(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0052432
(43) 공개일자 2017년05월12일

(51) 국제특허분류(Int. Cl.)
G06F 1/32 (2006.01) *G06F 17/16* (2006.01)
G06F 9/30 (2017.01)
(52) CPC특허분류
G06F 1/3234 (2013.01)
G06F 1/3243 (2013.01)
(21) 출원번호 10-2016-0017819
(22) 출원일자 2016년02월16일
심사청구일자 2016년02월16일
(30) 우선권주장
1020150152022 2015년10월30일 대한민국(KR)

(71) 출원인
세종대학교산학협력단
서울특별시 광진구 능동로 209 (군자동, 세종대학교)
(72) 발명자
박기호
서울특별시 노원구 중계로 184, 101동 903호 (중계동, 청구아파트)
기민관
서울특별시 동대문구 서울시립대로 14 104동 1104호 (답십리동,청계한신휴플러스아파트)
(74) 대리인
양성보

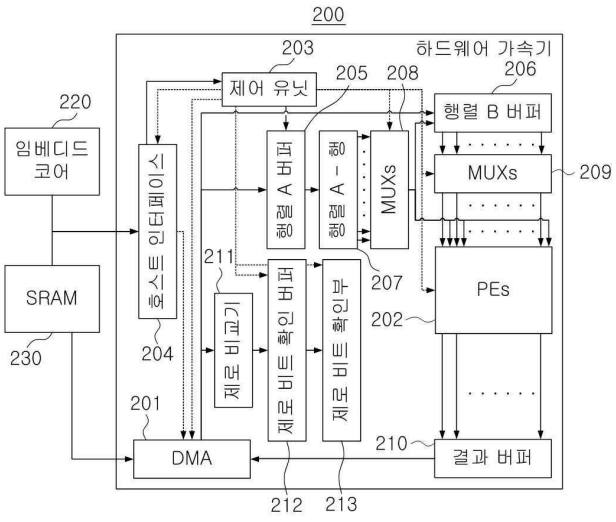
전체 청구항 수 : 총 17 항

(54) 발명의 명칭 **제로값을 피연산자로 갖는 연산자에 대한 연산을 스킵하는 연산 방법 및 연산 장치**

(57) 요 약

제로값을 피연산자로 갖는 연산자에 대한 연산을 스킵하는 연산 방법 및 연산 장치가 개시된다. 최근, 다양한 센서들에 의해 전달되는 데이터를 처리하기 위해 센서 허브 SoC(System on Chip)와 같은 특수 MCU(Micro Controller Unit)가 모바일 및 착용 형 휴대 장치에 채용되고 있다. 본 발명의 실시예들은 6-축 센서에 기반하여 운동 방향을 검출하는 하드웨어 가속기에 관한 것이다. 하드웨어 가속기의 구조는 센서 퓨전 알고리즘의 프로파일링(profiling)에 기초하여 설계될 수 있다. 성능 평가에서 본 발명의 실시예들에 따른 하드웨어 가속기는 100% 이상 실행 시간이 향상됨을 보여준다.

대 표 도 - 도2



(52) CPC특허분류
G06F 17/16 (2013.01)
G06F 9/30069 (2013.01)
G06F 9/30101 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1415135205
부처명 산업통상자원부
연구관리전문기관 한국산업기술평가관리원
연구사업명 시스템반도체상용화기술개발
연구과제명 휴대용스마트기기용 146mA/MHz 이하 저전력 센서신호처리 MCU 개발
기 여 율 1/1
주관기관 스탠딩에그(주)
연구기간 2014.06.01 ~ 2016.09.30

명 세 서

청구범위

청구항 1

연산 장치의 연산 방법에 있어서,

상기 연산 장치에서 복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 단계;

상기 연산 장치에서 상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 단계; 및

상기 복수의 연산기들 각각에서 상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 단계

를 포함하는 것을 특징으로 하는 연산 방법.

청구항 2

제1항에 있어서,

상기 복수의 제1 피연산자들은 a개의 행과 b개의 열로 구성된 제1 행렬의 n번째 행의 원소들이고,

상기 복수의 제2 피연산자들은 c개의 행과 d개의 열로 구성된 제2 행렬의 m번째 행의 원소들이고,

상기 a, 상기 b, 상기 c 및 상기 d는 자연수이고,

상기 n은 상기 a 이하의 자연수이고,

상기 m은 상기 c 이하의 자연수인 것을 특징으로 하는 연산 방법.

청구항 3

제2항에 있어서,

상기 브로드캐스팅하는 단계는,

상기 제1 행렬의 n번째 행의 원소들을 순차적으로 브로드캐스팅하고,

상기 연산을 처리하는 단계는,

상기 복수의 연산기 각각에서 상기 제1 행렬의 n번째 행의 원소들 중 하나의 원소와 상기 제2 행렬의 m번째 행의 모든 원소들 중 대응하는 원소간의 곱셈 연산을 처리하는 것을 특징으로 하는 연산 방법.

청구항 4

제3항에 있어서,

상기 제로 비트 확인 버퍼는 상기 제1 행렬의 n번째 행의 원소들 중 제로값인 원소들을 표시하기 위한 비트열을 저장하는 것을 특징으로 하는 연산 방법.

청구항 5

제2항에 있어서,

제1 행렬 버퍼에 e개의 행과 f개의 열로 구성된 제3 행렬을 로딩하는 단계; 및

상기 제3 행렬의 m번째 행이 상기 제2 행렬의 m번째 열로 치환되도록 제2 행렬 버퍼에 저장하여 상기 제3 행렬의 전치 행렬로서 상기 제2 행렬을 제2 행렬 버퍼에 로딩하는 단계

를 더 포함하고,

상기 e 및 상기 f는 자연수인 것을 특징으로 하는 연산 방법.

청구항 6

제5항에 있어서,

상기 제3 행렬의 전치 행렬로서 상기 제2 행렬이 상기 제2 행렬 버퍼에 로딩된 이후 상기 제1 행렬 버퍼에 상기 제1 행렬을 로딩하는 단계

를 더 포함하는 것을 특징으로 하는 연산 방법.

청구항 7

제6항에 있어서,

상기 제3 행렬은 상기 제1 행렬과 동일한 행렬이고,

상기 a와 상기 e의 값이 동일하고, 상기 b와 상기 f의 값이 동일한 것을 특징으로 하는 연산 방법.

청구항 8

제1항에 있어서,

상기 복수의 연산기 각각의 연산 결과를 결과 버퍼에 누적하여 저장하는 단계

를 더 포함하고,

상기 결과 버퍼는 상기 복수의 연산기 각각에 대응하는 복수의 저장소를 포함하여 대응하는 연산기의 연산 결과를 저장하는 것을 특징으로 하는 연산 방법.

청구항 9

연산 장치에 있어서,

복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 제로 비트 확인부;

상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 브로드캐스팅부; 및

상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 상기 복수의 연산기들

을 포함하는 것을 특징으로 하는 연산 장치.

청구항 10

제9항에 있어서,

상기 복수의 제1 피연산자들은 a개의 행과 b개의 열로 구성된 제1 행렬의 n번째 행의 원소들이고,

상기 복수의 제2 피연산자들은 c개의 행과 d개의 열로 구성된 제2 행렬의 m번째 행의 원소들이고,

상기 a, 상기 b, 상기 c 및 상기 d는 자연수이고,

상기 n은 상기 a 이하의 자연수이고,

상기 m은 상기 c 이하의 자연수인 것을 특징으로 하는 연산 장치.

청구항 11

제10항에 있어서,

상기 브로드캐스팅부는,

상기 제1 행렬의 n번째 행의 원소들을 순차적으로 브로드캐스팅하고,

상기 복수의 연산기 각각은,

상기 제1 행렬의 n번째 행의 원소들 중 하나의 원소와 상기 제2 행렬의 m번째 행의 모든 원소들 중 대응하는 원소간의 곱셈 연산을 처리하는 것을 특징으로 하는 연산 장치.

청구항 12

제10항에 있어서,

상기 제로 비트 확인부는, 상기 제1 행렬의 n번째 행의 원소들 중 제로값인 원소들을 표시하기 위한 비트열을 생성하여 상기 제로 비트 확인 버퍼에 저장하는 것을 특징으로 하는 연산 장치.

청구항 13

제10항에 있어서,

제1 행렬 버퍼에 e개의 행과 f개의 열로 구성된 제3 행렬을 로딩하고, 상기 제3 행렬의 m번째 행이 상기 제2 행렬의 m번째 열로 치환되도록 제2 행렬 버퍼에 저장하여 상기 제3 행렬의 전치 행렬로서 상기 제2 행렬을 제2 행렬 버퍼에 로딩하는 제어부

를 더 포함하고,

상기 e 및 상기 f는 자연수인 것을 특징으로 하는 연산 장치.

청구항 14

제13항에 있어서,

상기 제어부는,

상기 제3 행렬의 전치 행렬로서 상기 제2 행렬이 상기 제2 행렬 버퍼에 로딩된 이후 상기 제1 행렬 버퍼에 상기 제1 행렬을 로딩하는 것을 특징으로 하는 연산 장치.

청구항 15

제14항에 있어서,

상기 제3 행렬은 상기 제1 행렬과 동일한 행렬이고,

상기 a와 상기 e의 값이 동일하고, 상기 b와 상기 f의 값이 동일한 것을 특징으로 하는 연산 장치.

