비트는 병렬로 실행되는 박동 프로세싱 사이에서 전달되거나 병렬로 실행되지 않는 박동 프로세싱 사이에서 비트를 전달하도록 배열된 스칼라 레지스터에 출력되는 캐리 비트일 수 있다. 제2 박동 프로세싱에서, 제1 소스 벡터 레지스터(82)의 제1 부분으로부터 전달되는 M-비트(96)는 목적지 레지스터의 제2 부분의 최하위 M-비트(94)로서 저장된다. 제2 박동 프로세싱 동안, 프로세싱 회로부는 제1 소스 벡터 레지스터(80)의 제2 N-비트 부분의 최하위 N-M 비트(95)를 추출하고, 제1 소스 벡터 레지스터(80)의 제2 부분의 N-M 비트(95)를 목적지 벡터 레지스터(104)의 제2 부분의 최상위 N-M 비트(92)로서 저장하도록 배열된다. 이러한 방식으로, 프로세싱 회로부는 복수의 박동에 걸쳐 벡터 추출 및 병합 명령어를 지원한다. 이 예에서, 제어 파라미터는 부분(하나 이상의 추가 비트) 사이에서 전달할 M 비트(84)의 수를 나타낸다. 다른 예에서, 제어 파라미터는 제1 소스 벡터 레지스터(86)의 제1 부분으로부터 추출되고 목적지 벡터 레지스터의 제1 부분에 저장될 비트의 수를 나타낼 수 있다.

[0120] 도 10 내지 도 12는 본 기술의 다양한 구성에 따라 제2 소스 벡터 레지스터의 제1 부분으로부터 추출되는 비트를 개략적으로 도시한다. 벡터 추출 및 병합 명령어의 특정 사용 사례는 32-비트 경계에 정렬되지 않은 벡터를 생성하는 것이다. 특히, 일부 장치는 32-비트 경계에 정렬된 데이터를 로딩하도록 배열된다. 따라서, 32-비트만큼 시프트되는 데이터 값의 벡터를 생성하는 것은 비교적 간단하다. 그러나, 32-비트 경계와 정렬되지 않은 데이터를 생성하는 것은 로드 명령어만을 사용해서는 가능하지 않을 수 있거나 성능 패널티를 초래할 수도 있으므로 정렬된 로드를 사용하는 것이 바람직할 수도 있다. 32-비트 경계와 정렬되지 않은 데이터를 생성하기 위한 하나의 접근법은 수행될 시프트를 요구한다.

[0121] 도 10은 명시된 레지스터에 저장된 데이터가 16-비트 데이터인 경우를 개략적으로 도시한다. 도시된 제1 소스 벡터 레지스터는 각각 4바이트(32비트)를 포함하는 4개의 박동으로 분할된다. 제1 및 제2 소스 벡터 레지스터에 저장되는 데이터는 동일한 데이터 세트의 상이한 부분에 해당한다. 제2 소스 벡터 레지스터에 저장되는 데이터는 제1 소스 벡터 레지스터에 로딩된 데이터로부터 32-비트만큼 오프셋되었다. 16-비트 데이터의 경우, 전술한 사용 사례에 따라, 16비트만큼 시프트된 벡터를 생성하는 것이 바람직하다. 이러한 상황에서, 제2 소스 벡터 레지스터로부터 추출되는 하나 이상의 추가 비트는 제2 소스 벡터 레지스터의 제1 부분의 바이트 2 및 바이트 3(비트16 내지 비트31)이다. 제2 소스 벡터 레지스터의 예시된 부분으로부터 이들 비트를 제1 소스 벡터 레지스터에 저장된 시프트된 데이터와 함께 추출하는 조합은 32-비트 경계에 정렬되지 않은 목적지 벡터 레지스터에서의 데이터의 생성을 초래한다.

[0122] 도 11은 8-비트 데이터에 대해 이러한 시프트를 수행하기 위해 추출될 필요가 있는 제2 소스 벡터 레지스터의 부분을 개략적으로 도시한다. 특히, 24비트만큼 32-비트 경계와 정렬되지 않는 데이터 세트를 생성하기 위해,

제2 소스 벡터 레지스터의 바이트 1, 바이트 2, 및 바이트 3은 제1 박동 프로세싱에서의 하나 이상의 추가 비트로서 추출된다. 16비트만큼 32-비트 경계와 정렬되지 않는 데이터 세트를 생성하기 위해, 제2 소스 벡터 레지스터의 바이트 2 및 바이트 3은 제1 박동 프로세싱에서의 하나 이상의 추가 비트로서 추출된다. 8비트만큼 32-비트 경계와 정렬되지 않는 데이터 세트를 생성하기 위해, 제2 벡터 레지스터의 바이트 3은 제1 박동 프로세싱에서의 하나 이상의 추가 비트로서 추출된다. 이러한 방식으로, 32-비트 경계에 정렬되지 않은 데이터 요소를 갖는 벡터의 시퀀스를 생성하는 것이 가능하다.

[0123] 도 12는 목적지 데이터 벡터가 제2 소스 데이터 벡터인 경우, 8-비트 데이터에 대한 그러한 시프트를 수행하기 위해 추출될 필요가 있는 제2 소스 벡터 레지스터의 부분을 개략적으로 도시한다. 도시된 예에서, 3개의 벡터 추출 및 병합 명령어의 시퀀스가 적용된다. 벡터 추출 및 병합 명령어 각각은 제어 파라미터로서, 제1 소스 벡터 레지스터를 시프트할 상이한 수의 비트를 명시한다. 도 10의 예에서와 같이, 제2 소스 벡터 레지스터에 저장되는 데이터는 제1 소스 벡터 레지스터에 로딩된 데이터로부터 32-비트만큼 오프셋되었다. 도시된 예에서, 제2 소스 벡터 레지스터로부터 추출되는 하나 이상의 추가 비트는 최하위 바이트를 제외한 제2 소스 벡터 레지스터의 제1 부분의 최하위 바이트 세트를 포함한다. 제1 벡터 추출 및 병합 명령어에서, 24비트(3바이트)의 시프트가 제어 파라미터로서 정의된다. 결과적으로, 제2 소스 벡터 레지스터로부터 추출되는 바이트는 바이트 3, 바이트 2 및 바이트 1이다. 이들은 제1 벡터 추출 및 병합 명령어에 대한 목적지 벡터 레지스터의 콘텐츠로서, 32-비트 경계로부터 24비트만큼 오정렬된 값의 벡터를 생성하기 위해 복수의 박동 프로세싱 동안 프로세싱 회로부에 의해 연결된다. 제2 벡터 추출 및 병합 명령어에서, 16비트(2바이트)의 시프트가 제어 파라미터로서 정의되고, 제1 벡터 추출 및 병합 명령어의 목적지 벡터 레지스터가 제2 소스 벡터 레지스터로서 사용된다. 결과적으로, 제2 명령어의 제2 소스 벡터 레지스터로부터 추출되는 바이트는 바이트 3 및 바이트 2이다. 이들은 제2 벡터 추출 및 병합 명령어에 대한 목적지 벡터 레지스터의 콘텐츠로서, 32-비트 경계로부터 16비트만큼 오정렬된 값의 벡터를 생성하기 위해 복수의 박동 프로세싱 동안 프로세싱 회로부에 의해 연결된다. 제3 벡터 추출 및 병합 명령어에서, 8비트(1바이트)의 시프트가 제어 파라미터로서 정의되고, 제2 벡터 추출 및 병합 명령어의 목적지 벡터 레지스터가 제2 소스 벡터 레지스터로서 사용된다. 결과적으로, 제2 소스 벡터 레지스터로부터 추출되는 바이트는 바이트 3이다. 이 바이트는 제3 벡터 추출 및 병합 명령어에 대한 목적지 벡터 레지스터의 콘텐츠로서, 32-비트 경계로부터 8비트만큼 오정렬된 값의 벡터를 생성하기 위해 복수의 박동 프로세싱 동안 프로세싱 회로부에 의해 연결된다.

[0124] 도 13 내지 도 17은 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 연산의 시퀀스를 개략적으로 도시한다. 예시적인 목적으로, 벡터 레지스터의 요소는 벡터 추출 및 병합 명령어가 32-비트 경계에 정렬되지 않은 벡터를 생성하는 사용 경우에 대해 선택되었다. 이 사용 사례 예는 단지 예시적인 목적으로 선택되었으며, 본원에 설명된 기술이 제1 소스 벡터 레지스터와 제2 소스 벡터 레지스터의 콘텐츠 사이에 임의의 관계가 있을 것을 요구하지 않는다는 것이 당업자에게 쉽게 명백할 것이다. 특히, 본원에 설명된 일반 벡터 추출 및 병합 명령어의 경우, 제1 소스 벡터 레지스터에 저장되는 벡터는 메모리로부터 로딩되거나, 예를 들어, 하나 이상의 다른 연산의 결과로서 생성된 임의의 제1 벡터일 수 있다는 것이 명백할 것이다. 마찬가지로, 제2 소스 벡터 레지스터에 저장된 제2 소스 벡터는 임의의 제2 벡터일 수 있고, 일부 사용 경우에서, 프로그래머는 제1 소스 벡터 레지스터와 제2 소스 벡터 레지스터에 존재하는 요소 사이에 일부 중첩이 존재하도록 제1 벡터 및 제2 벡터를 선택하도록 선택할 수 있다. 다른 사용 경우에서, 프로그래머는 제1 소스 벡터 및 제2 소스 벡터를 선택하여 제1 소스 벡터 레지스터와 제2 소스 벡터 레지스터에 존재하는 요소 사이에 중첩이 존재하지 않도록 선택할 수 있다.

[0125] 도 13은 제1 소스 벡터 레지스터(110), 제2 소스 벡터 레지스터(112), 목적지 레지스터(114), 스칼라 레지스터 및 제어 정보를 명시하는 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 연산의 시퀀스를 개략적으로 도시한다. 제1 소스 벡터 레지스터(110), 제2 소스 벡터 레지스터(112) 및 목적지 레지스터(114) 각각은 복수의 박동 프로세싱에서 처리될 복수의 부분으로서 배열된다. 예시된 예에서, 프로세싱 회로부는 주어진 틱에 대한 단일 박동 프로세싱을 수행하도록 배열된다. 각 부분은 2개의 요소를 포함하고, 제어 정보는 단일 요소에 해당하는 시프트가 수행되어야 함을 명시한다. 단지 예시적인 목적으로만, 제1 소스 벡터 레지스터 및 제2 소스 벡터 레지스터는 128비트 벡터 레지스터이고, 넘버링된 데이터 아이템의 세트를 포함하는 것으로 예시된다. 특히, 제1 소스 벡터 레지스터는 데이터 아이템 9에서 데이터 아이템 2까지를 포함하고, 제2 소스 벡터 레지스터는 데이터 아이템 7에서 데이터 아이템 0까지를 포함한다. 따라서, 제1 소스 벡터 레지스터 및 제2 소스 벡터 레지스터는 32비트 경계에 정렬되는 메모리의 어드레스로부터 로딩되는 16비트 데이터 아이템을 포함한다. 제1 박동 프로세싱에서, 프로세싱 회로부는 제1 소스 벡터 레지스터(110(D))의 제1 부분의 최하위 요소(데이터 아이템 2)를 추출한다. 제1 소스 벡터 레지스터(110(D))의 제1 부분의 추출된 최하위 요소는

