



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0029530
(43) 공개일자 2025년03월05일

(51) 국제특허분류(Int. Cl.)

G06F 3/06 (2006.01)

(52) CPC특허분류

G06F 3/0656 (2013.01)

G06F 3/0658 (2013.01)

(21) 출원번호 10-2023-0110569

(22) 출원일자 2023년08월23일

심사청구일자 2023년08월23일

(71) 출원인

삼성전자주식회사

경기도 수원시 영통구 삼성로 129 (매탄동)

울산과학기술원

울산광역시 울주군 언양읍 유니스트길 50

(72) 발명자

이중은

울산광역시 울주군 언양읍 유니스트길 50

박재우

울산광역시 울주군 언양읍 유니스트길 50

이수길

울산광역시 울주군 언양읍 유니스트길 50

(74) 대리인

특허법인 무한

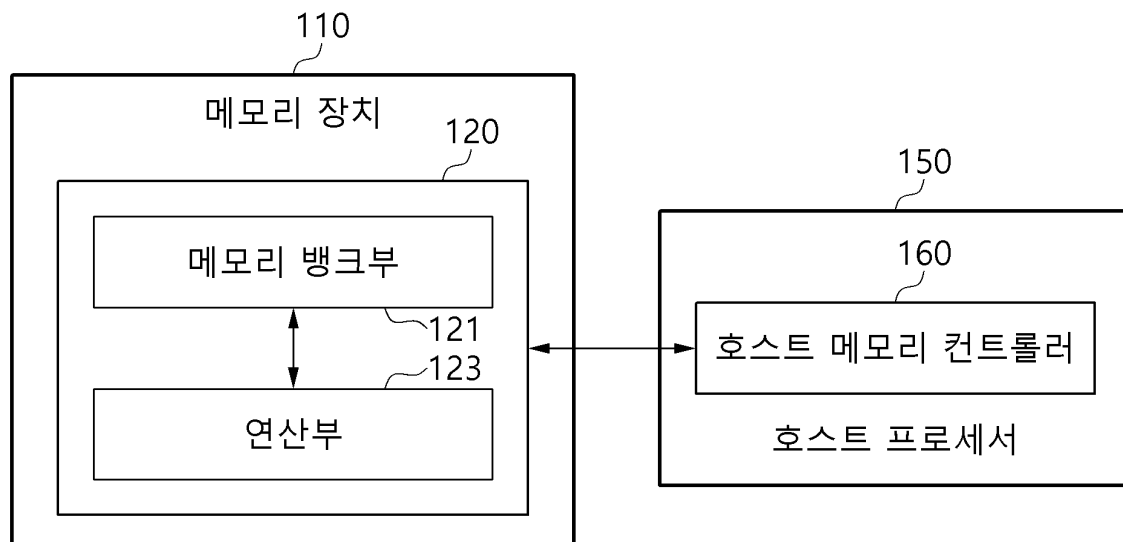
전체 청구항 수 : 총 21 항

(54) 발명의 명칭 메모리 장치 및 그 동작 방법

(57) 요약

메모리 장치 및 그 동작 방법이 개시된다. 일 실시예에 따른 메모리 장치는 메모리 셀 어레이, 복수의 스테이지들로 구성된 연산을 수행하는 로직 회로를 포함하는 연산기 및 상기 복수의 스테이지들 각각에 있어서, 호스트로부터 수신한 연산 커맨드에 대응하는 제1 데이터와 제2 데이터를 저장하는 복수의 버퍼들을 포함하고, 상기 연산기는 상기 제1 데이터 및 상기 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산(butterfly operation)을 수행하는 버터플라이 유닛을 포함할 수 있다.

대표도 - 도1



(52) CPC특허분류

G06F 3/0659 (2013.01)

G06F 2212/7203 (2013.01)

명세서

청구범위

청구항 1

메모리 셀 어레이;

복수의 스테이지들로 구성된 연산을 수행하는 로직 회로를 포함하는 연산기; 및

상기 복수의 스테이지들 각각에 있어서, 호스트로부터 수신한 연산 커맨드에 대응하는 제1 데이터와 제2 데이터를 저장하는 복수의 버퍼들

을 포함하고,

상기 연산기는

상기 제1 데이터 및 상기 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산(butterfly operation)을 수행하는 버터플라이 유닛

을 포함하는, 메모리 장치.

청구항 2

제1항에 있어서,

상기 연산기가 상기 복수의 스테이지들로 구성된 연산을 수행할 때, 상기 복수의 버퍼들은 상기 복수의 스테이지들로 구성된 연산의 입력 데이터 및 출력 데이터 중 적어도 하나를 저장할 수 있는, 메모리 장치.

청구항 3

제1항에 있어서,

상기 해당 스테이지 연산이 완료되면,

상기 제1 데이터를 저장하고 있는 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하고,

상기 제2 데이터를 저장하고 있는 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하는, 메모리 장치.

청구항 4

제1항에 있어서,

상기 연산 커맨드는

같은 버퍼 내의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 포함하는, 메모리 장치.

청구항 5

제4항에 있어서,

상기 호스트로부터 상기 제2 연산 커맨드를 수신할 경우,

상기 제1 데이터는 제1 버퍼에 저장되고, 상기 제2 데이터는 상기 제1 버퍼와 상이한 제2 버퍼에 저장되고,
상기 해당 스테이지 연산이 완료되면,
상기 제1 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하고,
상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는,
메모리 장치.

청구항 6

제4항에 있어서,
상기 호스트로부터 상기 제1 연산 커맨드를 수신할 경우,
상기 제1 데이터 및 상기 제2 데이터는 제2 버퍼에 저장되고,
상기 해당 스테이지 연산이 완료되면,
상기 제2 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하고,
상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는,
메모리 장치.

청구항 7

제1항에 있어서,
상기 복수의 스테이지들로 구성된 연산은
NTT(Number-Theoretic Transform) 연산 및 FFT(Fast Fourier Transform) 연산 중 적어도 하나를 포함하는, 메모리 장치.

청구항 8

제1항에 있어서,
상기 제1 데이터 및 상기 제2 데이터의 크기는
상기 복수의 버퍼들의 크기에 기초하여 결정되는, 메모리 장치.

청구항 9

제1항에 있어서,
행 버퍼(row buffer)를 더 포함하고,
상기 복수의 버퍼들은
상기 행 버퍼보다 더 작은 크기를 갖는, 메모리 장치.

청구항 10

제1항에 있어서,
상기 메모리 셀 어레이는

복수의 동적 랜덤 액세스 메모리(DRAM) 기반의 메모리 셀 어레이를 포함하는, 메모리 장치.

청구항 11

연산기를 포함하는 메모리 장치의 동작 방법에 있어서,
호스트로부터 커맨드를 수신하는 단계;
복수의 버퍼들 중에서, 상기 커맨드에 대응하는 제1 데이터 및 제2 데이터를 저장할 버퍼를 결정하는 단계;
상기 결정된 버퍼에 상기 제1 데이터 및 상기 제2 데이터를 저장하는 단계; 및
상기 제1 데이터 및 상기 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산을 수행하는 단계
를 포함하는 메모리 장치의 동작 방법.

청구항 12

제11항에 있어서,
상기 저장하는 단계는
상기 연산기가 상기 복수의 스테이지들로 구성된 연산을 수행할 때, 상기 복수의 버퍼들은 상기 복수의 스테이지들로 구성된 연산의 입력 데이터 및 출력 데이터 중 적어도 하나를 저장하는 단계
를 포함하는, 메모리 장치의 동작 방법.

청구항 13

제11항에 있어서,
상기 해당 스테이지 연산이 완료되면,
상기 제1 데이터를 저장하고 있는 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이가 연산 결과로 업데이트하는 단계; 및
상기 제2 데이터를 저장하고 있는 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이가 연산 결과로 업데이트하는 단계
를 더 포함하는, 메모리 장치의 동작 방법.

청구항 14

제11항에 있어서,
상기 호스트로부터 커맨드를 수신하는 단계는
같은 버퍼 내의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 수신하는 단계
를 포함하는, 메모리 장치의 동작 방법.