청구항 16

제9항에 있어서,

상기 복수의 연산기 각각의 연산 결과를 누적하여 저장하는 결과 버퍼

를 더 포함하고,

상기 결과 버퍼는 상기 복수의 연산기 각각에 대응하는 복수의 저장소를 포함하여 대응하는 연산기의 연산 결과를 저장하는 것을 특징으로 하는 연산 장치.

청구항 17

제9항 내지 제16항 중 어느 한 항의 연산 장치를 포함하는 것을 특징으로 하는 센서 허브 MCU(Micro Controller Unit).

발명의 설명

기 술 분 야

[0001] 아래의 설명은 제로값을 피연산자로 갖는 연산자에 대한 연산을 스킵하는 연산 방법 및 연산 장치에 관한 것이다.

배 경 기 술

[0002] IoT(Internet of Things) 서비스의 발전과 함께 최근 스마트 기기에서 다양한 센서들이 사용되고 있다. 도 1은 종래기술에 있어서, 센서 허브의 예를 도시한 도면이다. 스마트 기기(100)에서 자이로 센서(Gyro Sensor), 모션 센서(motion sensor), 주변광 센서(Ambient Light sensor), 가속도 센서(accelerometer sensor), 온도 습도 센서(Temperature Humidity sensor), 압력 센서(Pressure Sensor) 등과 같은 센서들(110)의 수는 증가하고 있으며, 센서들(110)의 센싱 데이터의 처리를 위해 스마트 기기(100)에 포함된 AP(Application Processor, 110)의 대기 시간은 점차 줄어들고 있기 때문에, 에너지 소비가 증가한다. 이때, 스마트 기기(100)에서 센서들(110)의 수의 증가는, 센서 데이터의 처리가 고성능의 그리고 높은 전력 소비량을 갖는 AP(110)에서 다루어질 때 특히 더 많은 에너지를 소비하게 된다. 따라서, 센서 허브 MCU(Micro Controller Unit)(120)가 저전력 임베디드 프로세서에 기반하여 훨씬 적은 에너지 소비로 센서 데이터를 처리하기 위해 스마트 기기(100)에 도입될 수 있다. 센서 허브 MCU(120)는 다양한 센서에 의해 취득된 데이터의 처리를 위해 특화된 저전력 MCU로서, 센서허브에서 수행되는 알고리즘은 센서 신호를 통해서 디바이스 혹은 사용자의 방향을 계산하며, 점차 정확한 방향 감지가 요구됨에 따라서 칼만 필터를 활용한 센서 퓨전 알고리즘도 활용되고 있다. 이러한 칼만 필터는 다양한 상호 보완적 센서들을 이용하기 때문에 높은 복잡도를 갖는다.

[0003] FPGA(Field Programmable Gate Array)를 갖는 전체 칼만 필터(Kalman filter) 알고리즘을 위한 효율적인 하드웨어 가속기를 설계하기 위한 종래기술들이 존재한다. 전용 하드웨어 가속기가 특정 타겟 시스템들이나 센서들을 위해 효율적일 수 있으나, 프로그래머빌리티(programmability)의 결여는 새로운 센서 및/또는 알고리즘들을 이용하는 시스템과 같은 다른 시스템들이 이러한 솔루션을 사용하는 것을 제한한다는 문제점이 있다. 따라서, 유연성(flexibility)과 프로그래머빌리티의 성능 향상을 달성할 수 있는, 센서 허브 MCU 구조(임베디드 프로세서와 하드웨어 가속기를 포함하는)가 요구된다.

[0004] <참고문헌: S. Cruz, D.M. Munoz, M. Conde an C.H. Llanos, G.A. Borges, "FPGA implementation of a sequential extended Kalman filter algorithm applied to mobile robotics localization problem," Circuits and Systems, pp. 1-4, Feb 2013.>

발명의 내용

해결하려는 과제

[0005] 본 발명의 실시예들은 센서 허브 MCU(Micro Controller Unit)를 위한 하드웨어 가속기에 관한 것으로, 방향 추정의 정확도를 향상시키고, 방향 추정을 위한 에너지 소비를 줄이기 위해 센서 퓨전을 위한 복합 칼만 필터(complex Kalman filter)를 처리할 수 있는 연산 방법 및 장치를 제공한다.

[0006] 또한, 더 확장된 프로그래머빌리티(programmability)를 갖고, 칼만 필터 처리 시간에 대한 성능을 100% 이상 향상시킬 수 있는 연산 방법 및 장치를 제공한다.

[0007] 행렬을 메모리로부터 레지스터에 저장할 때, 해당 행의 원소가 0(zero)인지 여부를 저장하는 제로 비트 레지스터를 구비하여, 연산하고자 하는 원소가 0의 값을 가진 경우에는 해당 원소에 대한 연산을 생략(skip)함으로써, 연산 수행 시간 및 연산에 요구되는 전력 소모를 줄일 수 있는 연산 방법 및 장치를 제공한다.

과제의 해결 수단

[0008] 연산 장치의 연산 방법에 있어서, 상기 연산 장치에서 복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 단계; 상기 연산 장치에서 상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 단계; 및 상기 복수의 연산기들 각각에서 상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 단계를 포함하는 것을 특징으로 하는 연산 방법을 제공한다.

[0009] 일측에 따르면, 상기 복수의 제1 피연산자들은 a개의 행과 b개의 열로 구성된 제1 행렬의 n번째 행의 원소들이

고, 상기 복수의 제2 피연산자들은 c개의 행과 d개의 열로 구성된 제2 행렬의 m번째 행의 원소들이고, 상기 a, 상기 b, 상기 c 및 상기 d는 자연수이고, 상기 n은 상기 a 이하의 자연수이고, 상기 m은 상기 c 이하의 자연수인 것을 특징으로 할 수 있다.

[0010] 다른 측면에 따르면, 상기 브로드캐스팅하는 단계는, 상기 제1 행렬의 n번째 행의 원소들을 순차적으로 브로드캐스팅하고, 상기 연산을 처리하는 단계는, 상기 복수의 연산기 각각에서 상기 제1 행렬의 n번째 행의 원소들 중 하나의 원소와 상기 제2 행렬의 m번째 행의 모든 원소들 중 대응하는 원소간의 곱셈 연산을 처리하는 것을 특징으로 할 수 있다.

[0011] 또 다른 측면에 따르면, 상기 제로 비트 확인 버퍼는 상기 제1 행렬의 n번째 행의 원소들 중 제로값인 원소들을 표시하기 위한 비트열을 저장하는 것을 특징으로 할 수 있다.

[0012] 또 다른 측면에 따르면, 상기 연산 방법은 제1 행렬 버퍼에 e개의 행과 f개의 열로 구성된 제3 행렬을 로딩하는 단계; 및 상기 제3 행렬의 m번째 행이 상기 제2 행렬의 m번째 열로 치환되도록 제2 행렬 버퍼에 저장하여 상기 제3 행렬의 전치 행렬로서 상기 제2 행렬을 제2 행렬 버퍼에 로딩하는 단계를 더 포함하고, 상기 e 및 상기 f는 자연수인 것을 특징으로 할 수 있다.

[0013] 또 다른 측면에 따르면, 상기 연산 방법은 상기 제3 행렬의 전치 행렬로서 상기 제2 행렬이 상기 제2 행렬 버퍼에 로딩된 이후 상기 제1 행렬 버퍼에 상기 제1 행렬을 로딩하는 단계를 더 포함하는 것을 특징으로 할 수 있다.

[0014] 또 다른 측면에 따르면, 상기 제3 행렬은 상기 제1 행렬과 동일한 행렬이고, 상기 a와 상기 e의 값이 동일하고, 상기 b와 상기 f의 값이 동일한 것을 특징으로 할 수 있다.

[0015] 또 다른 측면에 따르면, 상기 연산 방법은 상기 복수의 연산기 각각의 연산 결과를 결과 버퍼에 누적하여 저장하는 단계를 더 포함하고, 상기 결과 버퍼는 상기 복수의 연산기 각각에 대응하는 복수의 저장소를 포함하여 대응하는 연산기의 연산 결과를 저장하는 것을 특징으로 할 수 있다.

[0016] 연산 장치의 제1 행렬과 제2 행렬간의 연산 방법에 있어서, 상기 제1 행렬은 a개의 행과 b개의 열로 구성되고, 상기 제2 행렬은 c개의 행과 d개의 열로 구성되며, 상기 연산 방법은, 상기 연산 장치에서 상기 제1 행렬의 n번째 행을 제1 행렬 버퍼에 로딩하고, 상기 제2 행렬을 제2 행렬 버퍼에 로딩하는 단계; 상기 연산 장치에서 상기 제1 행렬의 n번째 행의 i번째 원소와 상기 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈을 계산하는 단계; 및 상기 연산 장치에서 상기 제1 행렬의 n번째 행의 i번째 원소와 상기 제2 행렬의 m번째 행의 j번째 원소간의 곱셈 결과를 결과 행렬을 저장하는 버퍼의 j 번째 저장소에 누적하여 저장하는 단계를 포함하고, 상기 곱셈을 계산하는 단계는, 상기 제1 행렬의 n번째 행의 i번째 원소의 값이 0(zero)인 경우, 상기 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈을 생략하고, 상기 a, 상기 b, 상기 c 및 상기 d는 자연수이고, 상기 n은 상기 a 이하의 자연수이고, 상기 i는 상기 b 이하의 자연수이고, 상기 m은 상기 c 이하의 자연수이고, 상기 j는 상기 d 이하의 자연수인 것을 특징으로 하는 연산 방법을 제공한다.