제2 소스 벡터 레지스터(112(D))의 제1 부분의 최상위 요소(데이터 아이템 1)와 연결되고, 연결 결과는 목적지 벡터 레지스터(114(D))의 제1 부분으로서 저장된다. 제1 박동 프로세싱 동안, 제1 소스 벡터 레지스터(110(D))의 제1 부분의 최상위 요소(데이터 아이템 3)는 캐리 데이터(116)로서 추출되고 스칼라 레지스터에 최상위 요소로서 저장된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(110(C))의 제2 부분의 최하위 요소(데이터 아이템 4)를 추출한다. 제1 소스 벡터 레지스터(110(C))의 제2 부분의 추출된 최하위 요소는 스칼라 레지스터의 최상위 요소(데이터 아이템 3)에 저장된 캐리 데이터(116)와 연결되고, 연결 결과는 목적지 벡터 레지스터(114(C))의 제2 부분에 저장된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터에 최상위 요소로서 저장될 캐리 데이터(118)로서 제1 소스 벡터 레지스터(110(C))의 제2 부분의 최상위 요소(데이터 아이템 5)를 추출한다. 제3 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(110(B))의 제3 부분의 최하위 요소(데이터 아이템 6)를 추출한다. 제1 소스 벡터 레지스터(110(B))의 제3 부분의 추출된 최하위 요소는 스칼라 레지스터의 최상위 요소(데이터 아이템 5)에 저장된 캐리 데이터(118)와 연결되고, 연결 결과는 목적지 벡터 레지스터(114(B))의 제3 부분에 저장된다. 제3 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터에 최상위 요소로서 저장될 캐리 데이터(120)로서 제1 소스 벡터 레지스터(110(B))의 제3 부분의 최상위 요소(데이터 아이템 7)를 추출한다. 제4 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(110(A))의 제4 부분의 최하위 요소(데이터 아이템 8)를 추출한다. 제1 소스 벡터 레지스터(110(C))의 제4 부분의 추출된 최하위 요소는 스칼라 레지스터의 최상위 요소(데이터 아이템 7)에 저장된 캐리 데이터(120)와 연결되고, 연결 결과는 목적지 벡터 레지스터(114(A))의 제4(마지막) 부분에 저장된다. 일부 대안적인 구성에서, 제4 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터에 최상위 요소로서 저장될 캐리 데이터로서 제1 소스 벡터 레지스터(110(A))의 제4 부분의 최상위 요소를 추출한다. 이 캐리 데이터는 벡터 추출 및 병합 명령어의 실행에 후속하여 스칼라 레지스터에 저장된 상태로 유지된다. 스칼라 레지스터의 미사용 요소(도 13에 표시된 최하위 요소)의 값은 임의적이다. 일부 예에서 이 요소는 0과 같은 더미 값으로 설정될 수 있다. 다른 예에서, 그것은 제1 소스 벡터 레지스터의 현재 부분으로부터의 인접한 요소의 값으로 설정될 수도 있다.

[0126] 도 14는 제1 소스 벡터 레지스터(140), 제2 소스 벡터 레지스터(142), 목적지 레지스터(144), 스칼라 레지스터 및 제어 정보를 명시하는 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 연산의 시퀀스를 개략적으로 도시한다. 도 13에서와 같이, 제1 소스 벡터 레지스터 및 제2 소스 벡터 레지스터 각각은 32-비트 경계에 정렬된 메모리 영역으로부터 추출되는 데이터 아이템을 포함한다. 도 13과 대조적으로, 제1 및 제2 소스 벡터 레지스터의 요소에 저장된 데이터 아이템 각각은 8비트 데이터 아이템이다. 제1 소스 벡터 레지스터(140), 제2 소스 벡터 레지스터(142) 및 목적지 레지스터(144) 각각은 복수의 박동 프로세싱에서 처리될 복수의 부분으로서 배열된다. 예시된 예에서, 프로세싱 회로부는 주어진 틱에 대한 단일 박동 프로세싱을 수행하도록 배열된다. 각 부분은 4개의 요소를 포함하고, 제어 정보는 2개의 요소에 해당하는 시프트가 수행되어야 함을 명시한다. 제1 박동 프로세싱에서, 프로세싱 회로부는 제1 소스 벡터 레지스터(140(D))의 제1 부분의 최하위 2개의 요소(데이터 아이템 5 및 4)를 추출한다. 제1 소스 벡터 레지스터(140(D))의 제1 부분의 추출된 최하위 2개의 요소는 제2 소스 벡터 레지스터(142(D))의 제1 부분의 최상위 2개의 요소(데이터 아이템 3 및 2)와 연결되고, 연결 결과는 목적지 벡터 레지스터(144(D))의 제1 부분으로서 저장된다. 제1 박동 프로세싱 동안, 제1 소스 벡터 레지스터(140(D))의 제1 부분(데이터 아이템 7에서 데이터 아이템 4까지)은 캐리 데이터(146)가 스칼라 레지스터에 저장됨에 따라 추출된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(140(C))의 제2 부분의 최하위 2개의 요소(데이터 아이템 9 및 8)를 추출한다. 제1 소스 벡터 레지스터(140(C))의 제2 부분의 추출된 최하위 2개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(146)의 최상위 2개의 요소(데이터 아이템 7 및 6)와 연결되고, 연결 결과는 목적지 벡터 레지스터(144(C))의 제2 부분에 저장된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 또한 제1 소스 벡터 레지스터(140(C))의 제2 부분(데이터 아이템 11에서 데이터 아이템 8까지)을 스칼라 레지스터에 저장될 캐리 데이터(148)로서 추출한다. 제3 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(140(B))의 제3 부분의 최하위 2개의 요소(데이터 아이템 13 및 12)를 추출한다. 제1 소스 벡터 레지스터(140(B))의 제3 부분의 추출된 최하위 2개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(148)의 최상위 2개의 요소(아이템 11 및 10)와 연결되고, 연결 결과는 목적지 벡터 레지스터(144(B))의 제3 부분에 저장된다. 제3 박동 프로세싱 동안 프로세싱 회로부는 또한 제1 소스 벡터 레지스터(140(B))의 제3 부분(아이템 15에서 데이터 아이템 12까지)을 스칼라 레지스터에 저장될 캐리 데이터(150)로서 추출한다. 제4 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(140(A))의 제4 부분의 최하위 2개의 요소(데이터 아이템 17 및 16)를 추출한다. 제1 소스 벡터 레지스터(140(A))의 제4 부분의 추출된 최하위 2개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(150)의 최상위 2개의 요소(데이터 아이

템 15 및 14)와 연결되고, 연결 결과는 목적지 벡터 레지스터(144(A))의 제4(마지막) 부분에 저장된다. 일부 대안적인 구성에서, 제4 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터에 저장될 캐리 데이터로서 제1 소스 벡터 레지스터(140(A))의 제4 부분을 추출한다. 이 캐리 데이터는 벡터 추출 및 병합 명령어의 실행에 후속하여 스칼라 레지스터에 저장된 상태로 유지된다.

[0127] 도 15는 대안적인 구현에 따른 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 연산의 시퀀스를 개략적으로 도시한다. 도 15는 제1 박동 프로세싱, 제2 박동 프로세싱, 및 제3 박동 프로세싱 각각에 대해, 제1 소스 벡터 레지스터(160)의 해당하는 부분으로부터 추출되어 스칼라 레지스터에 캐리 데이터로서 저장되는 데이터가 해당 부분의 최상위 2개의 요소이고 스칼라 레지스터의 최하위 2개의 요소로서 저장된다는 점에서 도 14와 상이하다. 특히, 이러한 연산은 다음과 같이 도 14와 관련하여 설명된 것과 상이하다: 제1 박동 프로세싱에서 프로세싱 회로부는 제1 소스 벡터 레지스터(160(D))의 제1 부분의 2개의 최상위 요소(데이터 아이템 7 및 6)를 추출하여 스칼라 레지스터의 최하위 2개의 요소에 캐리 데이터(166)로서 저장한다. 제2 박동 프로세싱에서, 데이터의 하나 이상의 추가 비트는 스칼라 레지스터의 2개의 최하위 요소로부터 추출되고, 프로세싱 회로부는 제1 소스 벡터 레지스터(160(C))의 제2 부분의 2개의 최상위 요소(데이터 아이템 11 및 10)를 추출하여 스칼라 레지스터의 최하위 2개의 요소에 캐리 데이터(168)로서 저장한다. 제3 박동 프로세싱에서, 데이터의 하나 이상의 추가 비트는 스칼라 레지스터의 2개의 최하위 요소로부터 추출되고, 프로세싱 회로부는 제1 소스 벡터 레지스터(160(B))의 제3 부분의 2개의 최상위 요소(데이터 아이템 15 및 14)를 추출하여 스칼라 레지스터의 최하위 2개의 요소에 캐리 데이터(170)로서 저장한다. 제4 박동 프로세싱에서, 데이터의 하나 이상의 추가 비트는 스칼라 레지스터의 2개의 최하위 요소로부터 추출된다. 스칼라 레지스터에서 캐리 데이터의 위치는 임의적이며, 도 14 및 도 15가 2개의 가능성을 도시하지만, 다른 구성도 가능하다는 것을 이해할 것이다.

[0128] 도 16은 제1 소스 벡터 레지스터(180), 제2 소스 벡터 레지스터(182), 목적지 레지스터(184), 스칼라 레지스터 및 제어 정보를 명시하는 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 연산의 시퀀스를 개략적으로 도시한다. 제1 소스 벡터 레지스터(180), 제2 소스 벡터 레지스터(182) 및 목적지 레지스터(184) 각각은 복수의 박동 프로세싱에서 처리될 복수의 부분으로서 배열된다. 도 16은, 프로세싱 회로부가 주어진 틱에 대한 복수의 박동 프로세싱 중 2개의 박동을 수행할 수 있는 하드웨어가 제공된다는 점에서, 도 15 및 도 14와 상이하다. 즉, 2개의 박동이 병렬적으로 수행된다. 제1 소스 벡터 레지스터(180) 및 제2 소스 벡터 레지스터(182)의 각 부분은 4개의 8-비트 요소를 포함하고, 제어 정보는 2개의 요소에 해당하는 시프트가 수행되어야 함을 명시한다. 제1 틱에 응답하여, 프로세싱 회로부는 제1 소스 벡터 레지스터(180(C), 180(D))의 2개의 최하위 부분에 해당하는 제1 및 제2 박동 프로세싱을 수행한다. 프로세싱 회로부는 제2 소스 벡터 레지스터(182(D))의 최하위 부분으로부터 최상위 2개의 요소(데이터 아이템 3 및 2)를 하나 이상의 추가 비트로서 추출하도록 배열된다. 하나 이상의 추가 비트는 제1 소스 벡터 레지스터(180(D))의 최하위 부분의 2개의 최하위 요소(데이터 아이템 5 및 4)와 연결된다. 연결의 결과는 목적지 벡터 레지스터(184(D))의 최하위 부분에 저장된다. 제1 소스 벡터 레지스터(180(D))의 최하위 부분의 2개의 최상위 요소(데이터 아이템 7 및 6)는 제2 박동에서 사용되도록 전달된다. 제2 박동이 제1 박동과 병렬로(동일한 틱으로) 수행되므로, 제1 소스 벡터 레지스터(180(D))의 최하위 부분의 2개의 최상위 요소(데이터 아이템 7 및 6)는 스칼라 레지스터를 요구하지 않고 전달된다. 따라서, 동일한 틱에서, 제2 박동 프로세싱의 일부로서, 제1 소스 벡터 레지스터(180(D))의 최하위 부분의 2개의 최상위 요소(데이터 아이템 7 및 6)는 제1 소스 벡터 레지스터(180(C))의 제2 부분의 2개의 최하위 요소(데이터 아이템 9 및 8)와 연결되기 위해 하나 이상의 추가 비트로서 전달된다. 연결의 결과는 목적지 벡터 레지스터(184(C))의 제2 부분에 저장된다. 프로세싱 회로부는 또한, 제1 소스 벡터 레지스터(180(C))의 제2 부분으로부터 다음 틱 동안 프로세싱을 위해 전달될 스칼라 레지스터(188)의 2개의 최하위 요소에 2개의 최상위 요소(데이터 아이템 11 및 10))를 저장하도록 배열된다. 프로세싱 회로부는 또한 벡터 추출 및 병합 명령어에 응답하여 수행될 제1 및 제2 박동 프로세싱에 대한 프로세싱이 완료되었음을 나타내는 상태 정보를 설정하도록 배열된다.