청구항 15

제11항에 있어서,
상기 호스트로부터 커맨드를 수신하는 단계는

상기 호스트로부터 상기 제2 연산 커맨드를 수신하는 단계

를 포함하고,

상기 저장하는 단계는

상기 제1 데이터를 제1 버퍼에 저장하는 단계; 및

상기 제2 데이터를 상기 제1 버퍼와 상이한 제2 버퍼에 저장하는 단계

를 포함하고,

상기 해당 스테이지 연산이 완료되면,

상기 제1 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계; 및

상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계

를 더 포함하는, 메모리 장치의 동작 방법.

청구항 16

제14항에 있어서,

상기 호스트로부터 커맨드를 수신하는 단계는

상기 호스트로부터 상기 제1 연산 커맨드를 수신하는 단계

를 포함하고,

상기 저장하는 단계는

상기 제1 데이터 및 상기 제2 데이터를 제2 버퍼에 저장하는 단계

를 포함하고,

상기 해당 스테이지 연산이 완료되면,

상기 제2 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계; 및

상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계

를 포함하는, 메모리 장치의 동작 방법.

청구항 17

제11항에 있어서,

상기 복수의 스테이지들로 구성된 연산은

NTT(Number-Theoretic Transform) 연산 및 FFT(Fast Fourier Transform) 연산 중 적어도 하나를 포함하는, 메모리 장치의 동작 방법.

청구항 18

제11항에 있어서,

상기 제1 데이터 및 상기 제2 데이터의 크기는

상기 복수의 버퍼들의 크기에 기초하여 결정되는, 메모리 장치의 동작 방법.

청구항 19

제11항에 있어서,

상기 메모리 장치는

행 버퍼(row buffer)를 더 포함하고,

상기 복수의 버퍼들은

상기 행 버퍼보다 더 작은 크기를 갖는, 메모리 장치의 동작 방법.

청구항 20

제11항에 있어서,

상기 메모리 셀 어레이는

복수의 동적 랜덤 액세스 메모리(DRAM) 기반의 메모리 셀 어레이를 포함하는, 메모리 장치의 동작 방법.

청구항 21

하드웨어와 결합되어 제11항 내지 제20항 중 어느 하나의 항의 방법을 실행시키기 위하여 매체에 저장된 컴퓨터 프로그램.

발명의 설명

기술 분야

[0001] 실시예들은 메모리 장치 및 그 동작 방법에 관한 것이다.

배경 기술

[0002] 차세대 양자내성암호(Post-Quantum Cryptography) 기법인 격자 기반 암호(Lattice-Based Cryptograph)와 동형 암호 기법들의 알고리즘에서 두 다항식의 곱셈은 매우 중요한 함수로서, 이를 효율적으로 처리하기 위해 NTT(Number-Theoretic Transform) 연산을 사용한다. 해당 NTT 연산을 SW로 구현할 경우 많은 시간이 걸리기 때문에 다양한 가속 기법이 연구된다.

[0003] NTT 연산을 가속할 수 있는 기존의 기법은 크게 두 가지로, GPU 기반의 가속 기법과 FPGA 기반의 NTT 가속 기법이 존재한다. GPU기반의 가속 방법은 가장 속도가 빠른 장점이 있으나, 전력 소비가 심한 문제가 있다. FPGA 기반의 가속 방법은 GPU 기반의 가속 방법과 비교하여 비슷한 처리 속도에 상대적으로 낮은 전력 소비를 가지고 있으나, 최대로 미리 정해진 길이 만큼의 NTT 연산을 처리할 수 있는 단점을 가지고 있다. 또한 대부분의 동형 암호 알고리즘은 보안 수준을 고려하여 높은 정밀도의 연산을 요구하는 반면, FPGA 는 고정밀도 연산에 효과적이지 않다.

[0004] 위에서 설명한 배경기술은 발명자가 본원의 개시 내용을 도출하는 과정에서 보유하거나 습득한 것으로서, 반드시 본 출원 전에 일반 공중에 공개된 공지기술이라고 할 수는 없다.

발명의 내용

과제의 해결 수단

- [0005] 일 실시예에 따른 메모리 장치는 메모리 셀 어레이; 복수의 스테이지들로 구성된 연산을 수행하는 로직 회로를 포함하는 연산기; 및 상기 복수의 스테이지들 각각에 있어서, 호스트로부터 수신한 연산 커맨드에 대응하는 제1 데이터와 제2 데이터를 저장하는 복수의 버퍼들을 포함하고, 상기 연산기는 상기 제1 데이터 및 상기 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산(butterfly operation)을 수행하는 버터플라이 유닛을 포함할 수 있다.
- [0006] 상기 연산기가 상기 복수의 스테이지들로 구성된 연산을 수행할 때, 상기 복수의 버퍼들은 상기 복수의 스테이지들로 구성된 연산의 입력 데이터 및 출력 데이터 중 적어도 하나를 저장할 수 있다.
- [0007] 상기 해당 스테이지 연산이 완료되면, 상기 제1 데이터를 저장하고 있는 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하고, 상기 제2 데이터를 저장하고 있는 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트할 수 있다.
- [0008] 상기 연산 커맨드는 같은 버퍼 내의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 포함할 수 있다.
- [0009] 상기 호스트로부터 상기 제2 연산 커맨드를 수신할 경우, 상기 제1 데이터는 제1 버퍼에 저장되고, 상기 제2 데이터는 상기 제1 버퍼와 상이한 제2 버퍼에 저장되고, 상기 해당 스테이지 연산이 완료되면, 상기 제1 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하고, 상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트할 수 있다.
- [0010] 상기 호스트로부터 상기 제1 연산 커맨드를 수신할 경우, 상기 제1 데이터 및 상기 제2 데이터는 제2 버퍼에 저장되고, 상기 해당 스테이지 연산이 완료되면, 상기 제2 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하고, 상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트할 수 있다.
- [0011] 상기 복수의 스테이지들로 구성된 연산은 NTT(Number-Theoretic Transform) 연산 및 FFT(Fast Fourier Transform) 연산 중 적어도 하나를 포함할 수 있다.
- [0012] 상기 제1 데이터 및 상기 제2 데이터의 크기는 상기 복수의 버퍼들의 크기에 기초하여 결정될 수 있다.
- [0013] 일 실시예에 따른 메모리 장치는 행 버퍼(row buffer)를 더 포함하고, 상기 복수의 버퍼들은 상기 행 버퍼보다 더 작은 크기를 갖을 수 있다.
- [0014] 상기 메모리 셀 어레이는 복수의 동적 랜덤 액세스 메모리(DRAM) 기반의 메모리 셀 어레이를 포함할 수 있다.
- [0015] 연산기를 포함하는 메모리 장치의 동작 방법에 있어서, 호스트로부터 커맨드를 수신하는 단계; 복수의 버퍼들 중에서, 상기 커맨드에 대응하는 제1 데이터 및 제2 데이터를 저장할 버퍼를 결정하는 단계; 상기 결정된 버퍼에 상기 제1 데이터 및 상기 제2 데이터를 저장하는 단계; 및 상기 제1 데이터 및 상기 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산을 수행하는 단계를 포함할 수 있다.
- [0016] 상기 저장하는 단계는 상기 연산기가 상기 복수의 스테이지들로 구성된 연산을 수행할 때, 상기 복수의 버퍼들은 상기 복수의 스테이지들로 구성된 연산의 입력 데이터 및 출력 데이터 중 적어도 하나를 저장하는 단계를 포함할 수 있다.
- [0017] 상기 해당 스테이지 연산이 완료되면, 일 실시예에 따른 메모리 장치의 동작 방법은 상기 제1 데이터를 저장하고 있는 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하는 단계; 및 상기 제2 데이터를 저장하고 있는 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하는 단계를 더 포함할 수 있다.
- [0018] 상기 호스트로부터 커맨드를 수신하는 단계는 같은 버퍼 내의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 수신하는 단계를 포함할 수 있다.
- [0019] 상기 호스트로부터 커맨드를 수신하는 단계는 상기 호스트로부터 상기 제2 연산 커맨드를 수신하는 단계를 포함하고, 상기 저장하는 단계는 상기 제1 데이터를 제1 버퍼에 저장하는 단계; 및 상기 제2 데이터를 상기 제1 버퍼와 상이한 제2 버퍼에 저장하는 단계를 포함하고, 상기 해당 스테이지 연산이 완료되면, 일 실시예에 따른 메모리 장치의 동작 방법은 상기 제1 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라이 연산 결과로 업데이트하는 단계; 및 상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터

플라가 연산 결과로 업데이트하는 단계를 더 포함할 수 있다.