[0017] 연산 장치에 있어서, 복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 제로 비트 확인부; 상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 브로드캐스팅부; 및 상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 상기 복수의 연산기들을 포함하는 것을 특징으로 하는 연산 장치를 제공한다.

[0018] 제1 행렬과 제2 행렬간의 행렬 연산을 위한 연산 장치에 있어서, 상기 제1 행렬은 a개의 행과 b개의 열로 구성되고, 상기 제2 행렬은 c개의 행과 d개의 열로 구성되며, 상기 연산 장치는, 상기 제1 행렬의 n번째 행을 로딩하는 제1 행렬 버퍼; 상기 제2 행렬을 로딩하는 제2 행렬 버퍼; 상기 제1 행렬의 n번째 행의 i번째 원소와 상기 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈을 계산하는 곱셈부; 및 상기 제1 행렬의 n번째 행의 i번째 원소와 상기 제2 행렬의 m번째 행의 j번째 원소간의 곱셈 결과를 j번째 저장소에 누적하여 저장하는 누적기를 포함하고, 상기 곱셈부는, 상기 제1 행렬의 n번째 행의 i번째 원소의 값이 0(zero)인 경우, 상기 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈을 생략하고, 상기 a, 상기 b, 상기 c 및 상기 d는 자연수이고, 상기 n은 상기 a 이하의 자연수이고, 상기 i는 상기 b 이하의 자연수이고, 상기 m은 상기 c 이하의 자연수이고, 상기 j는 상기 d 이하의 자연수인 것을 특징으로 하는 연산 장치가 제공된다.

[0019] 상기 연산 장치를 포함하는 것을 특징으로 하는 센서 허브 MCU(Micro Controller Unit)가 제공된다.

발명의 효과

[0020] 센서 허브 MCU(Micro Controller Unit)를 위한 하드웨어 가속기에 관한 것으로, 방향 추정의 정확도를 향상시키고, 방향 추정을 위한 에너지 소비를 줄이기 위한 센서 퓨전 알고리즘의 복합 칼만 필터(complex Kalman filter)를 처리할 수 있다.

[0021] 또한, 더 확장된 프로그래머빌리티(programmability)를 갖고, 칼만 필터 처리 시간에 대한 성능을 100% 이상 향상시킬 수 있다.

[0022] 행렬을 메모리로부터 레지스터에 저장할 때, 해당 행의 원소가 0(zero)인지 여부를 저장하는 제로 비트 레지스터를 구비하여, 연산하고자 하는 원소가 0의 값을 가진 경우에는 해당 원소에 대한 연산을 생략(skip)함으로써, 연산 수행 시간 및 연산에 요구되는 전력 소모를 줄일 수 있다.

도면의 간단한 설명

[0023] 도 1은 종래기술에 있어서, 센서 허브의 예를 도시한 도면이다.

도 2는 본 발명의 일실시예에 있어서, 하드웨어 가속기를 갖는 센서 허브 MCU의 전체 구조의 예를 도시한 도면이다.

도 3은 본 발명의 일실시예에 있어서, 행렬 정보의 전달 과정의 예를 도시한 도면이다.

도 4는 본 발명의 일실시예에 있어서, 행렬들의 데이터의 로딩 과정의 예를 도시한 도면이다.

도 5는 본 발명의 일실시예에 있어서, MAC 연산의 처리 과정의 예를 도시한 도면이다.

도 6은 본 발명의 일실시예에 있어서, 연산 결과의 저장 과정의 예를 도시한 도면이다.

도 7은 본 발명의 일실시예에 있어서, 원소 처리 구조(Processing Element (PE) Architecture)의 예를 도시한 도면이다.

도 8 내지 도 13은 본 발명의 일실시예에 있어서, 행렬 곱셈의 처리 과정의 예를 도시한 도면이다.

발명을 실시하기 위한 구체적인 내용

[0024] 이하, 실시예를 첨부한 도면을 참조하여 상세히 설명한다.

[0026] 1. 구조

[0028] A. 센서 퓨전 알고리즘 분석

[0029] 대부분의 센서 퓨전 메커니즘들은 칼만 필터를 포함하는 6-축 또는 9-축(가속도계, 자이로스코프 및 자력계)를 이용한다. 칼만 필터(Kalman filter)는 다양한 센서들로부터 얻어지는 데이터에 기반하여 방향을 정확하게 예측하기 위해 이용된다. 칼만 필터에 기반한 센서 퓨전의 주요 연산들은 두 가지 부분으로 분류될 수 있다. 첫 번째 부분은 기 정의된 상태 방정식에 기반하여 현재 상태를 예측하는 것이고, 두 번째 부분은 칼만 이득값(Kalman gain value)를 이용하여 예측된 방향을 보정하는 것이다. 이러한 센서 퓨전 알고리즘의 분석은 일례로, 오픈 소스인 프리스케일 센서 퓨전 소프트웨어(Freescale sensor fusion software, Freescale Sensor Fusion, http://www.freescale.com/ 참조) 및 프로파일링을 위한 DS-5 툴(ARM Development Tools, http://ds.arm.com/ 참조)에 기반하여 수행될 수 있다. DS-5에 기반한 프로파일링 결과는 아래 표 1에 나타난 바와 같이 칼만 이득값 계산과 오차 공분산 행렬(error covariance matrix) 계산이라는 두 가지 주요한 성능 병목 현상(performance bottleneck)의 기능들을 확인해준다. 이러한 기능들의 주요 연산들은 행렬 곱셈(matrix multiplication)과 행렬 전치(matrix transpose)로, 이러한 두 가지 연산들은 전체 실행 시간의 약 80%를 사용한다.

표 1

Function name	Execution time(%)
Rotation matrix	11
Sensor estimation & Update measurement matrix	1
Kalman gain	42
Error estimation & Error corrections	5
Error covariance matrix	35
Re-create the noise covariance matrix	4

[0030]

[0031] 표 1은 각 기능별 실행 시간의 예를 나타내고 있다.

[0033] B. 하드웨어 가속기 및 센서 허브 구조

[0034] 앞서 살펴본 바와 같이, 센서 퓨전의 주요 동작들은 행렬 곱셈과 행렬 전치와 같은 행렬 조작 연산들이다. 다른 센서 관련 연산들이 보통 행렬 조작 연산들에 기반할 뿐만 아니라, 이러한 연산들이 타겟 센서 퓨전 알고리즘의 주 커널(kernels)이기 때문에, 하드웨어 가속기의 구조는 행렬 조작 연산들에 초점을 맞춰 선택될 수 있다. 비록, 행렬 조작 연산들이 잘 알려져 있고, 다수의 연구들이 수행되고 있다고 해도, 센서 퓨전 프로세싱은 서로 다른 수의 센서들을 이용할 수 있기 때문에, 센서 허브 MCU(Micro Controller Unit)를 위한 하드웨어 가속기의 구조는 신중하게 설계되어야 한다. 행렬의 다양한 크기는 심지어 동일한 센서 퓨전 알고리즘에서도 조작되어야 한다. 예를 들어, 센서 퓨전 알고리즘이 9-축 센서들로 실행될 때, 12 × 12 행렬과 12 × 6 행렬간의 곱셈 및 6 × 12 행렬과 12 × 12 행렬간의 곱셈과 같은 행렬들의 곱셈이 수행되어야 한다. 제안된 하드웨어 가속기의 구조는 센서 퓨전 프로세싱의 이러한 특성들을 활용할 수 있도록 설계될 수 있다. 제안된 하드웨어 가속기는 행렬들의 다양한 크기를 처리하기 위해, 행렬 A에서 원소의 브로드캐스팅 방식(broadcasting scheme) 및 적응 제어 메커니즘을 채택할 수 있다.

[0035] 도 2는 본 발명의 일실시예에 있어서, 하드웨어 가속기를 갖는 센서 허브 MCU의 전체 구조의 예를 도시한 도면이다. 제안된 하드웨어 가속기(Hardware accelerator, 200)는 행렬 A와 행렬 B를 곱하기 위해 행렬 B의 전체와 행렬 A의 두 개의 행을 저장할 수 있다.

[0036] 곱셈이 시작될 때 DMA(Direct Memory Access, 201)는 행렬 B 전체와 행렬 A의 하나의 행을 로드할 수 있고, 곱셈이 진행됨에 따라 행렬 A의 나머지 행을 순차적으로 로드할 수 있다. PEs(Processing Elements, 202)는 MAC(multiplier-accumulator) 연산을 실행할 수 있고, 제어 유닛(Control unit, 203)은 다양한 행렬 크기를 위한 연산을 위한 제어신호를 구성요소들로 제공할 수 있다.

[0037] 1) 임베디드 코어(Embedded core, 220)는 호스트 인터페이스(Host interface, 204)를 통해 제어 유닛(203)으로 행렬 정보(행렬 크기, 행렬 주소(matrix address))를 전달할 수 있다. 도 3은 본 발명의 일실시예에 있어서, 행렬 정보의 전달 과정의 예를 도시한 도면이다. 도 3의 굵은 화살표들은 임베디드 코어(220)로부터 연산을 수행할 행렬 크기와 행렬 주소와 같은 행렬 정보가 호스트 인터페이스(204)를 통해 제어 유닛(203)으로 전달되는 과정을 나타내고 있다.