[0129] 제2 박동 프로세싱 동안, 프로세싱 회로부는 상태 정보로부터, 제1 및 제2 박동 프로세싱에 대해 프로세싱이 완료되었는지를 결정할 수 있다. 따라서, 프로세싱 회로부는 제1 소스 벡터 레지스터(180(B))의 제3 부분에 해당하는 제3 박동으로부터 프로세싱을 시작한다. 프로세싱 회로부는 제1 소스 벡터 레지스터(180(B))의 제3 부분으로부터 2개의 최하위 요소(데이터 아이템 13 및 12))를 추출하고, 이들 요소를 하나 이상의 추가 비트와 연결한다. 프로세싱 회로부는 처리되고 있는 박동이 제1 박동(최하위 부분)를 포함하지 않는다고 결정할 수 있기 때문에, 하나 이상의 추가 비트가 스칼라 레지스터(188)로부터 추출된다. 특히, 하나 이상의 추가 비트는, 추출되어 제1 소스 벡터 레지스터(180(B))의 제3 부분의 2개의 최하위 요소(데이터 아이템 13 및 12)와 연결되고 연결의 결과가 목적지 레지스터(184(B))의 제3 부분에 저장되는 스칼라 레지스터(188)의 2개의 최하위 요소(데

이터 아이템 11 및 10)를 포함한다. 프로세싱 회로부는 또한 제4 박동으로 전달될 제1 소스 벡터 레지스터(180(B))의 제3 부분의 2개의 최상위 요소(데이터 아이템 15 및 14)를 추출하도록 배열된다. 프로세싱 회로부는 주어진 틱에서 2개의 박동 프로세싱을 수행할 수 있기 때문에, 박동 3 및 박동 4는 병렬로 수행되고, 전달된 데이터는 스칼라 레지스터(188)에 저장을 요구하지 않는다. 오히려, 제1 소스 벡터 레지스터(180(B))의 제3 부분의 2개의 최상위 요소(데이터 아이템 15 및 14)는 제4 박동에서 사용될 하나 이상의 추가 비트로서 전달된다. 제4 박동 동안, 제1 소스 벡터 레지스터(180(A))의 제4(최상위) 부분의 2개의 최하위 요소(데이터 아이템 17 및 16)가 추출되고, 제1 소스 벡터 레지스터의 제3 부분으로부터 전달된 하나 이상의 추가 비트와 연결된다. 연결의 결과는 목적지 벡터 레지스터(184(A))의 제4 부분(최상위 부분)에 저장된다.

[0130] 일부 대안적인 구성에서, 제4 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터(188)에 저장될 캐리 데이터로서 제1 소스 벡터 레지스터(180(A))의 제4 부분을 추출한다. 이 캐리 데이터는 잠재적으로 추가 명령어의 부분으로서 사용될 벡터 추출 및 병합 명령어의 실행에 후속하여 스칼라 레지스터(188)에 저장된 상태로 유지된다.

[0131] 도 17은 제1 소스 벡터 레지스터(240), 제2 소스 벡터 레지스터(242), 목적지 레지스터(244), 스칼라 레지스터 및 제어 정보를 명시하는 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 연산의 시퀀스가 수행되는 대안적인 구성을 개략적으로 도시한다. 도 17은 추출 및 병합 명령어가 반대로 되었다는 점에서 도 14 내지 도 16과 차이가 있다. 특히, 벡터 추출 및 병합 명령어는 소스 벡터 레지스터의 최하위 부분보다는 명시된 레지스터의 최상위 부분으로부터 수행된다. 예시된 예에서, 프로세싱 회로부는 주어진 틱에 대한 단일 박동 프로세싱을 수행하도록 배열된다. 각 부분은 4개의 요소를 포함하고, 제어 정보는 하나의 요소에 해당하는 시프트가 수행되어야 함을 명시한다. 제1 박동 프로세싱(이 경우, 명시된 레지스터의 최상위 부분에 해당)에서, 프로세싱 회로부는 제1 소스 벡터 레지스터(240(A))의 제1 부분(최상위 부분)의 최상위 3개의 요소(데이터 아이템 15에서 데이터 아이템 13까지)를 추출한다. 제1 소스 벡터 레지스터(240(A))의 제1 부분의 추출된 최상위 3개의 요소는 제2 소스 벡터 레지스터(242(A))의 제1 부분(최상위 부분)의 최하위 요소(데이터 아이템 16)와 연결되고, 연결의 결과는 목적지 벡터 레지스터(244(A))의 제1 부분으로서 저장된다. 제1 박동 프로세싱 동안, 제1 소스 벡터 레지스터(240(A))의 제1 부분(데이터 아이템 15에서 데이터 아이템 12까지)은 캐리 데이터(246)로서 추출되고 스칼라 레지스터에 저장된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(240(B))의 제2 부분의 최상위 3개의 요소(데이터 아이템 11에서 데이터 아이템 9까지)를 추출한다. 제1 소스 벡터 레지스터(240(B))의 제2 부분의 추출된 최상위 3개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(246)의 최하위 요소(데이터 아이템 12)와 연결되고, 연결의 결과는 목적지 벡터 레지스터(244(B))의 제2 부분에 저장된다. 제2 박동 프로세싱 동안 프로세싱 회로부는 또한 제1 소스 벡터 레지스터(240(B))의 제2 부분(데이터 아이템 11에서 데이터 아이템 8까지)을 스칼라 레지스터에 저장될 캐리 데이터(248)로서 전달한다. 제3 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(240(C))의 제3 부분의 최상위 3개의 요소(데이터 아이템 7에서 데이터 아이템 5까지)를 추출한다. 제1 소스 벡터 레지스터(240(C))의 제3 부분의 추출된 최상위 3개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(248)의 최하위 요소(데이터 아이템 8)와 연결되고, 연결의 결과는 목적지 벡터 레지스터(244(C))의 제3 부분에 저장된다. 제3 박동 프로세싱 동안 프로세싱 회로부는 또한 제1 소스 벡터 레지스터(240(C))의 제3 부분(데이터 아이템 7에서 데이터 아이템 4까지)을 스칼라 레지스터에 저장될 캐리 데이터(250)로서 추출한다. 제4 박동 프로세싱 동안 프로세싱 회로부는 제1 소스 벡터 레지스터(210(D))의 제4 부분(최하위 부분)의 최상위 3개의 요소(데이터 아이템 3에서 데이터 아이템 1까지)를 추출한다. 제1 소스 벡터 레지스터(210(D))의 제4 부분의 추출된 최상위 3개의 요소는 스칼라 레지스터에 저장된 캐리 데이터(250)의 최하위 요소(데이터 아이템 4)와 연결되고, 연결의 결과는 목적지 벡터 레지스터(244(D))의 제4 부분(최하위)에 저장된다. 일부 대안적인 구성에서, 제4 박동 프로세싱 동안 프로세싱 회로부는 또한 스칼라 레지스터에 저장될 캐리 데이터로서 제1 소스 벡터 레지스터(240(D))의 제4 부분을 추출한다. 이 캐리 데이터는 벡터 추출 및 병합 명령어의 실행에 후속하여 스칼라 레지스터에 저장된 상태로 유지된다. 이전 도면에 도시된 바와 같이, 스칼라 레지스터 내에서 전달될 데이터 요소의 위치 및 스칼라 레지스터에서 미사용 요소의 값은 임의적이다. 다른 구성의 다양한 조합이, 예를 들어, 스칼라 레지스터의 최상위 요소를 전달하기 위한 요소를 저장하고, 미사용 요소를 0으로 설정하는 것이 가능하다.

[0132] 도 18은 벡터 추출 및 병합 명령어에 응답하여 프로세싱 회로부에 의해 수행되는 단계의 시퀀스를 개략적으로 도시한다. 흐름은 단계(S170)에서 시작하며, 여기서 제1 소스 벡터 레지스터, 제2 소스 벡터 레지스터, 목적지 벡터 레지스터 및 제어 파라미터를 명시하는 벡터 추출 및 병합 명령어가 디코더 회로부에 의해 수신되었는지 여부가 결정된다. 그렇지 않은 경우 흐름은 단계(S170)에서 유지된다. 단계(S170)에서, 디코더 회로부가 벡터 추출 및 병합 명령어를 수신하였다고 결정되면, 디코더 회로부는 벡터 추출 및 병합 명령어에 기초하여 제어 신