- [0020] 상기 호스트로부터 커맨드를 수신하는 단계는 상기 호스트로부터 상기 제1 연산 커맨드를 수신하는 단계를 포함하고, 상기 저장하는 단계는 상기 제1 데이터 및 상기 제2 데이터를 제2 버퍼에 저장하는 단계를 포함하고, 상기 해당 스테이지 연산이 완료되면, 상기 제2 버퍼는 상기 제1 데이터를, 상기 제1 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계; 및 상기 제2 버퍼는 상기 제2 데이터를, 상기 제2 데이터에 대응하는 상기 버터플라가 연산 결과로 업데이트하는 단계를 포함할 수 있다.
- [0021] 상기 복수의 스테이지들로 구성된 연산은 NTT(Number-Theoretic Transform) 연산 및 FFT(Fast Fourier Transform) 연산 중 적어도 하나를 포함할 수 있다.
- [0022] 상기 제1 데이터 및 상기 제2 데이터의 크기는 상기 복수의 버퍼들의 크기에 기초하여 결정될 수 있다.
- [0023] 상기 메모리 장치는 행 버퍼(row buffer)를 더 포함하고, 상기 복수의 버퍼들은 상기 행 버퍼보다 더 작은 크기를 갖을 수 있다.
- [0024] 상기 메모리 셀 어레이는 복수의 동적 랜덤 액세스 메모리(DRAM) 기반의 메모리 셀 어레이를 포함할 수 있다.

도면의 간단한 설명

- [0025] 도 1은 일 실시예에 따른 메모리 장치의 블록도를 도시한 도면이다.
- 도 2는 일 실시예에 따른 NTT 연산을 설명하기 위한 도면이다.
- 도 3a는 일 실시예에 따른 메모리 장치의 개략적인 블록도이다.
- 도 3b는 일 실시예에 따른 연산부의 개략적인 블록도이다.
- 도 4는 일 실시예에 따른 인 플레이스 업데이트의 예시를 도시한 도면이다.
- 도 5는 일 실시예에 따른 파이프라이닝 방법을 설명하기 위한 도면이다.
- 도 6은 일 실시예에 따른 메모리 장치의 동작 방법을 설명하기 위한 순서도이다.

발명을 실시하기 위한 구체적인 내용

- [0026] 본 명세서에서 개시되어 있는 특정한 구조적 또는 기능적 설명들은 단지 기술적 개념에 따른 실시예들을 설명하기 위한 목적으로 예시된 것으로서, 실제로 구현된 형태는 다양한 다른 모습을 가질 수 있으며 본 명세서에 설명된 실시예로만 한정되지 않는다.
- [0027] 제1 또는 제2 등의 용어는 다양한 구성요소들을 설명하는데 사용될 수 있지만, 이런 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 이해되어야 한다. 예를 들어 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소는 제1 구성요소로도 명명될 수 있다.
- [0028] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성요소가 다른 구성요소에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는, 중간에 다른 구성요소가 존재하지 않는 것으로 이해되어야 할 것이다. 구성요소들 간의 관계를 설명하는 표현들, 예를 들어 "~간의"와 "바로~간의" 또는 "~에 이웃하는"과 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.
- [0029] 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 실시된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0030] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 해당 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.

- [0031] 실시예들은 퍼스널 컴퓨터, 랩톱 컴퓨터, 태블릿 컴퓨터, 스마트 폰, 텔레비전, 스마트 가전 기기, 지능형 자동차, 키오스크, 웨어러블 장치 등 다양한 형태의 제품으로 구현될 수 있다. 이하, 실시예들을 첨부된 도면을 참조하여 상세하게 설명한다. 각 도면에 제시된 동일한 참조 부호는 동일한 부재를 나타낸다.
- [0032] 도 1은 일 실시예에 따른 메모리 장치의 블록도를 도시한 도면이다.
- [0033] 도 1을 참조하면, 도 1에 도시된 바와 같이, 본 개시의 실시예는 PIM(PIM: Processing In Memory) 메모리 시스템을 제공한다. 메모리 시스템은 메모리 장치(110)에 통합될 추가적인 컴퓨팅 리소스를 지원한다. 메모리 시스템은 그래픽 처리 장치(GPU) 또는 중앙 처리 장치(CPU)와 같은 호스트 프로세서(150)에 연결된 메모리 장치(110)를 포함한다. 메모리 장치(110)는 다항식의 곱셈 연산을 가속할 수 있고, 이를 이용한 양자 내성 암호(예를 들어, 기법인 격자 기반 암호, 동형암호 기법 등), AI(Artificial Intelligence) 및 HPC(High-Performance Computing) 응용에 대해 적용이 가능하고, 메모리 시스템은 서버 및 휴대기기에 적용될 수 있다.
- [0034] 일 실시예에서, 메모리 장치(110)는 메모리 다이(120)를 포함한다. 호스트 프로세서(150)는 메모리 장치(110)와 인터페이스하기 위한 호스트 메모리 컨트롤러(160)(또는, 호스트 컨트롤러)를 포함할 수 있다. 그러나 본 개시가 이에 한정되는 것은 아니다. 예를 들어, 호스트 메모리 컨트롤러(160)는 호스트 프로세서(150)와 분리될 수도 있다(예를 들어, 호스트 프로세서(160)와 별도의 다이 또는 동일한 다이로서). 또는, 메모리 장치(110)는 메모리 컨트롤러를 포함할 수 있고, 메모리 다이(120)는 내부 메모리 버스를 통해 메모리 컨트롤러와 연결될 수 있다.
- [0035] 호스트 메모리 컨트롤러(160)는 호스트 프로세서(150)로부터의 명령들의 실행을 조정하도록 구성될 수 있다. 명령어는 정규 커맨드(또는, 명령어)와 PIM 커맨드를 모두 포함할 수 있다. 예를 들어, 정규 커맨드(예를 들어, 메모리 내 기능 명령들이 아닌 전통적인 로드(읽기), 저장(쓰기), 활성화(ACT) 기능들)는 호스트 메모리 컨트롤러(160)에 의해 전송되고, 통상적인 방식으로 실행될 수 있다. 예를 들어, 정규 명령은 외부 버스를 통해 수신된 데이터를 메모리 다이(120)에 저장하는 명령 및 메모리 다이(120)로부터 데이터를 검색하고 외부 버스를 통해 호스트 프로세서(150)로 데이터를 전송하는 명령을 포함할 수 있다. 아래에서 상세히 설명하겠지만, PIM 연산을 위해 PIM 커맨드가 추가될 수 있다. PIM 커맨드는 같은 버퍼 내의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 포함할 수 있다. 아래에서, 제1 연산 커맨드는 C1으로, 제2 연산 커맨드는 C2로 지칭될 수 있다.
- [0036] 본 개시의 실시예의 양태는 인-메모리 컴퓨팅(IMC; In Memory Computing)의 사용에 관한 것으로, 일 실시예에 따른 메모리 시스템은 메모리 다이(120) 내부의 메모리 뱅크부(121)와 연산부(123)를 포함할 수 있다. 몇몇 비교되는 메모리 시스템(예: PNM(process near memory))은 메모리 다이 외부의 연산기(예: 산술 논리 유닛(ALU))를 포함하고, 연산기가 외부 버스를 통과하지 않고, 메모리 다이에 저장된 데이터에 대한 연산(operation)(예를 들어, 산술 연산(arithmetic operations))을 수행할 수 있도록 메모리 다이의 메모리 뱅크에 의해 공유된다.
- [0037] 본 개시의 실시예의 일부 양태는 연산부(123)를 메모리 다이(120)의 메모리 뱅크부(121)에 통합함으로써 메모리 경계 연산을 가속시키는 것에 관한 것이다. 예를 들어, 연산부(123)는 데이터를 보유한 메모리 뱅크부(121)와 동일한 물리적 반도체 다이에 있을 수 있다.
- [0038] 메모리 뱅크부(121)는 행(row) 및 열(column)(또는 페이지 및 열)로 배열된 메모리 셀의 어레이를 포함할 수 있다. 예를 들어, 메모리 뱅크부(121)는 n 개의 행(또는 페이지) 및 m 개의 열로 배열된 DRAM 셀을 포함할 수 있다(여기서, n 과 m 은 자연수). 복수의 비트 라인들(예: B_1 내지 B_m)(bitlines)은 열 방향을 따라 연장되고, 복수의 행 인에이블 라인들(예: R_1 내지 R_n)(row enable lines)은 어레이의 행 방향을 따라 연장되고 비트 라인들을 가로 지를 수 있다. 각 비트 라인은 해당 열의 모든 셀에 연결될 수 있다(예를 들어, 배열의 i 번째 열에 있는 모든 셀은 비트 라인 B_i 에 연결됨). 마찬가지로, 각 행 인에이블 라인(예: R_1 내지 R_n)은 대응하는 행의 각 메모리 셀에 연결될 수 있다(예: 어레이의 j 번째 행 또는 페이지의 모든 셀은 행 인에이블 라인(R_j)에 연결됨). 메모리 뱅크부(121)의 행의 메모리 셀은 또한 메모리 페이지라고 지칭될 수 있다.
- [0039] 연산부(123)는 양자 내성 암호(예를 들어, 기법인 격자 기반 암호, 동형암호 기법 등)에서 사용되는 두 다항식의 곱셈 연산을 효율적으로 처리하기 위한 연산(예를 들어, NTT 연산 또는 FFT(fast fourier transform) 연산)을 수행할 수 있다.