[0038] 2) 행렬 A의 두 개의 행 데이터와 행렬 B의 전체 데이터가 SRAM(Static Random Access Memory, 230)에서 DMA(201)를 통해 행렬 A 버퍼(205)와 행렬 B 버퍼(206)로 로딩될 수 있다. 도 4는 본 발명의 일실시예에 있어서, 행렬들의 데이터의 로딩 과정의 예를 도시한 도면이다. 도 4의 굵은 화살표들은 SRAM(230)과 같은 저장소에서 DMA(201)를 거쳐 행렬 A의 두 개의 행이 행렬 A 버퍼(205)로 로딩되고, 행렬 B가 행렬 B 버퍼(206)으로 로딩되는 과정을 나타내고 있다. 행렬 A - 행(207)은 행렬 A의 하나의 행을 의미할 수 있다.

[0039] 3) PEs(202)는 내부 레지스터(행렬 A 버퍼(205) 및 행렬 B 버퍼(206))로부터 데이터를 로딩하고, MAC(multiplier-accumulator) 연산을 실행할 수 있다. 도 5는 본 발명의 일실시예에 있어서, MAC 연산의 처리 과정의 예를 도시한 도면이다. 행렬 A를 위한 MUXs(multiplexers, 208)와 행렬 B를 위한 MUXs(209)를 통해 행렬 A의 하나의 행에 대한 각각의 원소들 그리고 행렬 B의 하나의 행에 대한 각각의 원소들이 PEs(202)로 전달될 수 있다. MAC 연산은 행렬 A의 첫 번째 행과 행렬 B의 전체 행들 각각에 대한 곱셈 연산 및 곱셈 연산의 결과들의 누적 연산을 의미할 수 있다. 이러한 MAC 연산에 대해서는 이후 더욱 자세히 설명한다.

[0040] 4) MAC 연산이 완료되면, 결과 버퍼(Result Buffer, 210)는 DMA(201)를 통해 연산 결과를 SRAM(230)에 저장할 수 있다. 도 6은 본 발명의 일실시예에 있어서, 연산 결과의 저장 과정의 예를 도시한 도면이다.

[0041] 도 7은 본 발명의 일실시예에 있어서, 원소 처리 구조(Processing Element (PE) Architecture)의 예를 도시한 도면이다. 제안된 하드웨어 가속기를 포함하는 전체 센서 허브 구조에서, 행렬 A의 하나의 원소는 브로드캐스팅되어 모든 곱셈기들(multipliers, 도 7에 도시된 복수의 "MUL"들)을 위한 피연산자가 될 수 있다. 곱셈기들을 위한 다른 피연산자들은 행렬 B의 전체 행들의 원소들 각각이 될 수 있다. 행렬 A - 행 버퍼(207)에 저장된 모든 원소들이 브로드캐스팅되고 행렬 B 버퍼(206)에 저장된 행렬 B의 모든 행들과 곱셈 및 누적이 되면, 누적기들(도 7에 도시된 복수의 덧셈기 "+"들과 복수의 레지스터 "Reg"들)은 행렬 A의 하나의 행의 하나의 원소와 행렬 B의 하나의 행의 모든 원소들 각각을 위한 곱셈 결과들의 행을 포함할 수 있다. 목적 행렬의 하나의 행을 위한 이러한 결과들은 DMA(201)를 통해 SRAM(230)로 전송될 수 있다. 이러한 계산은 전체 행렬 A와 행렬 B의 행렬 곱셈을 계산하기 위해 행렬 A의 모든 행들의 수만큼 반복될 수 있다. 신호 "is_rawA_end"는 행렬 A의 하나의 행의 모든 원소들에 대해 계산이 완료되었음을 알리기 위한 신호일 수 있다.

[0043] 연산 장치의 연산 방법은 연산 장치에서 복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 단계, 상기 연산 장치에서 상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 단계 및 상기 복수의 연산기들 각각에서 상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 단계를 포함할 수 있다. 이를 위한 연산 장치는 복수의 제1 피연산자들 중 제로(zero)값을 갖는 제1 피연산자를 확인하여 제로 비트 확인 버퍼를 통해 제로값을 갖는 제1 피연산자를 지시하는 제로 비트 확인부, 상기 복수의 제1 피연산자들을 순차적으로 상기 연산 장치가 포함하는 복수의 연산기들로 브로드캐스팅하되, 상기 제로 비트 확인 버퍼를 통해 제로값을 갖는 것으로 확인된 제1 피연산자의 브로드캐스팅을 스킵하는 브로드캐스팅부 및 상기 복수의 연산기들 각각에 대응하여 전달되는 복수의 제2 피연산자들과 상기 브로드캐스팅된 제1 피연산자간의 연산을 처리하는 상기 복수의 연산기들을 포함할 수 있다. 예를 들어, 제로 비트 확인부는 앞서 설명한 제로 비트 확인부(213)에 대응할 수 있고, 브로드캐스팅부는 행렬 A - 행(207)과 MUXs(208)에 대응할 수 있다. 또한, 복수의 연산기들은 PEs(202)가 포함하는 곱셈기들에 대응될 수 있다.

[0044] 보다 구체적인 예를 설명하면, 제1 행렬은 a개의 행과 b개의 열로 구성될 수 있고, 제2 행렬은 c 개의 행과 d개의 열로 구성될 수 있다. 여기서, a, b, c, d는 모두 자연수일 수 있다. 예를 들어, 제1 행렬은 상술한 행렬 A에 대응될 수 있고, 제2 행렬은 상술한 행렬 B에 대응될 수 있다.

[0045] 연산 장치는 제1 행렬의 n번째 행을 제1 행렬 버퍼에 로딩하고, 제2 행렬을 제2 행렬 버퍼에 로딩할 수 있다. 여기서, 제1 행렬 버퍼는 상술한 행렬 A 버퍼(205)에 대응될 수 있고, 제2 행렬 버퍼는 상술한 행렬 B 버퍼(206)에 대응될 수 있다.

[0046] 이때, 연산 장치는 제1 행렬의 n번째 행의 i번째 원소와 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈을 계산할 수 있다. 예를 들어, 제1 행렬의 첫 번째 행의 첫 번째 원소인 a(1, 1)이 제2 행렬의 첫 번째 행의 모든 원소들 각각과 곱셈 연산될 수 있다. 또한, a(1, 2)은 제2 행렬의 두 번째 행의 모든 원소들 각각과도 곱셈 연산될 수 있다. 다시 말해, 상기 n은 상기 a 이하의 자연수이고, 상기 i는 상기 b 이하의 자연수이며, 상기 m은 상기 c 이하의 자연수일 수 있다. 여기서 연산 장치는 제1 행렬의 n번째 행의 i번째 원소의 값이 0(zero)인 경우, 브로드캐스팅을 수행하지 않음으로써 제2 행렬의 m번째 행의 모든 원소들 각각간의 곱셈 및 누적 연산을 생략할 수 있다. 이때, 연산 장치는 제1 행렬의 n번째 행의 원소들 중 0의 값을 가진 원소를 제로 비트 확인 버퍼(212)를 통해 미리 표시해놓을 수 있으며, 제로 비트 확인 버퍼(212)를 통해 제1 행렬의 n번째 행의 i번째

원소의 값이 0인지 여부를 확인할 수 있다.

[0047] 또한, 연산 장치는 제1 행렬의 n번째 행의 i번째 원소와 제2 행렬의 m번째 행의 j번째 원소간의 곱셈 결과를 결과 행렬을 저장하는 버퍼의 j 번째 저장소에 누적하여 저장할 수 있다. 이때, 상기 j는 상기 d 이하의 자연수일 수 있다. 예를 들어, a(1, 1)과 제2 행렬의 첫 번째 행의 첫 번째 원소인 b(1, 1)간의 곱셈 결과가 결과 행렬의 첫 번째 저장소 c(1, 1)에 저장될 수 있다. 또한, a(1, 2)과 b(2, 1)간의 곱셈 결과가 결과 행렬의 첫 번째 저장소 c(1, 1)에 누적되어 저장될 수 있다.

[0048] 전치 행렬의 계산을 위해 연산 장치는 제1 행렬 버퍼에 e개의 행과 f개의 열로 구성된 제3 행렬을 먼저 로딩할 수 있다. 이때, 연산 장치는 제1 행렬 버퍼에 로딩된 제3 행렬의 m번째 행을 제2 행렬의 m번째 열로 치환하여 제3 행렬의 전치 행렬로서 제2 행렬을 제2 행렬 버퍼에 로딩할 수 있다. 이 경우, 상기 e와 상기 d의 값이 동일하고, 상기 f와 상기 c의 값이 동일할 수 있다. 이는 제1 행렬과 제3 행렬의 전치 행렬인 제2 행렬간의 곱셈 연산을 처리하기 위해 이용될 수 있다.

[0049] 또한, 제3 행렬은 제1 행렬과 동일한 행렬일 수 있다. 이 경우 상기 a와 상기 e의 값이 동일하고, 상기 b와 상기 f의 값이 동일할 수 있다. 이는 제1 행렬과 제1 행렬의 전치 행렬인 제2 행렬간의 곱셈 연산을 처리하기 위해 이용될 수 있다. 예를 들어, 제3 행렬이 제1 행렬과 동일하다면, 제3 행렬의 전치행렬인 제2 행렬과 제1 행렬간의 곱셈 연산은 결국 제1 행렬(또는 제3 행렬)과 제1 행렬(또는 제3 행렬)의 전치 행렬간의 곱셈 연산을 의미할 수 있다.

