(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2023-0095795
(43) 공개일자 2023년06월29일

(51) 국제특허분류(Int. Cl.)
G06N 3/063 (2023.01) *G06F 12/0802* (2016.01)
G06F 3/06 (2006.01)
(52) CPC특허분류
G06N 3/063 (2013.01)
G06F 12/0802 (2013.01)
(21) 출원번호 10-2022-0137080
(22) 출원일자 2022년10월24일
심사청구일자 없음
(30) 우선권주장
1020210184439 2021년12월22일 대한민국(KR)

(71) 출원인
에스케이하이닉스 주식회사
경기도 이천시 부발읍 경충대로 2091
포항공과대학교 산학협력단
경상북도 포항시 남구 청암로 77 (지곡동)
(72) 발명자
함형규
경상북도 포항시 남구 유동길48번길 14-20(효자동) 302
조현욱
인천광역시 부평구 세월천로 16(청천동, 청천푸르지오아파트) 102-1101
(뒷면에 계속)
(74) 대리인
김선종

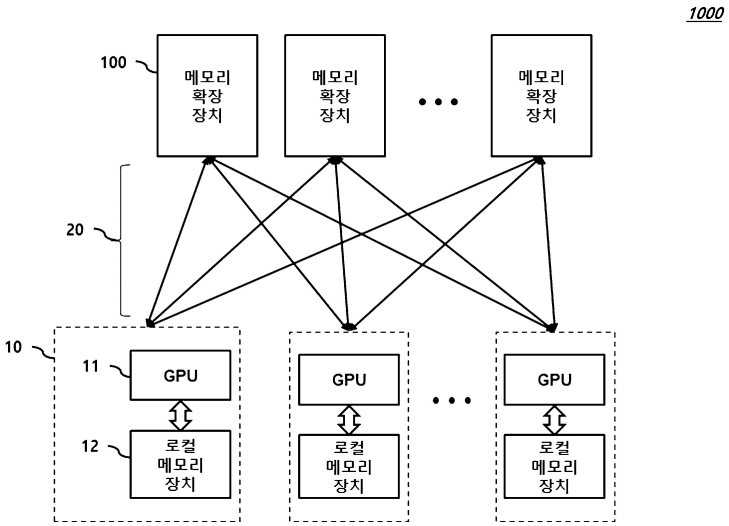
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **NDP 기능을 포함하는 호스트 장치 및 이를 포함하는 가속기 시스템**

(57) 요 약

본 기술에 의한 호스트 장치는 NDP 요청을 생성하는 단위 프로세서, NDP 요청을 수신하는 호스트 확장 제어 회로, 및 호스트 확장 제어 회로의 제어에 따라 NDP 요청에 대응하는 데이터를 저장하는 로컬 메모리 장치를 포함하되, 호스트 확장 제어 회로는 NDP 요청에 대응하여 로컬 메모리 장치에 대한 읽기 또는 쓰기를 수행하는 요청 처리 동작과 요청된 데이터를 이용한 연산 동작을 함께 수행한다.

대 표 도 - 도1



(52) CPC특허분류
G06F 3/061 (2013.01)
G06F 3/0658 (2013.01)
G06F 3/0659 (2013.01)
G06F 2212/1016 (2013.01)
G06F 2212/1032 (2013.01)

(72) 발명자
성효진
경상북도 포항시 남구 지곡로 155(지곡동, 교수아파트) 8-103
박은혁
경상북도 포항시 남구 지곡로 155(지곡동, 교수아파트) 9-1803
김광선
경상북도 포항시 남구 지곡로 155(지곡동, 교수아파트) 7-1603

이 발명을 지원한 국가연구개발사업

과제고유번호	1711134476
과제번호	2021-0-00310-001
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	SW컴퓨팅산업원천기술개발(R&D,정보화)
연구과제명	인공지능 학습/추론 효율성 향상을 위한 서버용 SW 프레임워크 개발
기 여 율	1/1
과제수행기관명	에스케이텔레콤(주)
연구기간	2021.04.01 ~ 2021.12.31

명 세 서

청구범위

청구항 1

NDP 요청을 생성하는 단위 프로세서;

상기 NDP 요청을 수신하는 호스트 확장 제어 회로; 및

상기 호스트 확장 제어 회로의 제어에 따라 상기 NDP 요청에 대응하는 데이터를 저장하는 로컬 메모리 장치

를 포함하되,

상기 호스트 확장 제어 회로는 상기 NDP 요청에 대응하여 상기 로컬 메모리 장치에 대한 읽기 또는 쓰기를 수행하는 요청 처리 동작과 요청된 데이터를 이용한 연산 동작을 함께 수행하는 호스트 장치.

청구항 2

청구항 1에 있어서, 상기 호스트 확장 제어 회로는

상기 NDP 요청을 수신하는 인터페이스 회로; 및

상기 인터페이스 회로에서 전달된 상기 NDP 요청에 대응하여 상기 요청 처리 동작과 상기 연산 동작을 제어하는 호스트 NDP 요청 제어 회로

를 포함하는 호스트 장치.