[0040] NTT 연산은 유한체 환(Ring)안에 간혀 동작하는 FFT 변환식이다. 환의 조건은 $x^n = -1$ 이며, n은 NTT에서 연산되는 다항식의 차수이다. NTT 연산을 기반으로 하는 다항식 곱셈기는 FFT 이론과 컨볼루션 이론(Convolution Theory)으로 구성되며 계산 복잡도를 줄임과 동시에 다항식 계산까지 수행할 수 있다. 이때, NTT가 환에서 동작을 하기 위해 모듈러 감소(Modular Reduction)를 필요로 한다. 이하에서, 연산부(123)가 NTT 연산을 수행하는 동작을 예시로 설명하나, 연산부(123)에서 수행 가능한 연산이 NTT 연산으로 한정되는 것은 아니다.

[0041] NTT 연산은 메모리 접근이 상당히 불규칙하고, 아주 많은 양의 데이터를 사용하기 때문에, 메모리 접근을 효율적으로 할 수 있도록 해주는 것이 성능 향상에 매우 중요할 수 있다. 그러나, DRAM은 메모리 셀 어레이의 하나의 행(row) 내에서는 데이터 재사용이 매우 자유로운 반면에, 다른 행을 접근하거나 또는 다른 뱅크로의 데이터 전달은 그다지 효율적이지 않을 수 있다. 일 실시예에 따른 메모리 뱅크부(121)는 복수의 버퍼들을 사용하여 행 및 뱅크 내에서의 데이터 재사용을 극대화할 수 있다.

[0042] 도 2는 일 실시예에 따른 NTT 연산을 설명하기 위한 도면이다.

[0043] 도 2를 참조하면, 도면(210)은 쿨리-투키 알고리즘을 사용하여 8-포인트 NTT 연산을 수행할 때의 데이터 플로우를 도시한 도면이다.

[0044] NTT 연산은 다음과 같이 정의된다. 다항식 $A(x)$ 는 $R_q(\text{ring})$ 안에 단혀 있으며, 구성원 a 는 길이가 n 벡터로 이루어져 있다. 따라서 벡터(Vector) $\mathbf{a} = (a_0, \dots, a_{n-1})$ 다항식 $A(x)$ 의 계수이다. $\text{NTT}(\mathbf{a})$ 는 수학적 1과 같이 정의된다.

수학적 1

$$A_i = \sum_{j=0}^{n-1} a_j w_n^{ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1$$

[0046] 개별적으로 진행된 두 입력 다항식 A와 B에 대한 NTT의 결과는 점 곱셈(Point-wise Multiplication)을 사용하여 같은 차수의 계수들끼리 점 곱셈을 수행한다. 점 곱셈 수행 결과는 다시 INTT 연산을 진행하여 최종적으로 두 다항식 A와 B에 대한 곱셈 결과를 얻는다. 역변환인 $\text{INTT}^{-1}(A)$ 는 수학적 2와 같이 정의된다.

수학적 2

$$a_i = n^{-1} \sum_{j=0}^{n-1} A_j w_n^{-ij} \bmod q, \text{ for } i = 0, 1, \dots, n-1$$

[0048] 수학적 2에 사용된 숫자 n 은 1의 n 제곱근(root of unity)이다. 즉 $w^n \equiv 1 \bmod q$ 의 식을 만족해야 한다. 또한 역변환인 INTT 에 사용하는 변수 n 과 w 는 $n \times n^{-1} \equiv 1 \bmod q$, $w \times w^{-1} \equiv 1 \bmod q$ 의 조건을 만족하는 숫자로 구성된다. NTT는 FFT 알고리즘 구조를 사용할 수 있다.

[0049] NTT 연산은 복수의 스테이지들로 구성될 수 있다. 예를 들어, NTT 연산은 쿨리-투키(Cooley-Tukey) 알고리즘을 통해 수행될 수 있다. 쿨리-투키 알고리즘 방식으로 크기 N 의 FFT 또는 NTT 변환을 하기 위해서는 $\log N$ 스테이지가 필요하며, 각 스테이지마다 버터플라이(butterfly) 연산이 필요할 수 있다. 도면(220)을 참조하면, 쿨리-투키 알고리즘의 버터플라이 연산은 모듈러 덧셈(ModAdd) 연산, 모듈러 뺄셈(ModSub) 연산 및 모듈러 곱셈(ModMult) 연산으로 구성될 수 있다.

[0050] 다만, NTT 연산을 수행하는 방법은 쿨리-투키 알고리즘으로 한정되지 않는다. 예를 들어, NTT 연산은 젠틀맨-산데(Gentleman-Sande) 알고리즘을 통해 수행될 수도 있고, 젠틀맨-산데 알고리즘을 구성하는 버터플라이 연산

은 콜리-투키 알고리즘의 버터플라이 연산과 상이할 수 있다.