[0050] 연산 장치는 하드웨어 가속기로서 센서 허브 MCU(Micro Controller Unit)에 포함될 수도 있다.

[0051] 도 8 내지 도 13은 본 발명의 일실시예에 있어서, 행렬 곱셈의 처리 과정의 예를 도시한 도면이다.

[0052] 도 8은 12 × 6의 행렬 A(810)와 6 × 12의 행렬 B(820)의 곱셈 연산을 통해 12 × 12의 결과 행렬(830)을 생성하고자 함을 나타내고 있다. 곱셈 연산을 위해 행렬 A(810)의 첫 번째 행의 첫 번째 원소 a(1, 1)가 행렬 B(820)의 첫 번째 행의 원소들(b(1, 1), …, b(1, 12)) 각각과 곱셈될 수 있다.

[0053] 도 9는 행렬 A(810)의 첫 번째 행의 첫 번째 원소 a(1, 1)와 행렬 B(810)의 첫 번째 행의 첫 번째 원소 b(1, 1)의 곱셈 결과가 결과 행렬(830)의 첫 번째 행의 첫 번째 원소 c(1, 1)에 저장됨을 나타내고 있다.

[0054] 도 10은 행렬 A(810)의 첫 번째 행의 첫 번째 원소 a(1, 1)가 행렬 B의 첫 번째 행의 원소들(b(1, 1), …, b(1, 12)) 각각과 곱셈될 수 있고, 그에 따라 결과 행렬(830)의 첫 번째 행의 마지막 원소 c(1, 12)에 원소 a(1, 1)와 원소 b(1, 12)의 곱셈 결과가 저장될 수 있음을 나타내고 있다.

[0055] 도 11은 행렬 A(810)의 첫 번째 행의 두 번째 원소 a(1, 2)가 행렬 B(820)의 두 번째 행의 원소들(b(2, 1), …, b(2, 12)) 각각과 곱셈될 수 있으며, 이를 위해 우선, 원소 a(1, 2)와 원소 b(2, 1)의 곱셈 결과가 결과 행렬(830)의 첫 번째 행의 첫 번째 원소 c(1, 1)에 누적되는 예를 나타내고 있다. 이미 원소 c(1, 1)에 "a(1, 1)*b(1, 1)"의 값이 저장되어 있기 때문에 원소 a(1, 2)와 원소 b(2, 1)의 곱셈 결과인 "a(1, 2)*b(2, 1)"이 누적되어 원소 c(1, 1)의 값은 "a(1, 1)*b(1, 1)+ a(1, 2)*b(2, 1)"가 될 수 있다.

[0056] 도 12는 행렬 A(810)의 첫 번째 행의 2 번째 원소 a(1, 2)가 행렬 B(820)의 두 번째 행의 원소들(b(2, 1), …, b(2, 12)) 각각과 곱셈됨에 따라 결과 행렬(830)의 첫 번째 행의 마지막 원소 c(1, 12)에 원소 a(1, 2)와 원소 b(2, 12)의 곱셈 결과가 누적된 예를 나타내고 있다. 이미 원소 c(1, 12)에 "a(1, 1)*b(1, 12)"의 값이 저장되어 있기 때문에 원소 a(1, 2)와 원소 b(2, 12)의 곱셈 결과인 "a(1, 2)*b(2, 12)"가 누적되어 원소 c(1, 12)의 값은 "a(1, 1)*b(1, 12)+ a(1, 2)*b(2, 12)"가 될 수 있다.

[0057] 도 13은 행렬 A(810)의 첫 번째 행의 원소들과 행렬 B(820)의 전체 행의 원소들간의 곱셈 및 누적이 진행됨에 따라 결과 행렬(830)의 첫 번째 행의 첫 번째 원소 c(1, 1)에 곱셈 결과들이 누적된 예를 나타내고 있다. 결과적으로, 행렬들간의 행과 행간의 곱셈 연산을 통해 c(1, 1)에는 행렬 A(810)의 첫 번째 행의 원소들(a(1, 1), a(1, 2), …, a(1, 6))과 행렬 B(820)의 첫 번째 열의 원소들 각각간의 곱셈 결과가 누적될 수 있다.

[0058] 이러한 도 8 내지 도 13의 과정은 행렬 A(810)의 나머지 행들에 대해서도 순차적으로 반복 처리되어 결과 행렬에 누적됨으로써 행렬 A(810)와 행렬 B(820)간의 곱셈 결과가 결과 행렬(830)로 나타나게 된다.

[0060] 제안된 하드웨어 가속기는 또한, 제어를 약간 수정하여 행렬 전치 연산을 수행할 수 있다. 전치될 타겟 행렬의 각각의 행은 행렬 A를 위한 행 버퍼(행렬 A - 행(207)에 로드될 수 있고, 행의 각각의 원소들은 전치를 위한 적

절한 위치에서 행렬 B 버퍼(206)로 전달될 수 있다. 이러한 연산은 전치될 타겟 행렬을 위한 전치 연산을 완수하기 위해 행렬의 전체 행을 위해 반복될 수 있다. 전치 연산이 완료되면 전치된 행렬의 모든 원소들이 행렬 B 버퍼(206)에 저장되기 때문에, 하드웨어 가속기는 전치된 행렬 B의 리로딩(reloading) 없이 행렬 A와 행렬 B 버퍼(206)에 저장된 행렬 A의 전치 행렬 A^T간의 행렬 곱셈을 수행할 수 있게 된다. 또한, 행렬 B의 전치 행렬 B^T를 먼저 구하여 행렬 B 버퍼(206)에 저장한 후, 행렬 A와 전치 행렬 B^T간의 행렬 곱셈을 수행할 수도 있다.

[0062] C. 제로(zero)를 위한 최적화

[0063] 하드웨어 가속기는 0(zero)의 값을 가질 수 있는 행렬 A의 원소를 처리하기 위해 특별한 특성을 가질 수 있다. 하드웨어 가속기는 행렬의 하나의 원소를 브로드캐스팅하고, 행렬 B의 행의 모든 원소와의 곱셈을 수행하기 때문에, 만약 행렬 A의 브로드캐스팅될 원소의 값이 0이라면 곱셈 연산과 누적 연산이 요구되지 않는다. 따라서, 다시 도 2를 참조하면, 제어 유닛은 행렬 A의 행이 DMA를 통해 SRAM으로부터 로딩될 때, 행렬 A의 행의 값을 확인(도 2의 제로 비교기(zero comparator, 211)를 통해)할 수 있다. 제로 비트 레지스터(zero bit register, 도 2의 제로 비트 확인 버퍼(zero bit check buffer, 212))가 행렬 A의 각각의 행 버퍼들을 위해 추가될 수 있고, 제로 비트 확인부(213)가 행 버퍼에 저장된 원소의 값이 0인지 아닌지를 표시하기 위해 이용될 수 있다. 제어 유닛은 이러한 제로 비트 레지스터를 이용하여 브로드캐스팅되어야 할 원소의 값이 0인 경우 브로드캐스팅을 하지 않는 것을 통해 연산(곱셈 연산 및 누적 연산)을 생략(skip)할 수 있다. 따라서 행렬이 포함하는 제로 값의 수에 비례하여 연산량이 감소될 수 있고, 연산에 요구되는 전력 소모를 줄일 수 있다.

[0065] 2. 평가

[0066] MacSim 시뮬레이터(Macsim, http://code.google.com/p/macsim/ 참조) 및 핀 트래이스 툴(Sion Berkowits, Tevi Devor, "Pin: Intel's Dynamic Binary Instrumentation Engine," CGO, Feb 2013. 참조)이 성능 평가를 위해 이용될 수 있다. 캐시 메모리를 갖지 않는 MCU의 클럭 주파수는 100MHz로 상정되었다. 제안된 하드웨어 가속기를 갖는 MCU는 아래 표 2와 같이 하드웨어 가속기를 갖지 않는 기준 구조보다 두 배의 성능 향상을 달성하였다. 전치와 제로값 원소 연산을 위한 특성 역시 추가의 성능 향상을 달성하였다.

표 2

[0067]

Types	*Execution time*	*Speedup*
Baseline	1.426	1
Matrix multiplication(MM)	0.703	2.03
MM + transpose	0.689	2.07
MM + transpose + zero opt	0.675	2.11

[0069] 3. 결론

[0070] 이처럼 본 발명의 실시예들에 따른 특화된 하드웨어 가속기를 포함하는 센서 허브 구조는 센서 퓨전 알고리즘을 효과적으로 처리할 수 있다. 성능 향상 결과는 제안된 하드웨어 가속기가 커다란 속도 향상을 달성할 수 있음을 보여준다. 이러한 하드웨어 가속기는 모션 탐지 및 상황 인식 어플리케이션들(context awareness applications)과 같은 다른 센서 데이터 처리 어플리케이션들에 적용하기 위해 확장될 수 있다.