호를 생성한다. 그런 다음, 흐름은 제어 신호에 기초하여, 값 K가 상태 정보에 기초하여 설정되는 단계(S172)로 진행한다. 상태 정보가 박동 프로세싱이 수행되지 않았음을 나타내는 경우, K는 제1 박동 프로세싱을 나타내도록 설정된다. 반면에, 상태 정보가 복수의 박동 중 제1 하나 이상의 박동이 완료되었음을 나타내면, K는 복수의 박동 중 제1 완료되지 않은 박동을 나타내도록 설정된다. 그런 다음, 흐름은 단계(S174)로 진행하고, 여기서 프로세싱 회로부는 제어 파라미터에 의해 명시된 비트를 제1 소스 벡터 레지스터의 K번째 부분으로부터 추출한다. 그런 다음, 흐름은 단계(S176)로 진행하고, 여기서 K는 부분이 제1 부분임을 표시하는지 여부를 결정한다. 그렇다면 흐름은 단계(S178)로 진행하고, 여기서 프로세싱 회로부는 제2 소스 벡터 레지스터의 제1 부분으로부터 (제어 파라미터에 의해 표시된 바와 같이) 하나 이상의 추가 비트를 추출한다. 그런 다음, 흐름은 단계(S182)로 진행된다. 반면에, 단계 S176에서, K가 K번째 부분이 제1 부분이 아님을 나타낸다고 결정된 경우, 흐름은 단계 S180으로 진행하고, 여기서 하나 이상의 추가 비트는 제1 소스 벡터 레지스터의 (K-1) 부분으로부터 전달된 하나 이상의 추가 비트로서 획득된다. 캐리는, 예를 들어, 프로세싱 회로부가 틱당 1개의 박동 이상을 수행하기에 충분한 하드웨어를 제공받으면, 프로세싱 회로부 내의 내부 캐리일 수 있다. 대안적으로, 캐리 데이터는 하나 이상의 추가 비트가 벡터 추출 및 병합 명령어의 선행 박동의 부분으로서 저장된 스칼라 레지스터로부터 추출될 수 있다. 그런 다음, 흐름은 단계(S182)로 진행된다. 단계(S182)에서, 하나 이상의 추출된 비트는 하나 이상의 추가 비트와 연결된다. 그런 다음, 흐름은 단계(S184)로 진행하고, 여기서 연결의 결과는 대상 레지스터의 K번째 부분에 저장된다. 그런 다음, 흐름은 단계(S186)로 진행하고, 여기서 K번째 부분이 제1 소스 벡터 레지스터의 마지막 부분인지 여부를 결정한다. 예이면, 흐름은 단계(S170)로 복귀한다. 단계(S186)에서, K번째 부분이 마지막 부분이 아니라고 결정된 경우, 이어서 흐름은 단계(S188)로 진행하고, 여기서 목적지 레지스터에 저장되지 않은 제1 소스 벡터 레지스터의 K번째 부분의 적어도 하나의 비트가 (K+1)번째 박동에서 처리되도록 전달된다. 캐리는, 예를 들어, 프로세싱 회로부가 틱당 다수(복수)의 박동 프로세싱을 수행하기에 충분한 하드웨어를 제공받으면, 프로세싱 회로부 내의 내부 캐리일 수 있다. 대안적으로, 캐리는 벡터 추출 및 병합 명령어에 명시된 스칼라 레지스터에 제1 소스 벡터 레지스터의 K번째 부분의 적어도 하나의 비트를 저장함으로써 수행될 수 있다. 그런 다음, 흐름은 단계(S190)로 진행하여 K가 증가하고 흐름은 단계(S174)로 복귀한다.

[0133] 도 18의 단계의 시퀀스가 K를 순차적으로 증가시킴으로써 설명되었지만, 여기서 틱당 복수의 박동 프로세싱을 수행하기에 충분한 하드웨어가 제공되면, 동일한 틱 내에서 수행되고 있는 각각의 박동(K의 각각의 값)에 해당하는 단계는 병렬로 수행된다. 예를 들어, 박동 K 및 K+1이 병렬로 수행되고 있다면, 단계(S174)는 제1 소스 벡터 레지스터의 K 번째 부분에서 제어 파라미터에 의해 명시된 비트를 추출하는 것과 병렬로 제1 소스 벡터 레지스터의 (K+1)번째 부분에 대한 제어 파라미터에 의해 명시된 비트를 추출하는 것을 포함할 수 있다. 그런 다음 K 및 (K+1)번째 박동 각각에 대한 하나 이상의 추가 비트가 병렬로 추출된다. 잠재적으로, K가 K번째 부분이 제1 부분임을 나타내는 경우, K번째 부분에 대한 하나 이상의 추가 비트는 제1 소스 벡터 레지스터의 K번째 부분으로부터 추출되는 (K+1)번째 부분에 대한 하나 이상의 추가 비트와 병렬로 제2 소스 벡터 레지스터로부터 추출될 수 있다. 연결 단계(S182)는 K번째 및 (K+1)번째 부분에 대해 병렬로 수행될 것이고, K번째 및 (K+1)번째 부분에 대한 저장 단계(S184)는 병렬로 수행될 수 있다. 단계(S186)에서, K가 마지막 부분에 해당하는지 여부를 결정하는 것은 처리되고 있는 K의 가장 높은 부분(최상위 부분)에 기초하여 이루어지며, 흐름이 계속된다면, 이 결정에 기초하여, 단계(S188)에서 후속 틱 프로세싱을 위해 (K+1)번째 부분으로부터 캐리를 추출할 것이다. 하드웨어 제공의 세부 사항에 의존하여, 임의의 수의 박동 프로세싱이 병렬로 수행될 수 있다는 것이 당업자에 의해 인식될 것이다.

[0134] 도 19는 본 기술의 다양한 구성에 따른 데이터 프로세싱 장치의 제조를 위한 컴퓨터 판독가능 코드를 포함하는 비일시적 컴퓨터 판독가능 매체를 개략적으로 도시한다. 비일시적 컴퓨터 판독가능 매체(1000) 상에 저장된 컴퓨터 판독가능 코드(1002)에 기초하여 제조가 수행된다. 컴퓨터 판독가능 코드는 개념을 구현하는 장치를 포함하는 집적 회로를 제조하기 위해 전자 설계 자동화(EDA) 스테이지를 포함하는, 반도체 설계 및 제조 프로세스의 하나 이상의 스테이지에서 사용될 수 있다. 제조 프로세스는 위에 설명된 구성을 구현하거나 하나 이상의 집적 회로로 구현된 장치의 제조를 용이하게 하거나 그렇지 않으면 위에 설명된 구성을 구현하도록 FPGA를 구성하기 위해 필드 프로그래밍가능 게이트 어레이(FPGA)와 같은 하나 이상의 프로그래밍가능 하드웨어 유닛에 직접 컴퓨터 판독가능 코드(1002)의 적용을 수반한다. 예로서, 제조된 설계(1004)는 도 7과 관련하여 설명된 바와 같이 레지스터(32), 디코더 회로부(38) 및 프로세싱 회로부(40)를 갖는 장치(30)를 포함한다. 그러나, 제조된 설계는 도 9 내지 도 18과 관련하여 설명된 바와 같이 벡터 추출 및 병합 명령어를 구현할 수 있는 도 1, 도 7, 및 도 8에 제시된 회로 중 임의의 회로에 해당할 수 있다.

[0135] 도 20은 사용될 수도 있는 시뮬레이터 구현을 도시한다. 앞서 설명한 예는 관련 기술을 지원하는 특정 프로세

싱 하드웨어를 동작시키기 위한 장치 및 방법의 측면에서 본 발명을 구현하지만, 본원에 설명된 예에 따라 컴퓨터 프로그램의 사용을 통해 구현되는 명령어 실행 환경을 제공하는 것이 또한 가능하다. 이러한 컴퓨터 프로그램은, 하드웨어 아키텍처의 소프트웨어 기반 구현을 제공하는 한, 시뮬레이터로서 종종 지칭된다. 다양한 시뮬레이터 컴퓨터 프로그램은 에뮬레이터, 가상 머신, 모델, 및 동적 바이너리 변환기를 비롯한, 바이너리 변환기를 포함한다. 전형적으로, 시뮬레이터 구현예는 호스트 운영 체제(510)를 선택적으로 실행하고, 시뮬레이터 프로그램(505)을 지원하는 호스트 프로세서(515)상에서 실행될 수 있다. 일부 배열에서, 하드웨어와 제공된 명령어 실행 환경 사이에 다수의 시뮬레이션 계층이 있을 수 있고/있거나, 동일한 호스트 프로세서 상에 제공된 다수의 별개의 명령어 실행 환경이 있을 수 있다. 역사적으로, 합리적인 속도로 실행되는 시뮬레이터 구현을 제공하기 위해 강력한 프로세서가 필요하였지만, 그러한 접근법은 특정 상황에서, 예컨대 호환성 또는 재사용 이유 때문에 다른 프로세서에 기본적인(native) 코드를 실행하려는 요구가 있을 때, 정당화될 수 있다. 예를 들어, 시뮬레이터 구현은 호스트 프로세서 하드웨어에 의해 지원되지 않는 추가적인 기능을 갖는 명령어 실행 환경을 제공하거나, 상이한 하드웨어 아키텍처와 전형적으로 관련되는 명령어 실행 환경을 제공할 수 있다. 시뮬레이션의 개요는 문헌["Some Efficient Architecture Simulation Techniques", Robert Bedichek, Winter 1990, USENIX Conference, Pages 53 to 63]에 제공되어 있다.

[0136] 예가 특정 하드웨어 구조 또는 특징을 참조하여 전술된 경우, 시뮬레이션된 구현에서 동등한 기능이 적합한 소프트웨어 구조 또는 특징에 의해 제공될 수 있다. 예를 들어, 특정 회로부는 시뮬레이션된 구현에 컴퓨터 프로그램 로직으로 제공될 수 있다. 마찬가지로, 레지스터 또는 캐시와 같은 메모리 하드웨어가 시뮬레이션된 구현에서 소프트웨어 데이터 구조로 제공될 수 있다. 앞서 설명한 예에서 언급된 하드웨어 요소 중 하나 이상이 호스트 하드웨어 상에 존재하는 배열에서, 일부 시뮬레이션된 구현은 적합한 경우 호스트 하드웨어를 사용할 수 있다.

[0137] 시뮬레이터 프로그램(505)은 컴퓨터 판독가능 저장 매체(이는 비-일시적 매체일 수 있음) 상에 저장될 수 있으며, 타겟 코드(500)(이는 애플리케이션, 운영 체제, 및 하이퍼바이저를 포함할 수 있음)에 가상 하드웨어 인터페이스(명령어 실행 환경)를 제공하고, 이는 시뮬레이터 프로그램(505)에 의해 모델링되는 하드웨어 아키텍처의 하드웨어 인터페이스와 동일하다. 따라서, 타겟 코드(500)의 프로그램 명령어는 시뮬레이터 프로그램(505)을 사용하는 명령어 실행 환경 내로부터 실행될 수 있고, 따라서 위에서 논의한 장치(30)의 하드웨어 특징을 실제로 갖지 않는 호스트 컴퓨터(515)가 그러한 특징을 에뮬레이팅할 수 있다. 시뮬레이터 프로그램은 레지스터(32)의 거동을 에뮬레이트하기 위한 레지스터 로직(532), 디코더 회로부(38)의 거동을 에뮬레이트하기 위한 디코더 회로부 로직(538), 및 프로세싱 회로부(40)의 거동을 에뮬레이트하기 위한 프로세싱 로직(540)을 포함할 수 있다. 또한, 시뮬레이터 프로그램은 도 9 내지 도 18과 관련하여 설명된 바와 같이 벡터 추출 및 병합 명령어를 구현할 수 있는 도 1, 도 7, 및 도 8에 제시된 회로 중 임의의 회로를 구현하기 위한 로직을 포함할 수 있다. 따라서, 본원에서 설명된 기술은, 도 20의 예에서 시뮬레이터 프로그램(505)에 의해 소프트웨어에서 수행될 수 있다.

[0138] 간략하게 전체적으로 요약하면, 장치, 방법 및 매체가 제공된다. 장치는 제어 파라미터, 제1 벡터 레지스터, 제2 벡터 레지스터, 및 목적지 벡터 레지스터를 명시하는 벡터 추출 및 병합 명령어에 응답하여 제어 신호를 생성하는 디코더 회로부를 포함한다. 장치는 제어 신호에 응답하여, 복수의 박동 프로세싱을 수행하기 위한 프로세싱 회로부를 포함하고, 각각의 박동은 적어도 제1 벡터 레지스터 및 목적지 벡터 레지스터의 부분에 해당하는 프로세싱을 포함한다. K번째 박동에 대한 프로세싱은, 제어 파라미터에 의해 명시된 비트를 제1 벡터 레지스터의 K번째 부분으로부터 추출하는 것, 비트를 추가 비트와 연결하는 것, 및 결과를 목적지 레지스터의 K번째 부분에 저장하는 것을 포함한다. 추가 비트는, 제1 부분의 경우, 제2 벡터 레지스터의 제1 부분으로부터 추출되고, 그렇지 않으면, 제1 벡터 레지스터의 (K-1)번째 부분으로부터 추출된다.