- [0051] 도 3a는 일 실시예에 따른 메모리 장치의 개략적인 블록도이다. 도 1 내지 도 2를 참조하여 설명한 내용은 도 3a에 동일하게 적용될 수 있고, 중복되는 내용은 생략될 수 있다.
- [0052] 도 3a를 참조하면, 일 실시예에 따른 메모리 장치는 메모리 셀 어레이(310), 행 버퍼(320), 선택부(330), 제1 아톰 버퍼(340), 제2 아톰 버퍼(350) 및 연산부(360)를 포함할 수 있다. 이하 사용되는 '. 부', '. 기' 등의 용어는 기능이나 동작을 처리하는 단위를 의미하며, 이는 하드웨어나 소프트웨어, 또는, 하드웨어 및 소프트웨어의 결합으로 구현될 수 있다.
- [0053] 셀 어레이(310)의 특정 행으로부터 데이터를 읽을 때, 해당 셀의 커패시터에 저장된 전압들에 대응하는 전압들은 비트 라인들을 따라 전송되고 행 버퍼(320)에 의해 읽혀질 수 있다. 행 버퍼(320)는 비트 라인들 중 대응하는 하나에 연결될 수 있다. 행 버퍼(320)는 "프리 치 차지(precharge)"명령에 의해 소거될 때까지 현재 행(또는 페이지)으로부터 읽혀진 데이터를 저장할 수 있다.
- [0054] 선택부(330)는 행 버퍼(320)에 저장된 데이터 중에서, 열 주소에 대응하는 데이터를 선택하는 장치로, 예를 들어, 멀티플렉서일 수 있다. 그러나, 선택기는 멀티플렉서로 한정되는 것은 아니며, 복수의 데이터들 중 특정 데이터를 선택할 수 있는 다양한 장치들을 포함할 수 있다.
- [0055] 메모리 장치는 복수의 아톰 버퍼(예를 들어, 제1 아톰 버퍼(340)와 제2 아톰 버퍼(350))를 포함할 수 있다. 아톰 버퍼란 효율적인 연산 및 메모리 접근을 위해 사용할 수 있는 행 버퍼(320) 보다 작은 크기의 버퍼를 의미할 수 있다. 여기서 아톰(atom)은 행(row)보다 작은 단위의 크기로, 예를 들어 32 Byte일 수 있다.
- [0056] 제1 아톰 버퍼(340)는 기존 DRAM 메모리 장치에 포함된 버퍼일 수 있다. 예를 들어, 제1 아톰 버퍼(340)는 글로벌 I/O(Global I/O)를 위해 사용하는 GSA(Global Sense Amplifier)일 수 있다. 제1 아톰 버퍼(340)는 추가 버퍼인 제2 아톰 버퍼(350)와 구분하기 위하여 프라이머리 아톰 버퍼(primary atom buffer)라고 지칭할 수 있다.
- [0057] 제2 아톰 버퍼(350)는 세컨더리 아톰 버퍼로 지칭될 수 있고, 제1 아톰 버퍼(340)(예를 들어, GSA) 이외에 추가로 사용하는 하나 이상의 아톰 버퍼일 수 있다.
- [0058] 기존에 메모리 뱅크당 행 버퍼가 하나만 있는 구조에서도 버터플라이 연산이 가능하기는 하지만, 메모리 뱅크 접근 효율이 매우 나쁠 수 있다. 특히, NTT 연산은 다른 행에 있는 데이터 사이의 연산이 매우 많기 때문에, 행 버퍼만을 이용하여 NTT 연산을 수행하기에는 메모리 접근 효율이 매우 떨어질 수 있다.
- [0059] 이에, 일 실시예에 따른 메모리 장치는 행 버퍼보다 훨씬 작은 크기의 제2 아톰 버퍼(350)를 추가하여 전술한 문제를 해결할 수 있다. 제2 아톰 버퍼(350)를 추가로 사용할 경우, 1)연산 결과를 해당 버퍼에 덮어쓰는 인플레이스 업데이트(in-place update)가 가능하고, 2)파이프라이닝(pipelining)을 통해 병렬 연산 처리가 가능할 수 있다.
- [0060] 1)인 플레이스 업데이트에 대하여 설명하면, 각 데이터는 각 스테이지 당 두 번씩 사용되기 때문에 원래는 입력 데이터를 저장할 버퍼 뿐만 아니라 출력 데이터를 저장할 버퍼도 필요할 수 있다. 하지만 스테이지 단위(스테이지 단위는 버터플라이 연산 단위와 동일)로 스케줄을 하게 되면, 각 버터플라이 연산에 사용된 데이터는 해당 스테이지 내에서는 다시 사용될 일이 없으므로 입력 데이터를 삭제해도 무방하다. 다시 말해, 버터플라이 연산의 결과를 버터플라이 연산의 입력에 사용한 버퍼에 다시 저장하는, 즉 입력을 덮어쓰는 식으로 구현해도 데이터 재사용을 충분히 할 수 있다.
- [0061] 버터플라이 연산은 2개의 인풋과 2개의 아웃풋을 가지며, 인 플레이스 업데이트를 위해서는 2개의 인풋을 두 버퍼에 나누어 저장해 둘 필요가 있다. 만일 버퍼가 하나 뿐이라면 한 번의 버터플라이 연산에서 하나의 아웃풋만을 덮어쓸 수 있고, 다른 하나의 아웃풋을 계산하기 위해 다시 한 번의 버터플라이 연산이 필요하나, 이때 한 인풋은 이미 이전 연산의 아웃풋으로 덮여져있었기 때문에 두 번째의 버터플라이 연산을 수행할 수 없다. 인플레이스 업데이트를 하지 않는 경우에는 아웃풋을 저장하기 위해 또다른 행 액티베이션(row activation)이 요구되어 메모리 접근 효율이 매우 나빠지게 된다.
- [0062] 반면에, 제2 아톰 버퍼(350)를 사용하면 인트라 아톰 케이스(두 인풋 데이터가 같은 아톰 버퍼상에 존재하는 케이스)이 아닌 인트라 행 케이스(두 인풋 데이터가 같은 행 버퍼(320) 상에 존재하는 케이스) 또는 인터 로 케이스(두 인풋 데이터가 다른 행 버퍼(320) 상에 존재하는 케이스)에서도 버터플라이 연산의 인 플레이스 업데이트

가 문제없이 가능하다.