[0072] 이와 같이, 본 발명의 실시예들에 따르면, 센서 허브 MCU(Micro Controller Unit)를 위한 하드웨어 가속기와 관련하여 방향 추정의 정확도를 향상시키고, 방향 추정을 위한 에너지 소비를 줄이기 위해 센서 퓨전을 위한 복합 칼만 필터(complex Kalman filter)를 처리할 수 있다. 또한, 더 확장된 프로그래머빌리티(programmability)를

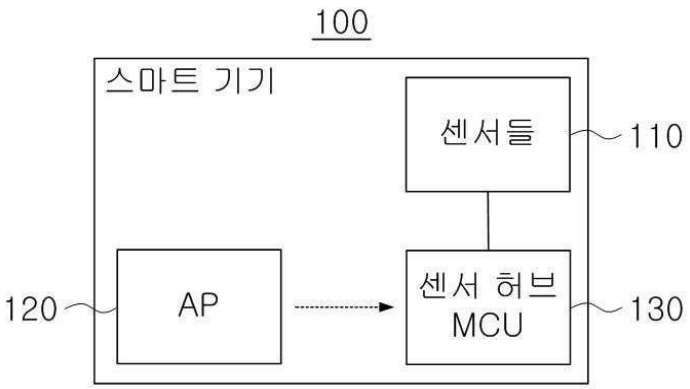
갖고, 칼만 필터 처리 시간에 대한 성능을 100% 이상 향상시킬 수 있다. 또한, 행렬을 메모리로부터 레지스터에 저장할 때, 해당 행의 원소가 0(zero)인지 여부를 저장하는 제로 비트 레지스터를 구비하여, 연산하고자 하는 원소가 0의 값을 가진 경우에는 해당 원소에 대한 연산을 생략(skip)함으로써, 연산 수행 시간 및 연산에 요구되는 전력 소모를 줄일 수 있다.

[0074] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

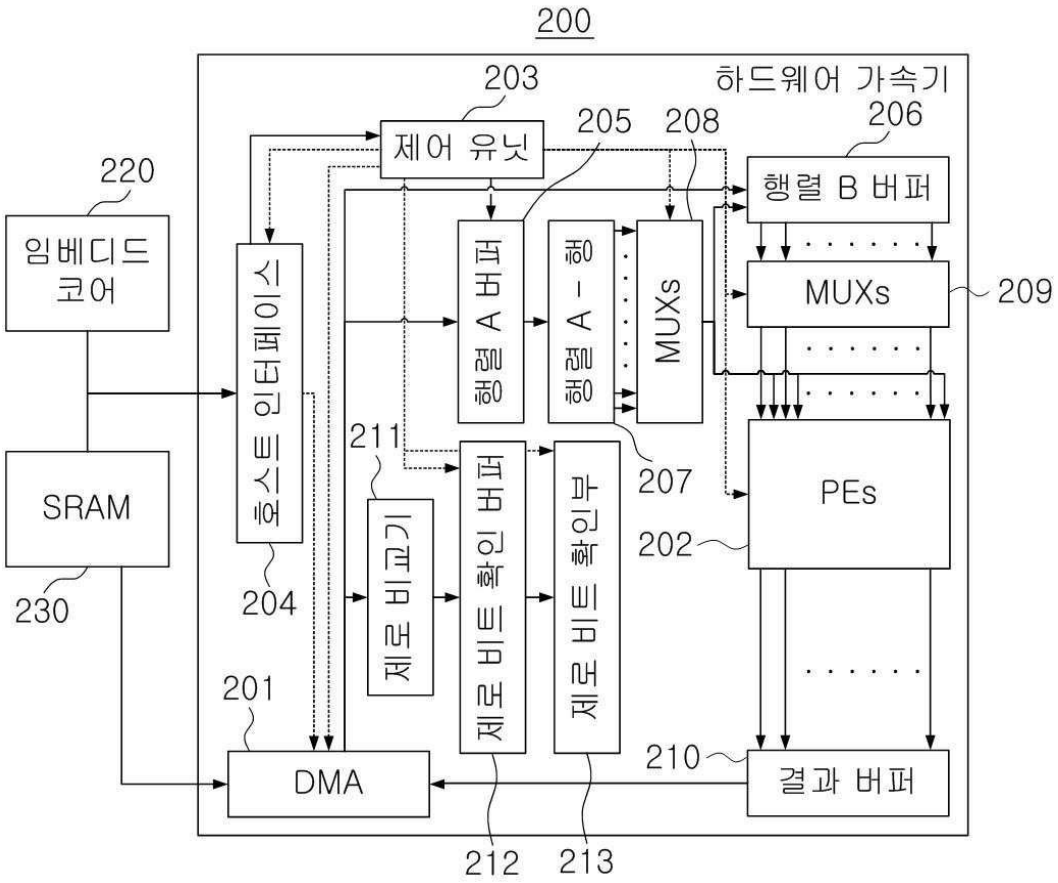
[0075] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