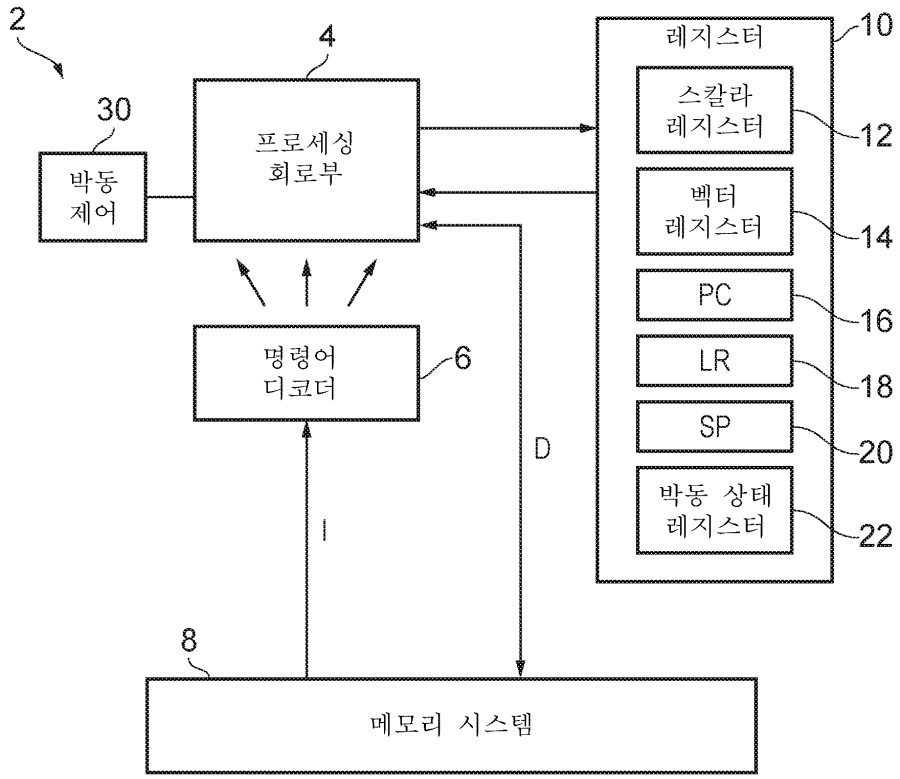
[0139] 본 출원에서, "...하도록 구성된"이라는 말은 장치의 요소가 정의된 동작을 수행할 수 있는 구성을 갖는다는 것을 의미하는 데 사용된다. 이러한 맥락에서, "구성"은 하드웨어 또는 소프트웨어의 상호연결(interconnection)의 배열 또는 방식을 의미한다. 예를 들어, 장치는 정의된 동작을 제공하는 전용 하드웨어를 가질 수 있거나, 프로세서 또는 다른 프로세싱 디바이스는 기능을 수행하도록 프로그래밍될 수 있다. "하도록 구성된"은, 장치 요소가, 정의된 동작을 제공하기 위해 임의의 방식으로 변경될 필요가 있음을 암시하지는 않는다.

[0140] 예시적인 구성이 첨부 도면을 참조하여 본원에 상세히 설명되었지만, 본 발명은 이들 정확한 구성에 제한되지 않고, 다양한 변경, 추가 및 수정이 첨부된 청구범위에 의해 정의된 바와 같이 본 발명의 범위 및 사상으로부터 벗어나지 않고 당업자에 의해 이루어질 수 있다는 점이 이해되어야 한다. 예를 들어, 본 발명의 범주로부터 벗

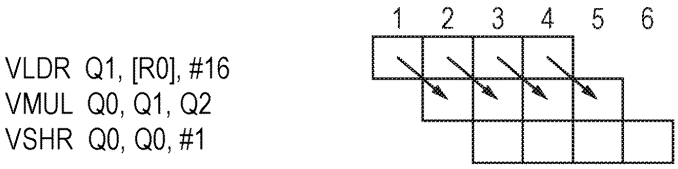
어남이 없이 독립항의 특징과 종속항의 특징의 다양한 조합이 이루어질 수 있다.

도면

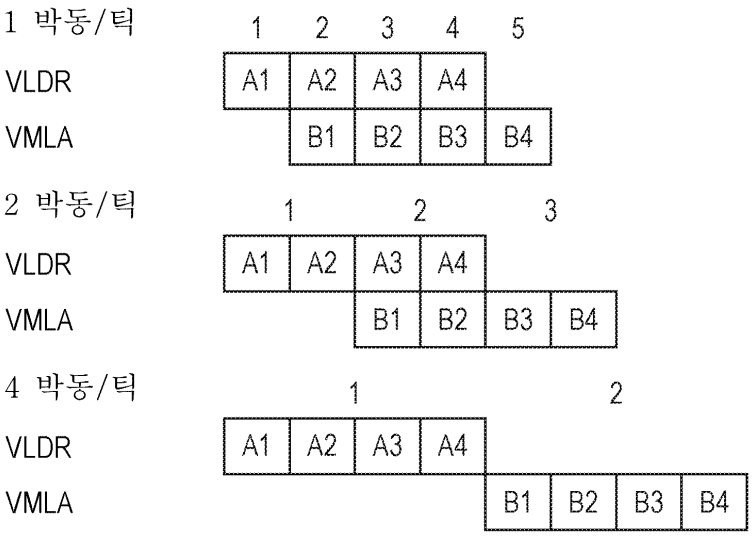
도면1



도면2



도면3



도면4

22

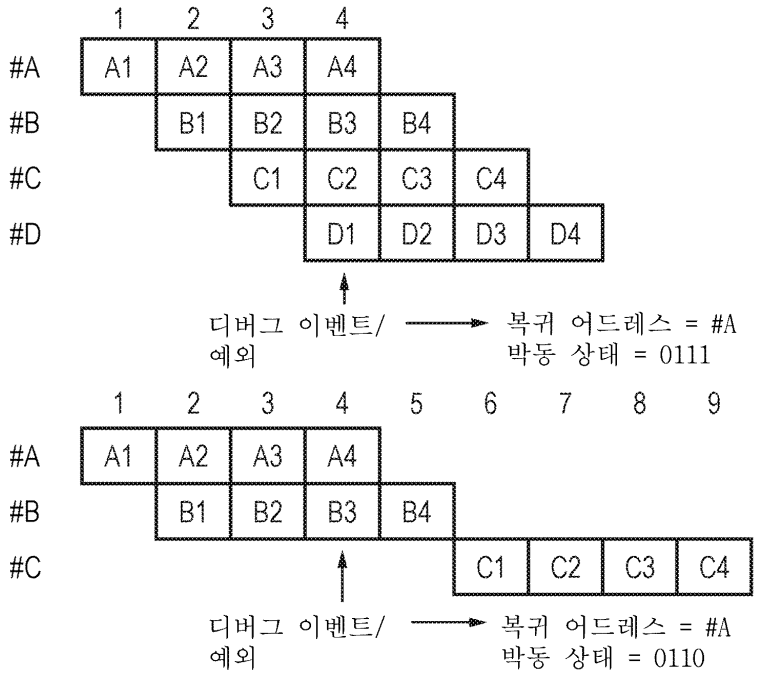
박동 상태 정보	완료된 박동
0000	비활성
0001	A1
0010	A1 A2
0011	A1 A2 B1
0100	A1 A2 A3
0101	A1 A2 A3 B1
0110	A1 A2 A3 B1 B2
0111	A1 A2 A3 B1 B2 C1
1XXX	예약됨