- [0063] 2)파이프라이닝을 통해 병렬 연산 처리에 대해 설명하면, 아톰 버퍼 또는 행 버퍼의 크기보다 더 큰 크기의 입력 단위를 처리하기 위해 복수의 제2 아톰 버퍼(350)를 사용하는 기법이다. 파이프라이닝은 기본적으로 연산에 걸리는 시간과 메모리 접근에 걸리는 시간을 서로 겹치게 함으로써 전체 실행시간을 단축할 수 있다. 다시 말해, 파이프라이닝을 통해 연산과 메모리 읽기/쓰기를 동시에 진행할 수 있다. 나아가, 파이프라이닝을 통해 인터 로 케이스의 경우에는, 로 액티베이션의 개수를 줄임으로써 추가적인 실행시간 단축이 가능하다. 파이프라이닝을 통한 병렬 연산 처리 방법은 아래에서 도 5를 참조하여 상세히 설명한다.
- [0064] 도 3b는 일 실시예에 따른 연산부의 개략적인 블록도이다.
- [0065] 도 3b를 참조하면, 일 실시예에 따른 연산부(360)는 TFG(twiddle factor generator)(361), 버터플라이 유닛(BU; butterfly unit)(363) 및 LSU(Load Store Unit)(365)를 포함할 수 있다.
- [0066] TFG(361)은 NTT 연산 또는 FFT 연산과 관련된 알고리즘에서 사용되는 회전 개념의 복소수 계수인 트위들 팩터를 생성하는 장치로, TFG(361)는 버터플라이 유닛(363)에서 버터플라이 연산을 수행할 수 있도록 트위들 팩터를 전달할 수 있다.
- [0067] LSU(365)는 메모리와 데이터 간의 상호 작용을 관리하는 장치로, 버터플라이 유닛(363)이 아톰 버퍼(제1 아톰 버퍼(340), 제2 아톰 버퍼(350))에 접근할 수 있도록 할 수 있다.
- [0068] 버터플라이 유닛(363)은 전술한 버터플라이 연산을 수행하기 위한 장치로, 모듈러 덧셈 연산기, 모듈러 뺄셈 연산기 및 모듈러 곱셈 연산기를 포함할 수 있다. 그러나, 버터플라이 유닛(363)의 구성은 전술한 사항으로 한정되지 않는다.
- [0069] 도 4는 일 실시예에 따른 인 플레이스 업데이트의 예시를 도시한 도면이다.
- [0070] 도 4를 참조하면, 도면(410)은 인트라 아톰 케이스(두 인풋 데이터가 같은 아톰 버퍼상에 존재하는 케이스)에서의 인 플레이스 업데이트 동작을, 도면(420)은 인트라 로 케이스(두 인풋 데이터가 같은 행 버퍼 상에 존재하는 케이스)에서의 인 플레이스 업데이트 동작을, 도면(430)은 인터 로 케이스(두 인풋 데이터가 다른 행 버퍼 상에 존재하는 케이스)에서의 인 플레이스 업데이트 동작의 예를 도시한 도면으로, RD, WR, PR/ACT 모두 기존의 DRAM 명령어인 로드(읽기), 저장(쓰기), 프리차지/활성화(PR/ACT)를 의미하고, C1, C2는 PIM 커맨드인 제1 연산 커맨드와 제2 연산 커맨드를 의미할 수 있다.
- [0071] 일 실시예에 따른 메모리 장치는 큰 데이터의 NTT를 세가지 단계로 구분하여 전체 연산을 작은 dram atom 크기의 연산으로 쪼개어 수행할 수 있다. 메모리 장치는 1)기존의 DRAM 명령어를 사용하여 아톰 버퍼에 연산에 필요한 DRAM 아톰 크기의 데이터를 저장하고, 2)C1 혹은 C2 명령을 사용해 연산을 수행한 후, 3)인 플레이스로 DRAM에 데이터를 다시 업데이트할 수 있다.
- [0072] 이를 위해, 호스트 프로세서(예를 들어, CPU)는 NTT 연산을 원하는 데이터를 PIM 기능이 활성화된 메모리 영역으로 복사하고, 메모리 위치와 데이터의 길이, NTT 파라미터 정보를 가지고 PIM 기능이 활성화된 메모리 영역에 읽기/쓰기 그리고 연산 명령(예를 들어, C1 또는 C2)을 보낸다. NTT 연산과 관련된 모든 명령이 끝난 후 호스트 프로세서는 메모리 읽기 명령을 통해 NTT 연산이 완료된 데이터를 읽을 수 있다.
- [0073] 도면(410)을 참조하면, 인트라 아톰 케이스에서는 RD 커맨드에 따라 제2 아톰 버퍼(S)에 두 데이터가 저장되고, C1 커맨드에 따라 연산기(C)에서 연산을 수행하고, WR 커맨드에 따라 다시 제2 아톰 버퍼(S)에 연산 결과를 덮어 쓸 수 있다.
- [0074] 도면(420)을 참조하면, 인트라 로 케이스에서는 RD 커맨드에 따라 제2 아톰 버퍼(S)에 제1 데이터가 저장되고, 제1 아톰 버퍼(P)에 제2 데이터가 저장되고, C2 커맨드에 따라 연산기(C)에서 연산을 수행하고, WR 커맨드에 따라 제1 데이터에 대응하는 연산 결과를 제2 아톰 버퍼(S)에 덮어 쓰고, 제2 데이터에 대응하는 연산 결과를 제1 아톰 버퍼(P)에 덮어 쓸 수 있다.
- [0075] 도면(430)을 참조하면, 인터 로 케이스에서는 RD 커맨드에 따라 제2 아톰 버퍼(S)에 제1 데이터가 저장되고, PR/ACT 커맨드에 따라 대상 행을 변경하고, 변경된 행에서 RD 커맨드에 따라 제1 아톰 버퍼(P)에 제2 데이터가 저장되고, C2 커맨드에 따라 연산기(C)에서 연산을 수행하고, WR 커맨드에 따라 제2 데이터에 대응하는 연산 결과를 제1 아톰 버퍼(P)에 덮어 쓰고, PR/ACT 커맨드에 따라 대상 행을 변경하고, 변경된 행에서 WR 커맨드에 따라 제1 데이터에 대응하는 연산 결과를 제2 아톰 버퍼(S)에 덮어 쓸 수 있다.

- [0076] 도 5는 일 실시예에 따른 파이프라이닝 방법을 설명하기 위한 도면이다.
- [0077] 도 5를 참조하면, 도면(510)은 인트라 아톰 케이스에서 파이프라이닝을 사용하지 않고 두 번의 연산을 수행하는 예시를 도시한 도면이고, 도면(515)은 인트라 아톰 케이스에서 2개의 아톰 버퍼를 이용한 파이프라이닝을 통해 두 번의 연산을 수행하는 예시를 도시한 도면이다.
- [0078] 도면(510)을 참조하면, 하나의 아톰 버퍼(0번 아톰 버퍼)만 사용하는 인트라 아톰 케이스에서, 첫번째 데이터 쌍에 대한 연산이 완료되면 두번째 데이터 쌍에 대한 연산이 수행된다. 반면에, 도면(515)을 참조하면, 두 개의 아톰 버퍼(0번 아톰 버퍼와 1번 아톰 버퍼)를 사용하는 인트라 아톰 케이스에서, 0번 아톰 버퍼에 저장된 첫번째 데이터 쌍에 대한 연산을 수행할 때, 1번 아톰 버퍼에 두 번째 데이터 쌍을 저장할 수 있고, 두번째 데이터 쌍에 대한 연산을 수행할 때 0번 아톰 버퍼에 첫번째 데이터 쌍의 연산 결과를 덮어 쓸 수 있다. 도면(510)과 도면(515)을 참조하면, 파이프라이닝을 통해 데이터 수령을 위해 기다리는 시간이 줄어드는 것을 확인할 수 있다.
- [0079] 도면(520)은 인트라 로 케이스에서 파이프라이닝을 사용하지 않고 두 번의 연산을 수행하는 예시를 도시한 도면이고, 도면(525)은 인트라 로 케이스에서 2개의 아톰 버퍼를 이용한 파이프라이닝을 통해 두 번의 연산을 수행하는 예시를 도시한 도면이다.
- [0080] 도면(520)을 참조하면, 두개의 아톰 버퍼(0번 아톰 버퍼와 1번 아톰 버퍼)를 사용하는 인트라 로 케이스에서, 첫번째 데이터 쌍에 대한 연산이 완료되면 두번째 데이터 쌍에 대한 연산이 수행된다. 반면에, 도면(525)을 참조하면, 네 개의 아톰 버퍼(0번 아톰 버퍼 내지 3번 아톰 버퍼)를 사용하는 인트라 로 케이스에서, 0번 아톰 버퍼 및 1번 아톰 버퍼에 저장된 첫번째 데이터 쌍에 대한 연산을 수행할 때, 2번 아톰 버퍼 및 3번 아톰 버퍼에 두 번째 데이터 쌍을 저장할 수 있고, 두번째 데이터 쌍에 대한 연산을 수행할 때 0번 아톰 버퍼 및 1번 아톰 버퍼에 첫번째 데이터 쌍의 연산 결과를 덮어 쓸 수 있다. 도면(520)과 도면(525)을 참조하면, 파이프라이닝을 통해 데이터 수령을 위해 기다리는 시간이 줄어드는 것을 확인할 수 있다.
- [0081] 도면(530)은 인터 로 케이스에서 파이프라이닝을 사용하지 않고 두 번의 연산을 수행하는 예시를 도시한 도면이고, 도면(535)은 인터 로 케이스에서 2개의 아톰 버퍼를 이용한 파이프라이닝을 통해 두 번의 연산을 수행하는 예시를 도시한 도면이다.
- [0082] 도면(530)을 참조하면, 두개의 아톰 버퍼(0번 아톰 버퍼와 1번 아톰 버퍼)를 사용하는 인터 로 케이스에서, 첫번째 데이터 쌍에 대한 연산이 완료되면 두번째 데이터 쌍에 대한 연산이 수행된다. 반면에, 도면(535)을 참조하면, 네 개의 아톰 버퍼(0번 아톰 버퍼 내지 3번 아톰 버퍼)를 사용하는 인터 로 케이스에서, 0번 아톰 버퍼에 첫번째 데이터 쌍의 제1 데이터를 저장하고, 2번 아톰 버퍼에 두번째 데이터 쌍의 제1 데이터를 저장한 뒤, 대상 행을 변경하고, 변경된 행에서 1번 아톰 버퍼에 첫번째 데이터 쌍의 제2 데이터를 저장하고, 3번 아톰 버퍼에 두번째 데이터 쌍의 제2 데이터를 저장한다. 3번 아톰 버퍼에 두번째 데이터 쌍의 제2 데이터를 저장할 때, 첫번째 데이터 쌍에 대한 연산이 수행될 수 있고, 두번째 데이터 쌍에 대한 연산이 수행될 때, 첫번째 데이터 쌍의 제2 데이터에 대한 연산 결과가 1번 아톰 버퍼에 저장될 수 있다. 이후, 두번째 데이터 쌍의 제2 데이터에 대한 연산 결과가 3번 아톰 버퍼에 저장될 수 있고, 대상 행이 변경되고, 변경된 행에서 0번 아톰 버퍼에 첫번째 데이터 쌍의 제1 데이터에 대한 연산 결과를 저장하고, 2번 아톰 버퍼에 두번째 데이터 쌍의 제1 데이터에 대한 연산 결과를 저장할 수 있다.
- [0083] 도면(530)과 도면(535)을 참조하면, 파이프라이닝을 통해 연산에 걸리는 시간과 메모리 접근에 걸리는 시간을 서로 겹치게 함으로써 전체 실행시간을 단축할 수 있을 뿐만 아니라, 나아가, 로 액티베이션의 개수를 줄임(도면(530)의 경우에는 4번, 도면(535)의 경우에는 2번)으로써 추가적인 실행시간 단축이 가능하다.
- [0084] 도 6은 일 실시예에 따른 메모리 장치의 동작 방법을 설명하기 위한 순서도이다.
- [0085] 설명의 편의를 위해, 단계들(610 내지 640)은 도 1에 도시된 메모리 장치(110)를 사용하여 수행되는 것으로 기술된다. 그러나 이 단계들(610 내지 640)은 어떤 다른 적절한 전자 기기를 통해, 그리고 어떤 적절한 시스템 내에서도 사용될 수 있을 것이다.
- [0086] 나아가, 도 6의 동작은 도시된 순서 및 방식으로 수행될 수 있지만, 도시된 실시예의 사상 및 범위를 벗어나지 않으면서 일부 동작의 순서가 변경되거나 일부 동작이 생략될 수 있다. 도 6에 도시된 다수의 동작은 병렬로 또는 동시에 수행될 수 있다.
- [0087] 단계(610)에서, 메모리 장치(110)는 호스트로부터 커맨드를 수신할 수 있다. 메모리 장치(110)는 같은 버퍼 내