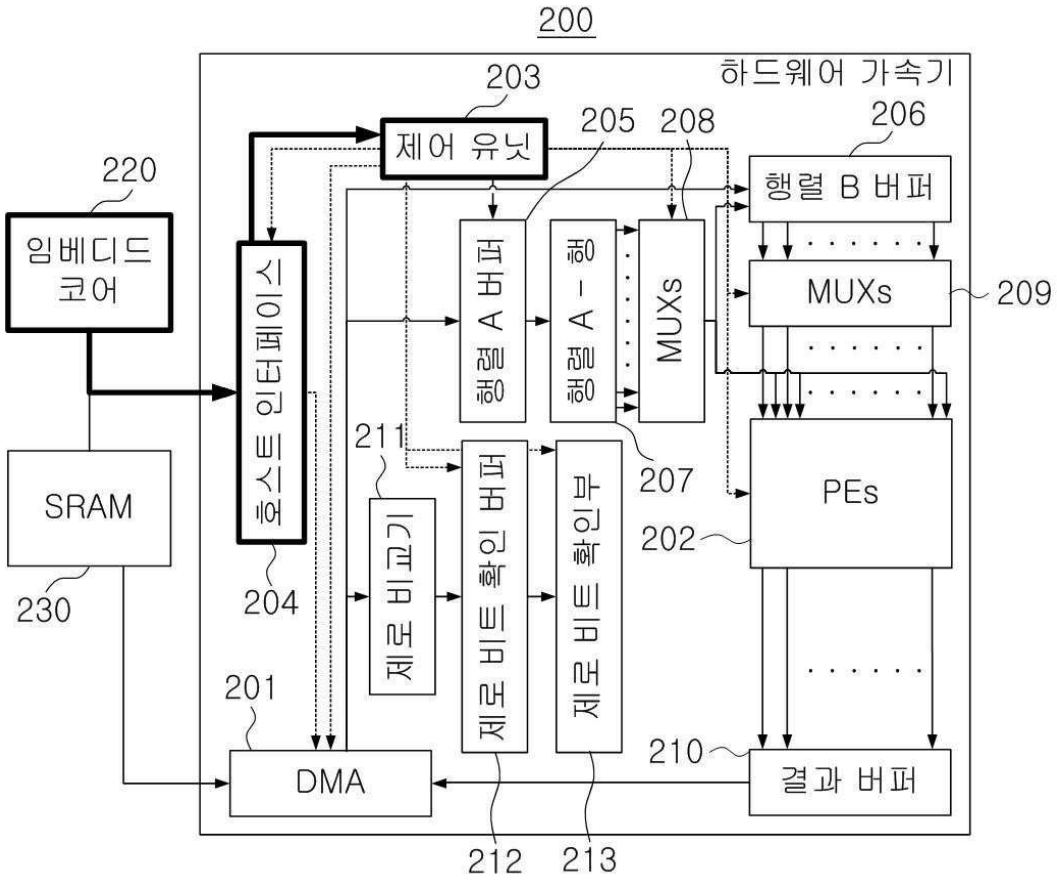
도면1



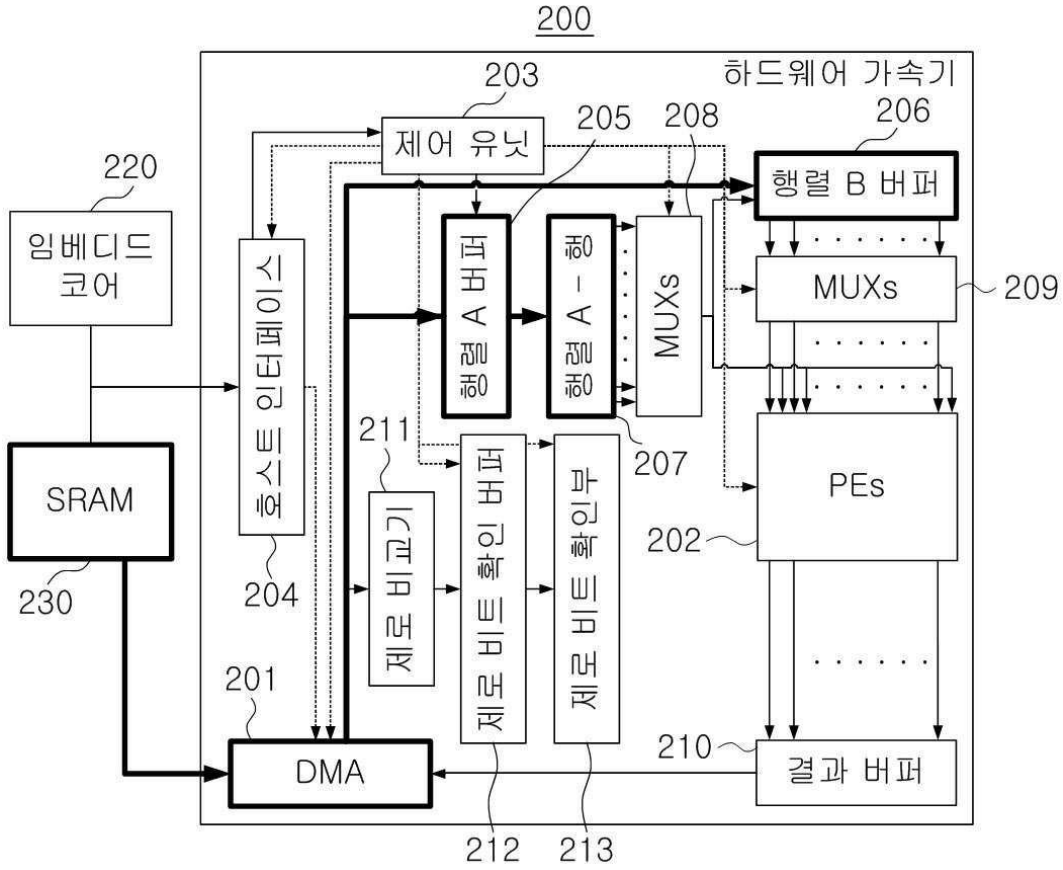
도면2


200
하드웨어 가속기
203
제어 유닛
205
208
206
행렬 B 버퍼
220
임베디드 코어
호스트 인터페이스
행렬 A 버퍼
행렬 A - 행
MUXs
MUXs
209
211
SRAM
제로 비교기
제로 비트 확인 버퍼
제로 비트 확인부
207
PEs
202
230
204
201
DMA
210
결과 버퍼
212
213

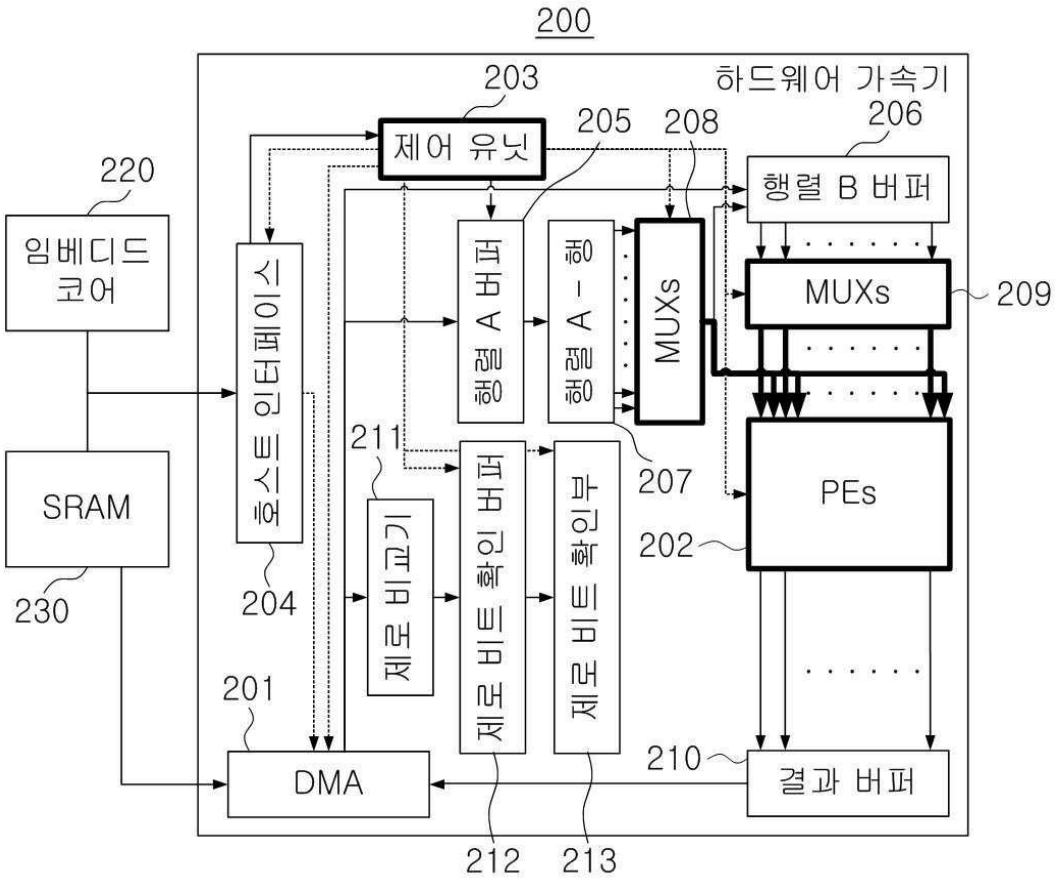
도면3


200
하드웨어 가속기
203
제어 유닛
205
208
206
행렬 B 버퍼
220
임베디드 코어
호스트 인터페이스
행렬 A 버퍼
행렬 A – 행
MUXs
MUXs
209
211
207
SRAM
PEs
202
230
204
제로 비트 비교기
제로 비트 확인 버퍼
제로 비트 확인부
201
210
DMA
결과 버퍼
212
213

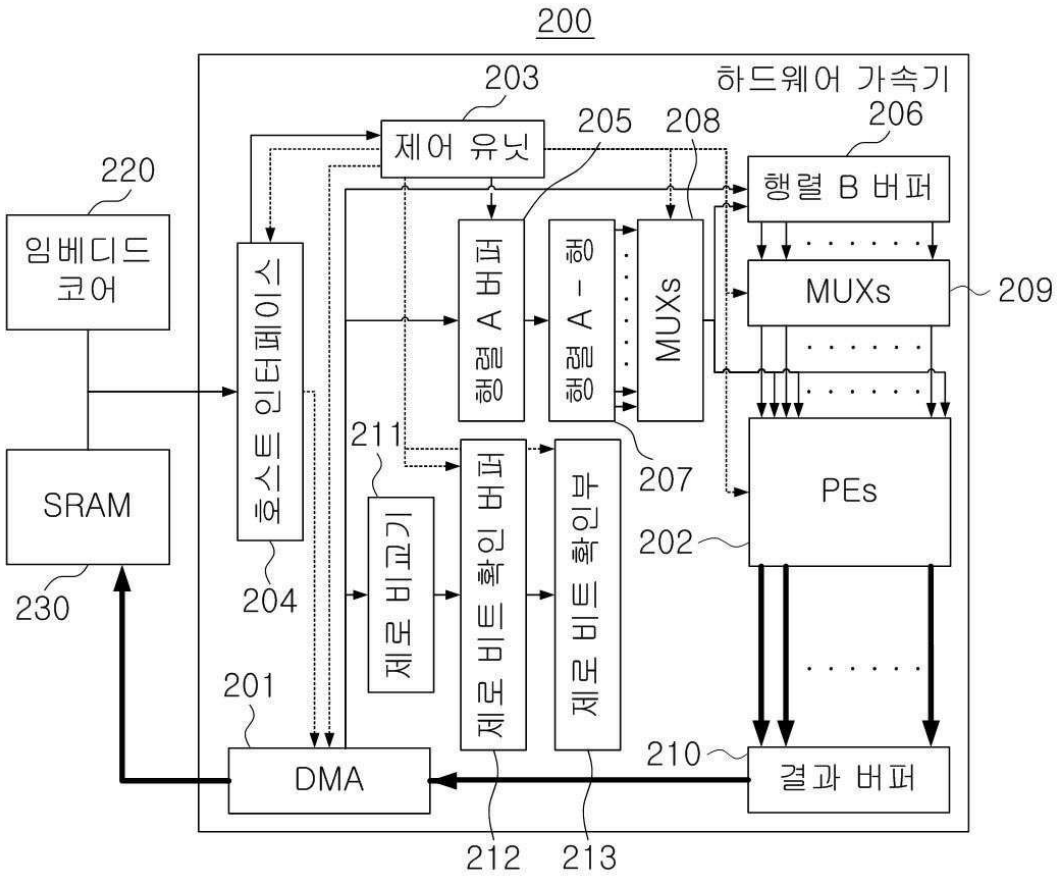
도면4


200
하드웨어 가속기
203
제어 유닛
205
208
206
행렬 B 버퍼
220
임베디드 코어
행렬 A 버퍼
행렬 A - 행
MUXs
MUXs
209
호스트 인터페이스
211
SRAM
207
PEs
202
제로 비트 확인 버퍼
제로 비트 확인부
제로 비교기
230
204
201
210
DMA
결과 버퍼
212
213