Ax = 가장 오래된 미완료된 명령어의 X번째 박동

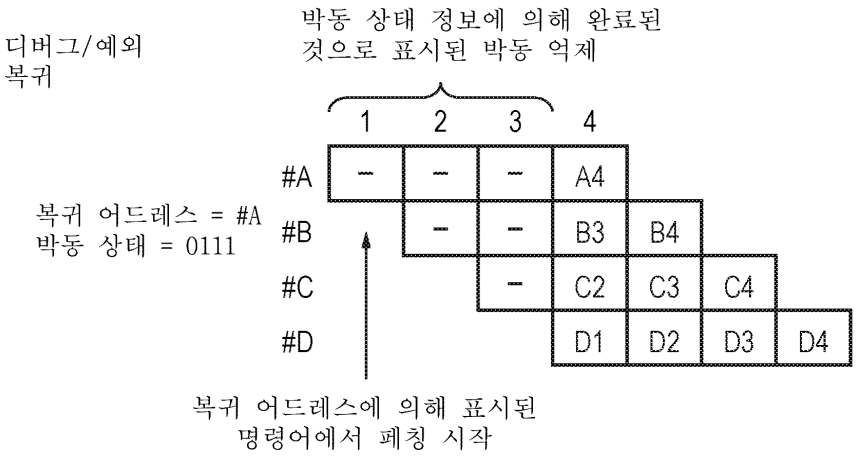
Bx = A 이후 다음 벡터 명령어의 X번째 박동

Cx = B 이후 다음 벡터 명령어의 X번째 박동

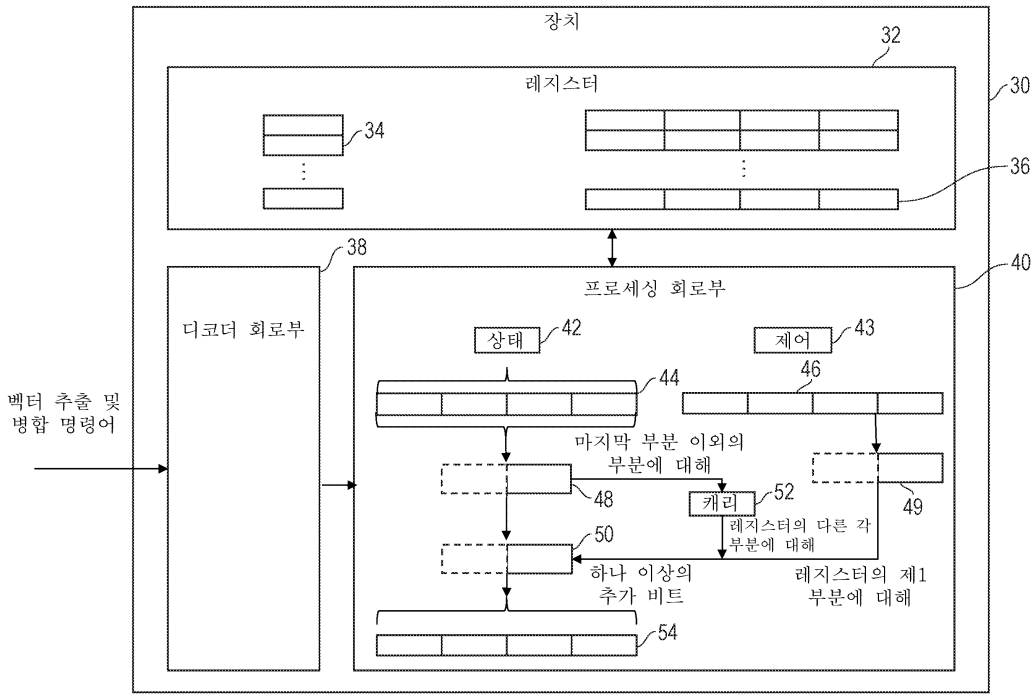
도면5



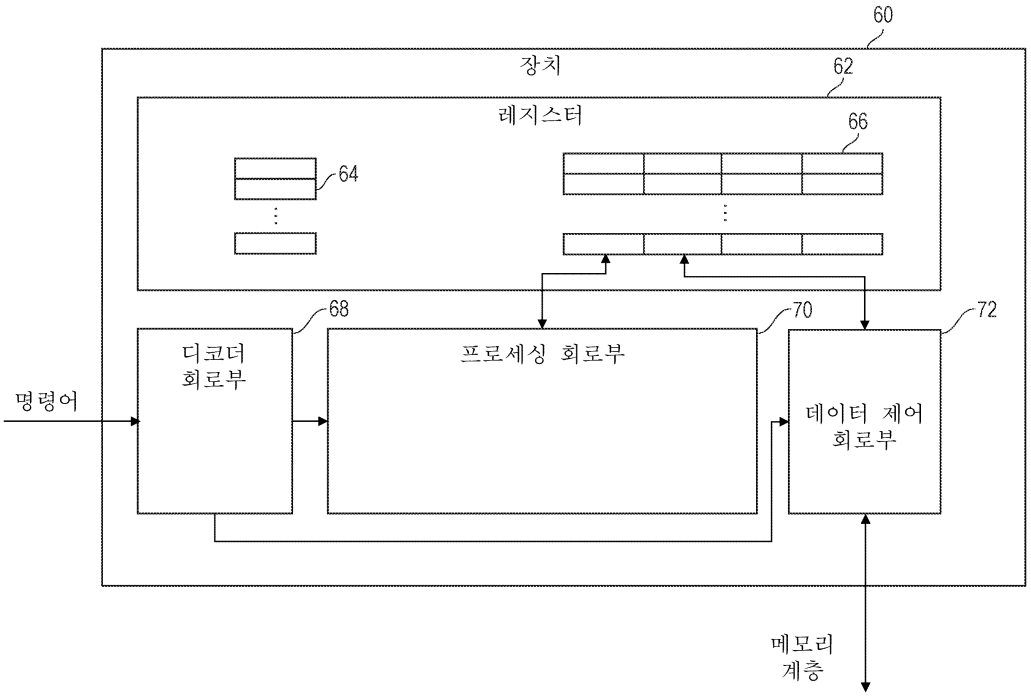
도면6


디버그/예외
복귀
박동 상태 정보에 의해 완료된
것으로 표시된 박동 억제
1
2
3
4
#A
#B
#C
#D
A4
B3
B4
C2
C3
C4
D1
D2
D3
D4
복귀 어드레스 = #A
박동 상태 = 0111
복귀 어드레스에 의해 표시된
명령어에서 페칭 시작

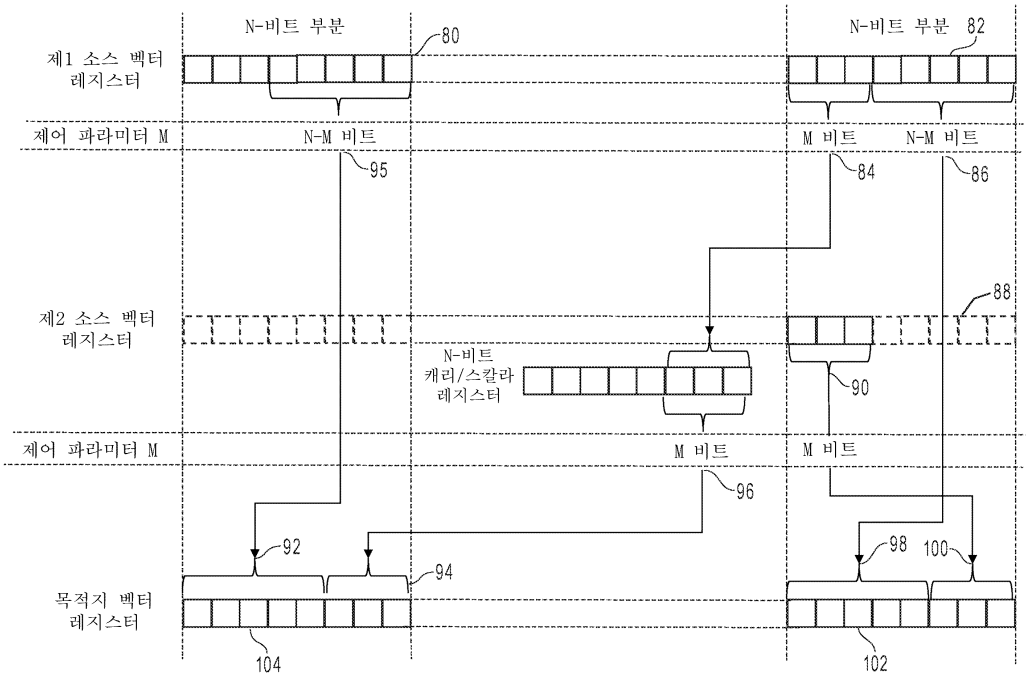
도면7


장치
32
레지스터
30
34
36
38
40
디코더 회로부
프로세싱 회로부
상태
42
제어
43
44
46
벡터 추출 및
병합 명령어
마지막 부분 이외의
부분에 대해
48
캐리
52
49
레지스터의 다른 각
부분에 대해
50
하나 이상의
추가 비트
레지스터의 제1
부분에 대해
54

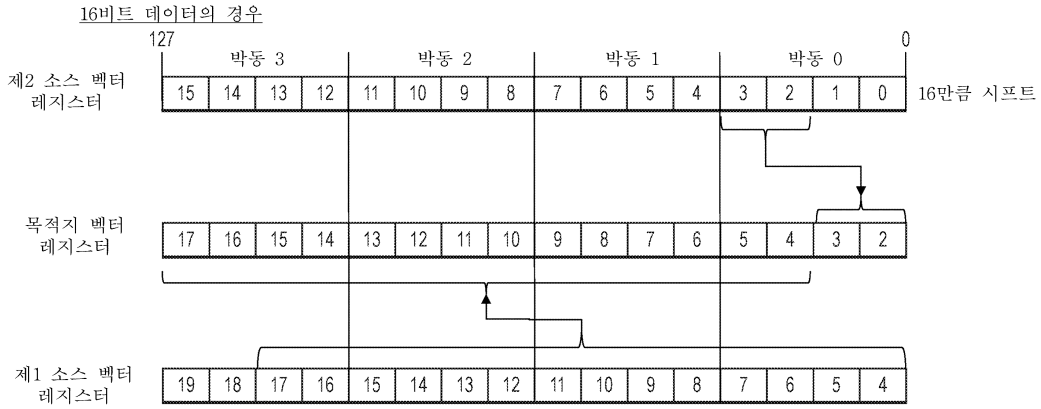
도면8


60
장치
62
레지스터
66
64
68
70
72
디코더
회로부
프로세싱 회로부
데이터 제어
회로부
명령어
메모리
계층

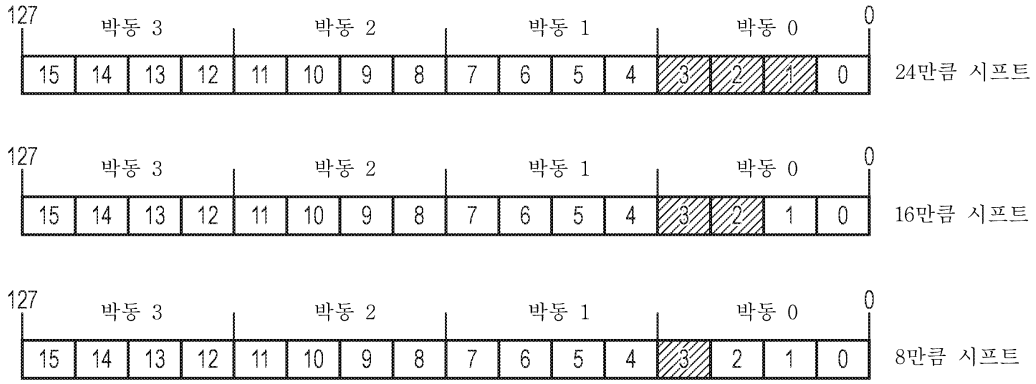
도면9


N-비트 부분
80
N-비트 부분 82
제1 소스 벡터
레지스터
제어 파라미터 M
N-M 비트
M 비트
N-M 비트
95
84
86
88
제2 소스 벡터
레지스터
N-비트
캐리/스칼라
레지스터
90
제어 파라미터 M
M 비트
M 비트
96
92
94
98
100
목적지 벡터
레지스터
104
102