의 데이터 사이의 연산을 지시하는 제1 연산 커맨드와 서로 다른 버퍼 내의 데이터 사이의 연산을 지시하는 제2 연산 커맨드를 수신할 수 있다.

- [0088] 단계(620)에서, 메모리 장치(110)는 복수의 버퍼들 중에서, 커맨드에 대응하는 제1 데이터 및 제2 데이터를 저장할 버퍼를 결정할 수 있다.
- [0089] 단계(630)에서, 메모리 장치(110)는 결정된 버퍼에 제1 데이터 및 제2 데이터를 저장할 수 있다. 메모리 장치(110)는 연산기가 복수의 스테이지들로 구성된 연산을 수행할 때, 복수의 버퍼들은 복수의 스테이지들로 구성된 연산의 입력 데이터 및 출력 데이터 중 적어도 하나를 저장할 수 있다.
- [0090] 단계(640)에서, 메모리 장치(110)는 제1 데이터 및 제2 데이터 중 적어도 하나에 기초하여 버터플라이 연산을 수행할 수 있다.
- [0091] 메모리 장치(110)는 해당 스테이지 연산이 완료되면, 제1 데이터를 저장하고 있는 버퍼는 제1 데이터를, 제1 데이터에 대응하는 버터플라이가 연산 결과로 업데이트할 수 있고, 제2 데이터를 저장하고 있는 버퍼는 제2 데이터를, 제2 데이터에 대응하는 버터플라이가 연산 결과로 업데이트할 수 있다.
- [0092] 메모리 장치(110)가 호스트로부터 제2 연산 커맨드를 수신할 경우, 제1 데이터를 제1 버퍼에 저장하고, 제2 데이터를 제1 버퍼와 상이한 제2 버퍼에 저장하고, 해당 스테이지 연산이 완료되면, 제1 버퍼는 제1 데이터를, 제1 데이터에 대응하는 버터플라이가 연산 결과로 업데이트하고, 제2 버퍼는 제2 데이터를, 제2 데이터에 대응하는 버터플라이가 연산 결과로 업데이트할 수 있다.
- [0093] 메모리 장치(110)가 호스트로부터 제1 연산 커맨드를 수신할 경우, 제1 데이터 및 제2 데이터를 제2 버퍼에 저장하고, 해당 스테이지 연산이 완료되면, 제2 버퍼는 제1 데이터를, 제1 데이터에 대응하는 버터플라이가 연산 결과로 업데이트하고, 제2 버퍼는 제2 데이터를, 제2 데이터에 대응하는 버터플라이가 연산 결과로 업데이트할 수 있다.
- [0094] 이상에서 설명된 실시예들은 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치, 방법 및 구성요소는, 예를 들어, 프로세서, 컨트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 운영 체제 상에서 수행되는 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 컨트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.
- [0095] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(signal wave)에 영구적으로, 또는 일시적으로 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.
- [0096] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만

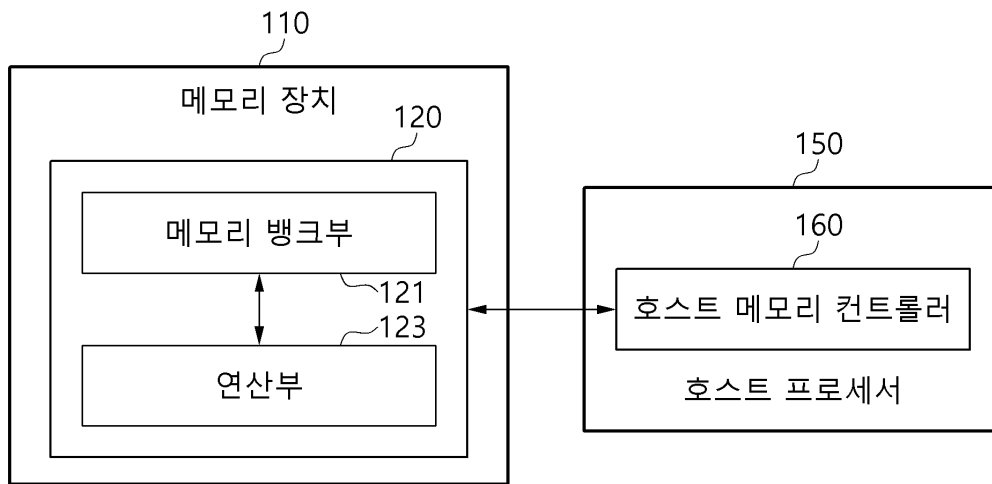
아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

[0097] 이상과 같이 실시예들이 비록 한정된 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기를 기초로 다양한 기술적 수정 및 변형을 적용할 수 있다. 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.

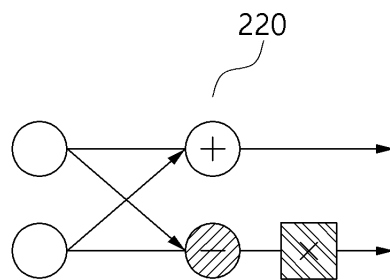
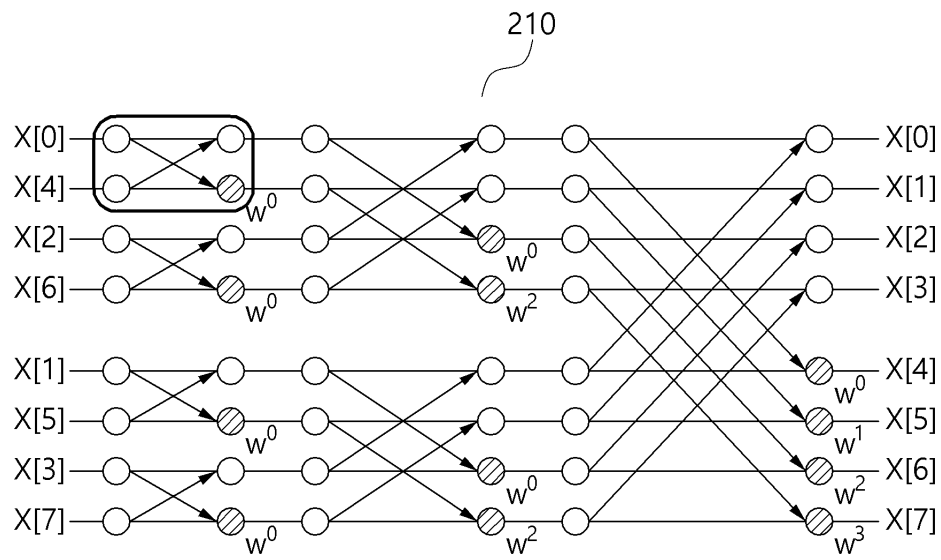
[0098] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