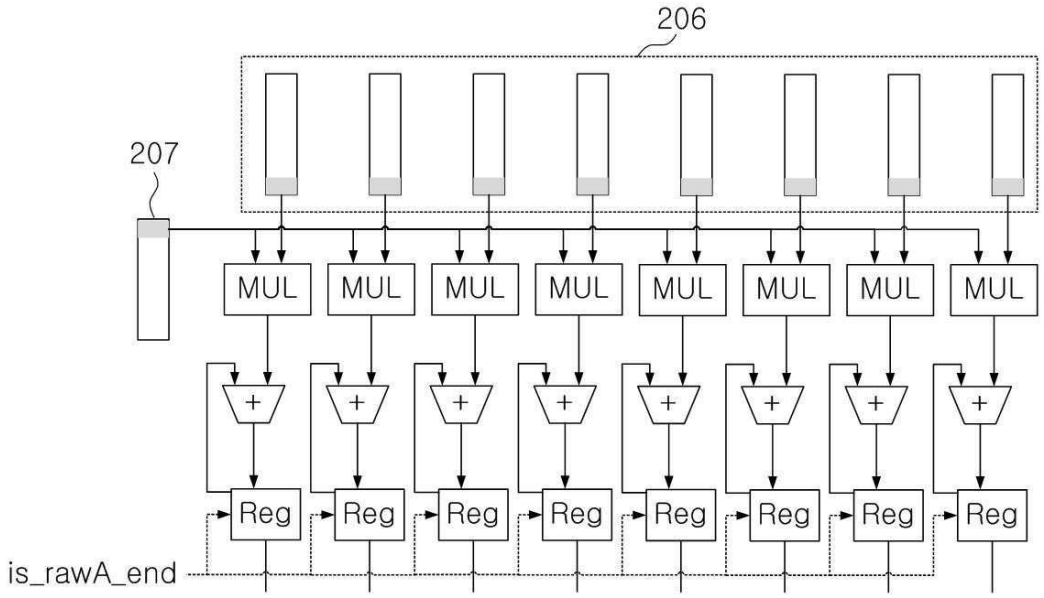
도면5


200
하드웨어 가속기
203
제어 유닛
205
208
206
행렬 B 버퍼
220
임베디드 코어
호스트 인터페이스
행렬 A 버퍼
행렬 A - 행
MUXs
MUXs
209
211
207
PEs
SRAM
202
230
204
제로 비교기
제로 비트 확인 버퍼
제로 비트 확인부
201
210
DMA
결과 버퍼
212
213

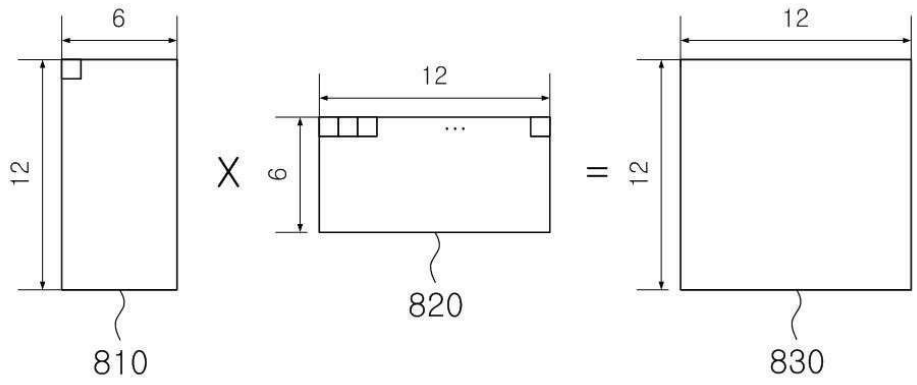
도면6


200
하드웨어 가속기
203
제어 유닛
205
208
206
행렬 B 버퍼
220
임베디드 코어
호스트 인터페이스
행렬 A 버퍼
행렬 A - 행
MUXs
MUXs
209
211
207
PEs
SRAM
제로 비교기
제로 비트 확인 버퍼
제로 비트 확인부
202
230
204
201
210
DMA
결과 버퍼
212
213

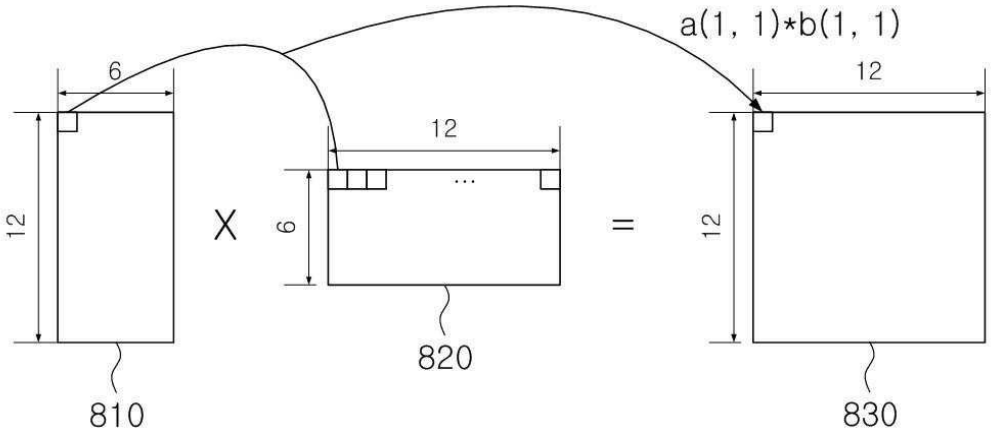
도면7


206
207
MUL
+
Reg
is_rawA_end

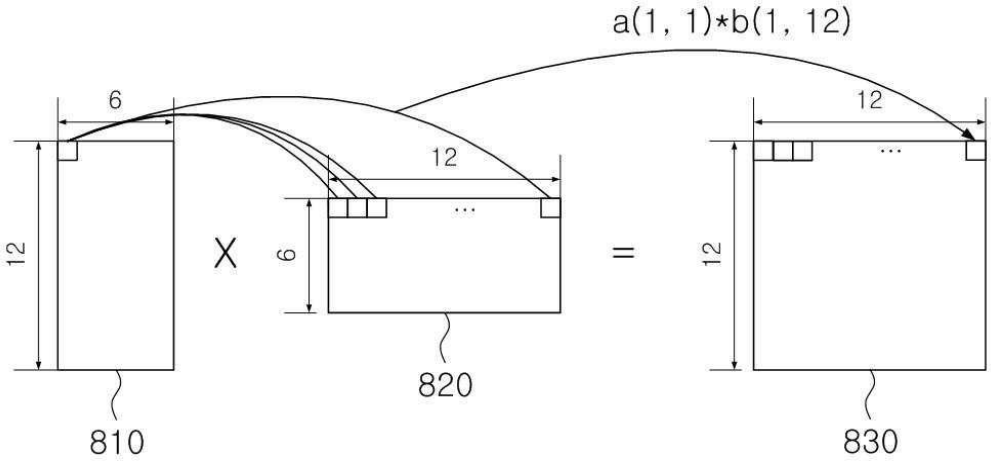
도면8


6
12
X
12
6
820
=
12
12
810
830

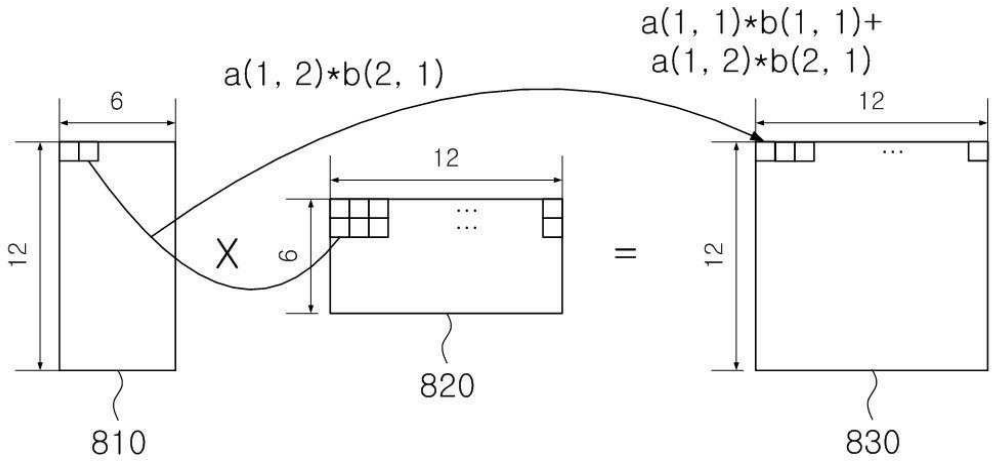
도면9


a(1, 1)*b(1, 1)
6
12
12
X
12
6
820
=
12
12
810
830

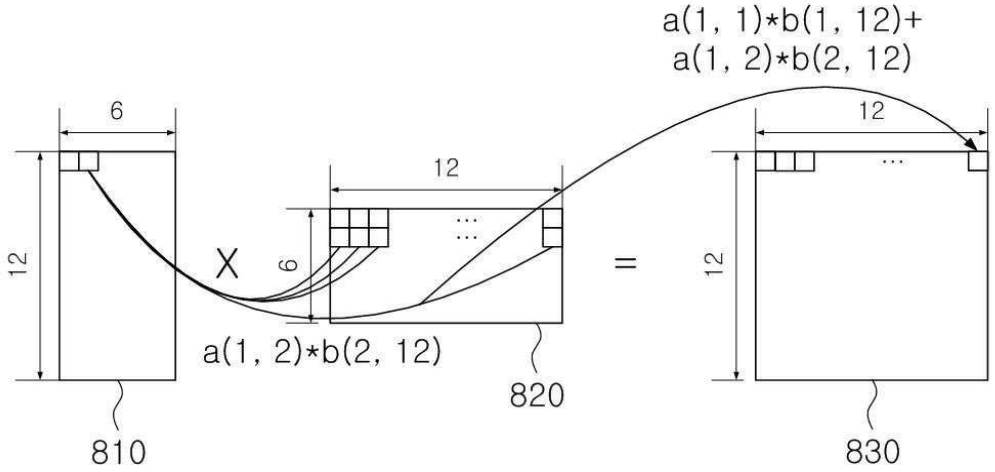
도면10


a(1, 1)*b(1, 12)
6
12
12
X
12
6
820
=
12
12
810
830

도면11



도면12



도면13