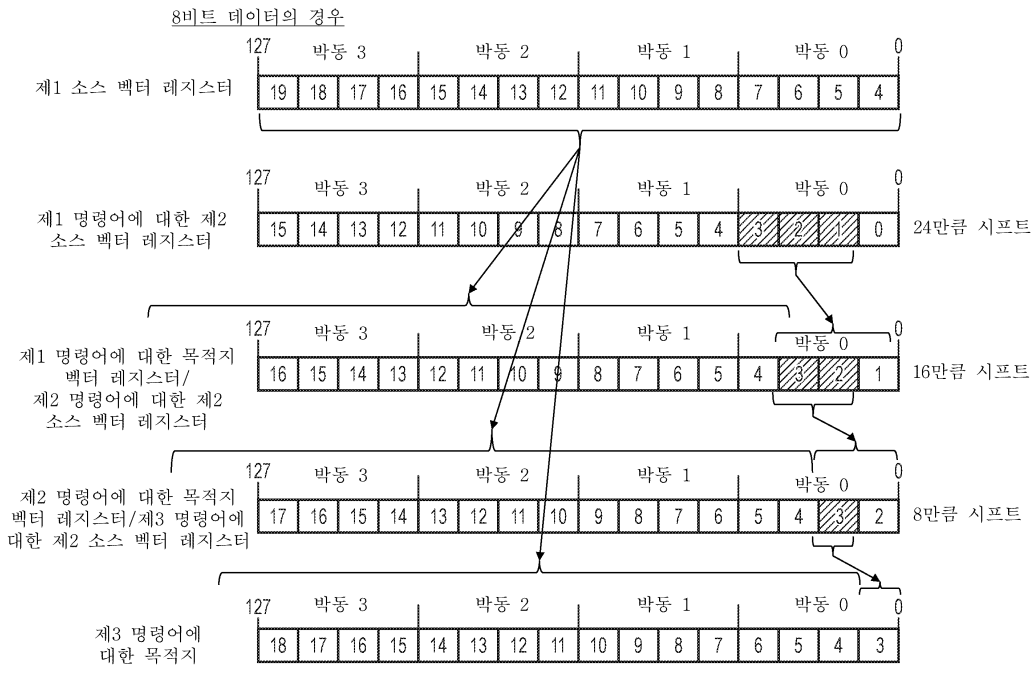
도면10



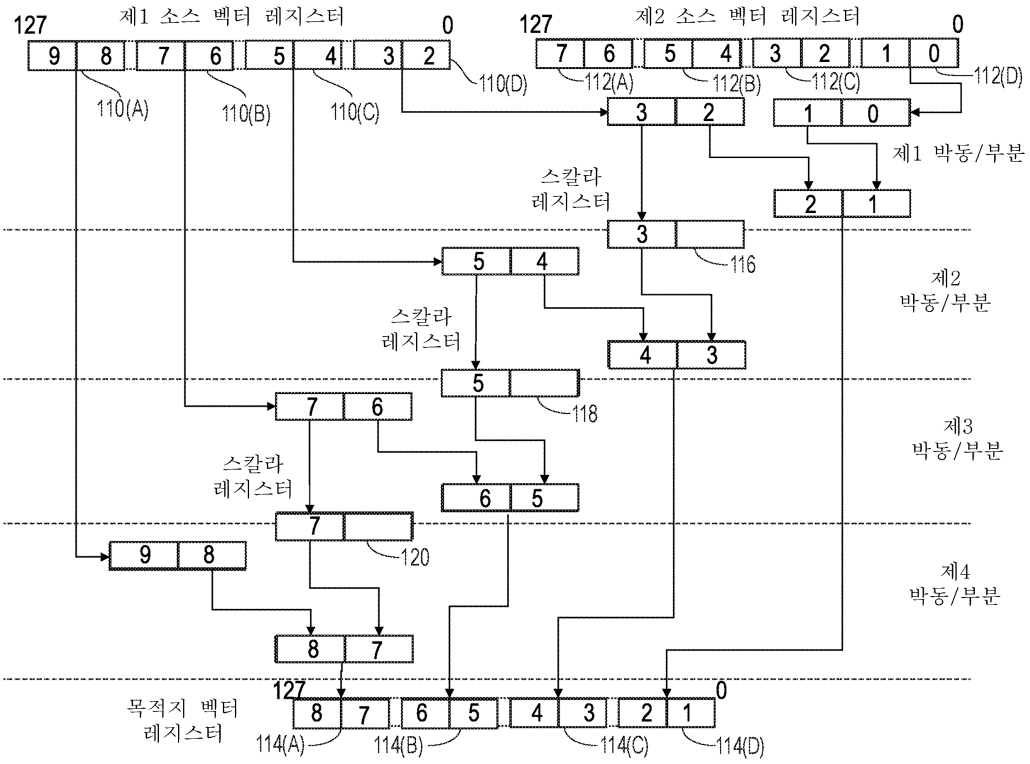
도면11



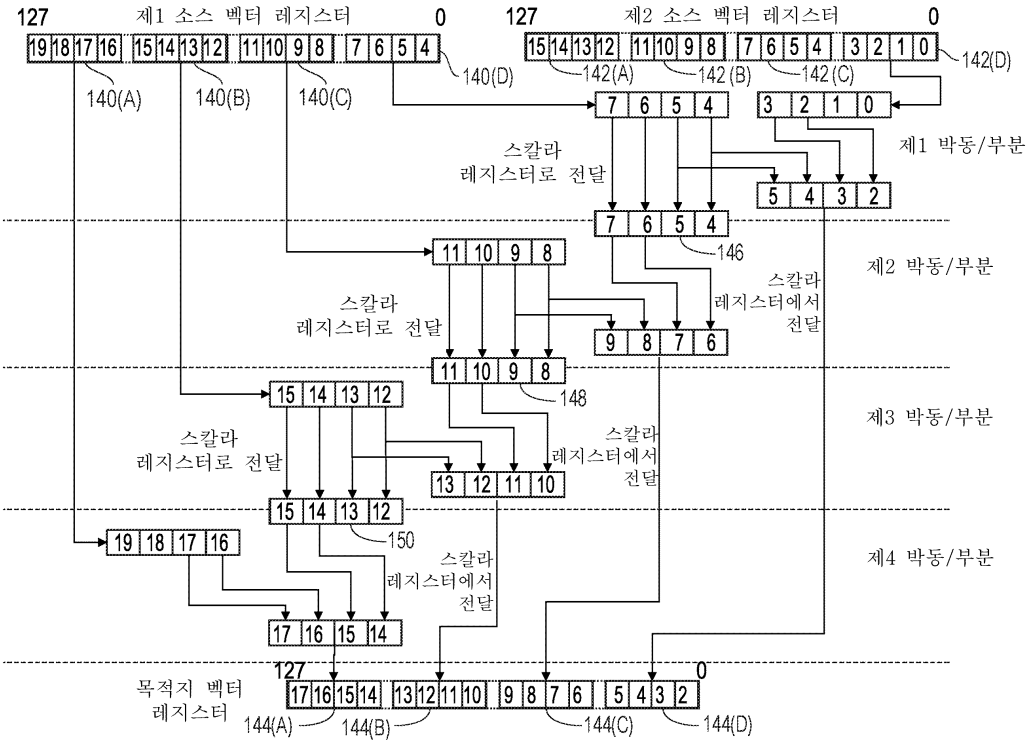
도면12



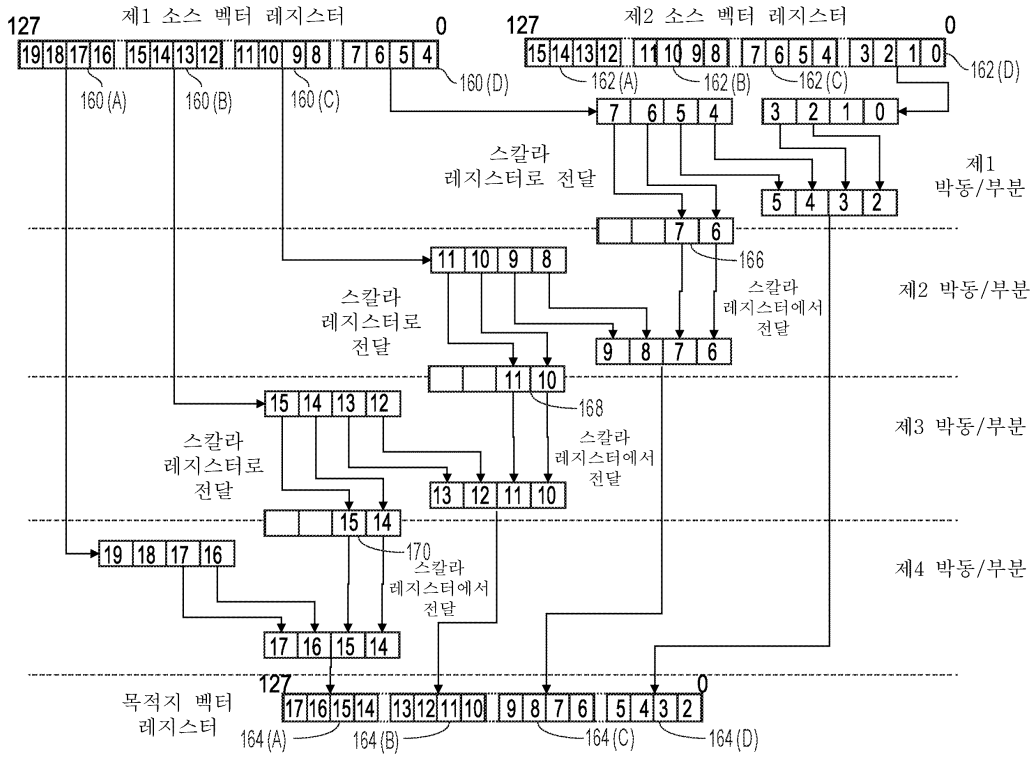
도면13


127
제1 소스 벡터 레지스터
0
9 8 7 6 5 4 3 2
110(A)
110(B)
110(C)
110(D)
127
제2 소스 벡터 레지스터
0
7 6 5 4 3 2 1 0
112(A)
112(B)
112(C)
112(D)
3 2
1 0
제1 박동/부분
스칼라 레지스터
2 1
3
116
5 4
제2 박동/부분
스칼라 레지스터
4 3
5
118
7 6
제3 박동/부분
스칼라 레지스터
6 5
7
120
9 8
제4 박동/부분
8 7
127
목적지 벡터 레지스터
0
8 7 6 5 4 3 2 1
114(A)
114(B)
114(C)
114(D)

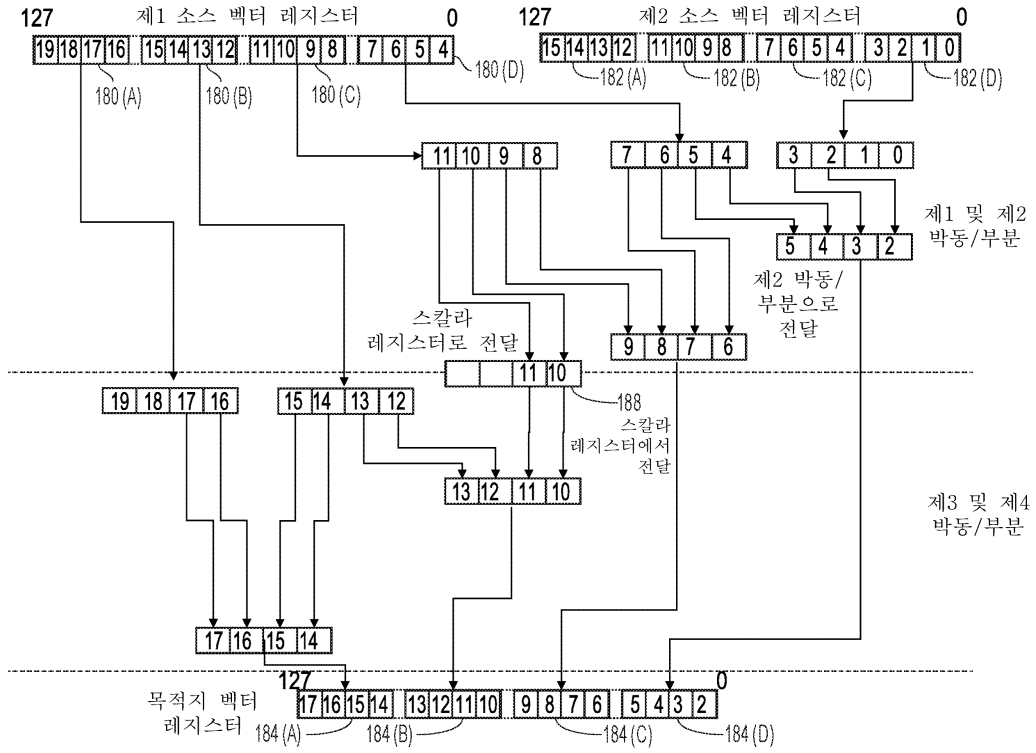
도면14


127
제1 소스 벡터 레지스터
0
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4
140(A)
140(B)
140(C)
140(D)
127
제2 소스 벡터 레지스터
0
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
142(A)
142(B)
142(C)
142(D)
7 6 5 4
3 2 1 0
제1 박동/부분
스칼라 레지스터로 전달
5 4 3 2
7 6 5 4
146
제2 박동/부분
스칼라 레지스터로 전달
스칼라 레지스터에서 전달
11 10 9 8
9 8 7 6
11 10 9 8
148
제3 박동/부분
스칼라 레지스터로 전달
스칼라 레지스터에서 전달
15 14 13 12
13 12 11 10
15 14 13 12
150
제4 박동/부분
스칼라 레지스터에서 전달
19 18 17 16
17 16 15 14
127
목적지 벡터 레지스터
0
17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2
144(A)
144(B)
144(C)
144(D)