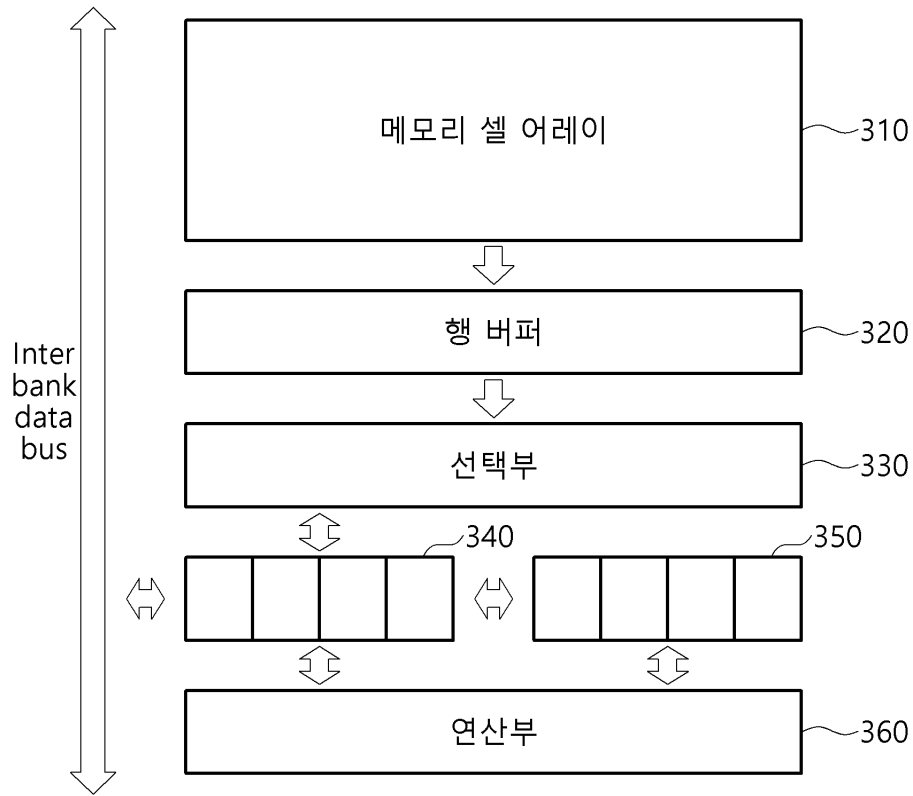
도면1



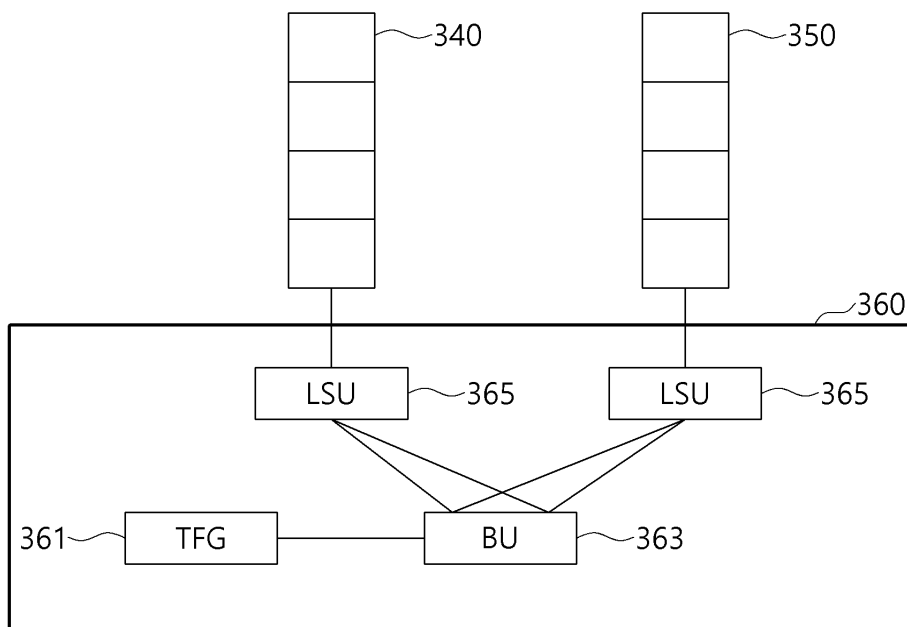
도면2



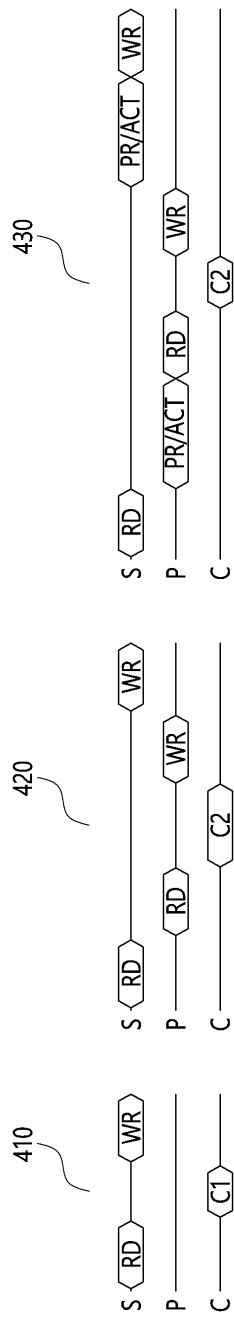
도면3a



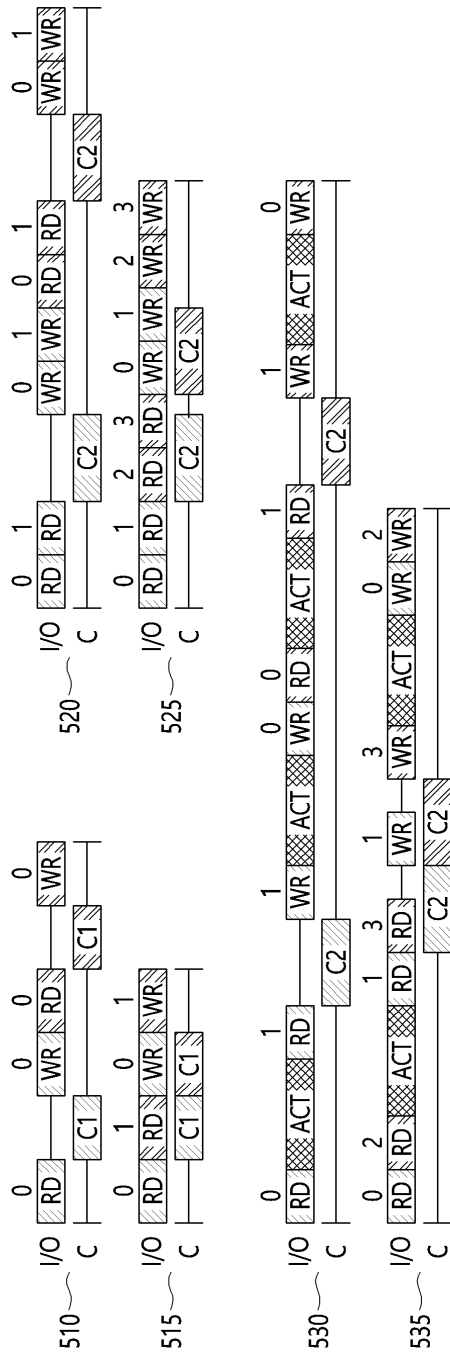
도면3b



도면4



도면5



도면6