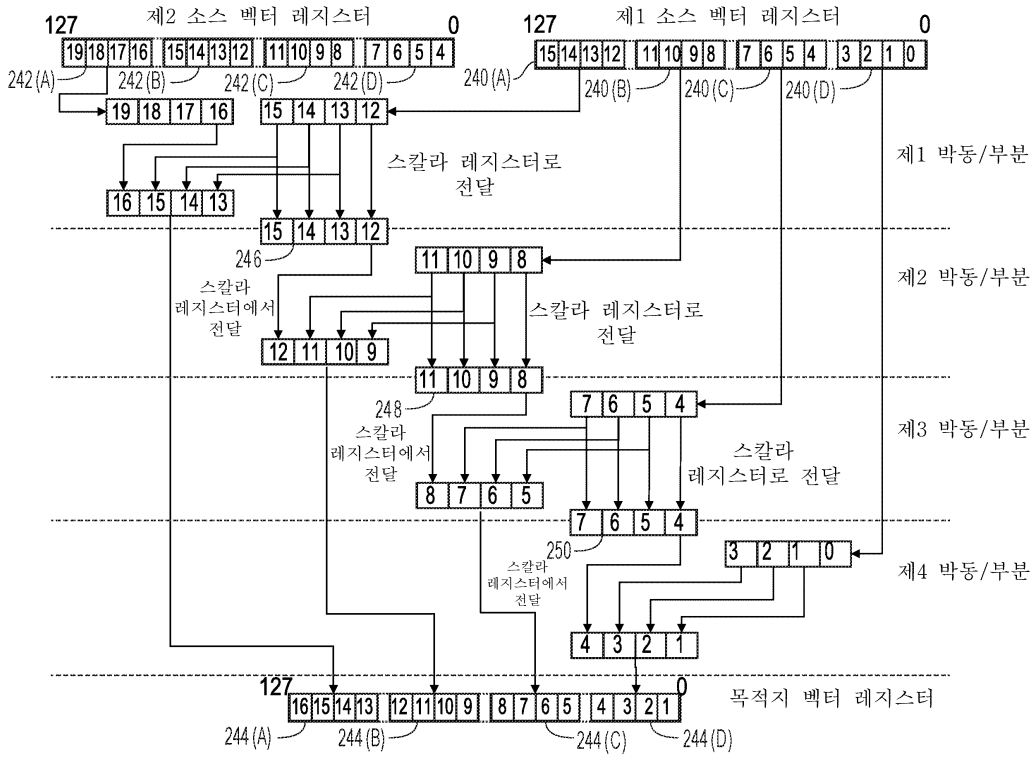
도면15


제1 소스 벡터 레지스터
제2 소스 벡터 레지스터
127
0
160(A)
160(B)
160(C)
160(D)
162(A)
162(B)
162(C)
162(D)
스칼라 레지스터로 전달
제1 박동/부분
제2 박동/부분
제3 박동/부분
제4 박동/부분
166
168
170
스칼라 레지스터에서 전달
목적지 벡터 레지스터
164(A)
164(B)
164(C)
164(D)

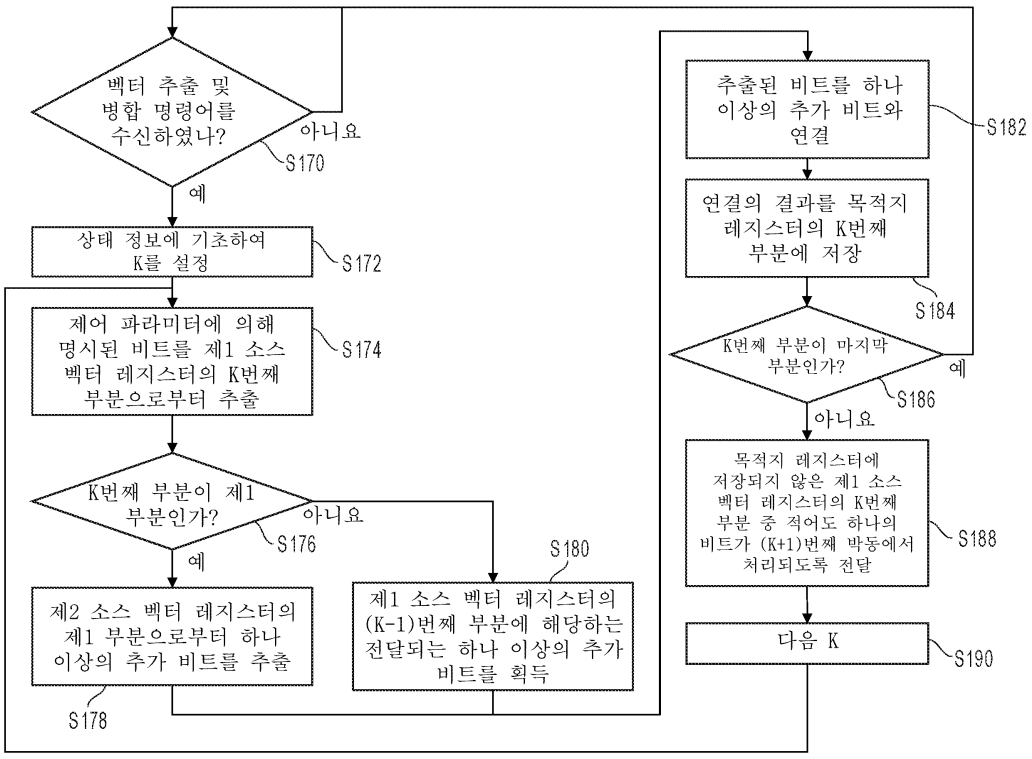
도면16


제1 소스 벡터 레지스터
제2 소스 벡터 레지스터
127
0
180(A)
180(B)
180(C)
180(D)
182(A)
182(B)
182(C)
182(D)
제1 및 제2 박동/부분
제2 박동/부분으로 전달
스칼라 레지스터로 전달
188
스칼라 레지스터에서 전달
제3 및 제4 박동/부분
목적지 벡터 레지스터
184(A)
184(B)
184(C)
184(D)

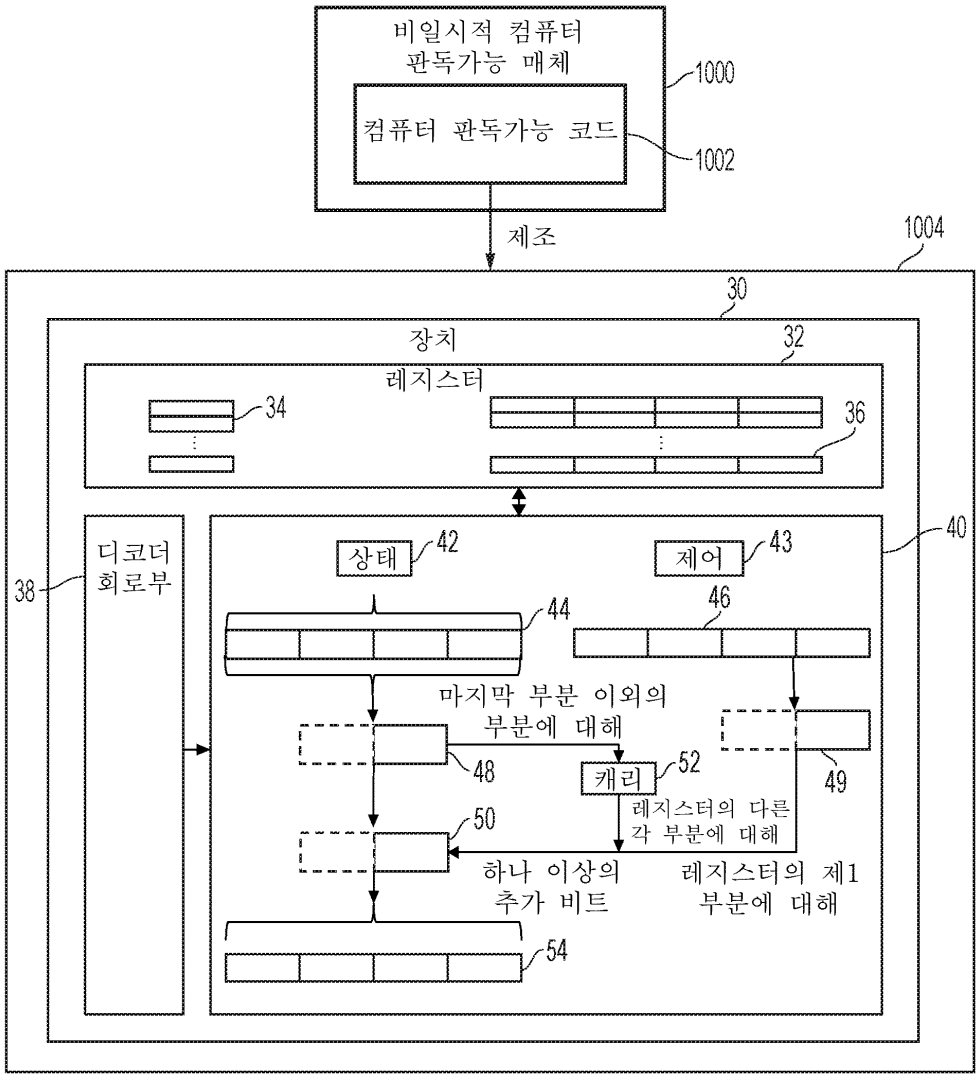
도면17


제2 소스 벡터 레지스터
제1 소스 벡터 레지스터
242(A)
242(B)
242(C)
242(D)
240(A)
240(B)
240(C)
240(D)
스칼라 레지스터로 전달
제1 박동/부분
제2 박동/부분
제3 박동/부분
제4 박동/부분
246
248
250
스칼라 레지스터에서 전달
목적지 벡터 레지스터
244(A)
244(B)
244(C)
244(D)

도면18


벡터 추출 및 병합 명령어를 수신하였나?
아니요
S170
예
상태 정보에 기초하여 K를 설정
S172
제어 파라미터에 의해 명시된 비트를 제1 소스 벡터 레지스터의 K번째 부분으로부터 추출
S174
K번째 부분이 제1 부분인가?
아니요
S176
예
제2 소스 벡터 레지스터의 제1 부분으로부터 하나 이상의 추가 비트를 추출
S178
S180
제1 소스 벡터 레지스터의 (K-1)번째 부분에 해당하는 전달되는 하나 이상의 추가 비트를 획득
추출된 비트를 하나 이상의 추가 비트와 연결
S182
연결의 결과를 목적지 레지스터의 K번째 부분에 저장
S184
K번째 부분이 마지막 부분인가?
예
S186
아니요
목적지 레지스터에 저장되지 않은 제1 소스 벡터 레지스터의 K번째 부분 중 적어도 하나의 비트가 (K+1)번째 박동에서 처리되도록 전달
S188
다음 K
S190

도면19


비일시적 컴퓨터 판독가능 매체
1000
컴퓨터 판독가능 코드
1002
제조
1004
30
장치
32
레지스터
34
36
40
디코더 회로부
38
상태
42
제어
43
44
46
마지막 부분 이외의 부분에 대해
48
캐리
52
49
50
레지스터의 다른 각 부분에 대해
하나 이상의 추가 비트
레지스터의 제1 부분에 대해
54

도면20

시뮬레이터 구현