청구항 3

청구항 2에 있어서, 상기 호스트 NDP 요청 제어 회로는

상기 NDP 요청을 식별하는 필터 회로;

상기 필터 회로에서 식별된 상기 NDP 요청에 따라 상기 요청 처리 동작을 위한 요청을 전달하고 상기 연산 동작을 수행하는 NDP 회로; 및

상기 전달된 요청에 따라 상기 로컬 메모리 장치를 제어하는 메모리 컨트롤러

를 포함하는 호스트 장치.

청구항 4

청구항 3에 있어서,

상기 호스트 NDP 요청 제어 회로는 상기 필터 회로와 상기 메모리 컨트롤러 사이에 연결된 캐시 메모리 및 캐시 메모리 제어기를 더 포함하고,

상기 호스트 확장 제어 회로는 연산 동작이 필요하지 않은 단순 요청을 더 수신하고,

상기 필터 회로가 상기 단순 요청을 식별하는 경우 상기 단순 요청을 상기 캐시 메모리 제어기를 통해 상기 메모리 컨트롤러로 바이패스 하는 호스트 장치.

청구항 5

청구항 4에 있어서, 상기 필터 회로는 주소 정보를 포함하는 테이블을 저장하고, 상기 필터 회로는 상기 주소 정보를 참조하여 NDP 요청과 단순 요청을 식별하는 호스트 장치.

청구항 6

청구항 3에 있어서, 상기 NDP 회로는

상기 NDP 요청에 대응하는 연산 동작을 수행하는 연산 회로;

상기 연산 동작을 위한 인스트럭션과 상기 요청 처리 동작을 위한 요청을 저장하는 인스트럭션 저장 회로; 및

상기 연산 동작에 필요한 다수의 레지스터를 포함하는 레지스터 파일

을 포함하는 호스트 장치.

청구항 7

청구항 6에 있어서, 상기 NDP 회로는 다수의 인스트럭션을 미리 저장하는 인스트럭션 캐시를 더 포함하되, 상기 인스트럭션 저장 회로는 상기 NDP 요청에 대응하는 인스트럭션을 상기 인스트럭션 캐시로부터 수신하여 저장하는 호스트 장치.

청구항 8

청구항 7에 있어서, 상기 NDP 회로는 상기 NDP 요청에 포함된 정보를 이용하여 디코딩 동작을 수행하는 요청 디코더를 더 포함하되,

상기 요청 디코더는 NDP 요청과 인스트럭션 캐시 주소를 연관하여 저장하는 NDP 커널 테이블을 저장하는 호스트 장치.

청구항 9

청구항 8에 있어서, 상기 NDP 회로는

NDP 요청과 연산 동작 시 사용할 레지스터의 시작 주소를 연관하여 저장하는 마이크로 컨텍스트 저장 회로; 및

상기 시작 주소를 참조하여 연산 동작 시 사용하는 레지스터의 주소를 생성하는 레지스터 주소 변환 회로

를 더 포함하는 호스트 장치.

청구항 10

청구항 2에 있어서, 상기 호스트 확장 제어 회로는 상기 인터페이스 회로에 연결되어 NDP 요청을 생성하는 DMA 회로를 더 포함하고,

상기 DMA 회로에서 생성된 NDP 요청은 상기 인터페이스 회로를 통해 상기 NDP 요청 제어 회로에 제공되거나 외부로 제공되는 호스트 장치.

청구항 11

단위 프로세서를 포함하는 호스트 장치, 메모리 확장 장치, 및 상기 호스트 장치와 상기 메모리 확장 장치를 연결하는 인터커넥트 네트워크를 포함하되,

상기 호스트 장치는 상기 단위 프로세서에서 제공되는 NDP 요청을 수신하는 호스트 확장 제어 회로; 및 상기 호스트 확장 제어 회로의 제어에 따라 상기 NDP 요청에 대응하는 데이터를 저장하는 로컬 메모리 장치를 포함하고,

상기 호스트 확장 제어 회로는 상기 NDP 요청에 대응하여 상기 로컬 메모리 장치에 대한 읽기 또는 쓰기를 수행하는 요청 처리 동작과 요청된 데이터를 이용한 연산 동작을 함께 수행하는 가속기 시스템.

청구항 12

청구항 11에 있어서, 상기 호스트 확장 제어 회로는

상기 NDP 요청을 수신하는 인터페이스 회로; 및

상기 인터페이스 회로에서 전달된 상기 NDP 요청에 대응하여 상기 요청 처리 동작과 상기 연산 동작을 제어하는 호스트 NDP 요청 제어 회로

를 포함하는 가속기 시스템

청구항 13

청구항 12에 있어서, 상기 호스트 NDP 요청 제어 회로는

상기 NDP 요청을 식별하는 필터 회로;

상기 필터 회로에서 식별된 상기 NDP 요청에 따라 상기 요청 처리 동작을 위한 요청을 전달하고 상기 연산 동작을 수행하는 NDP 회로; 및

상기 전달된 요청에 따라 상기 로컬 메모리 장치를 제어하는 메모리 컨트롤러

를 포함하는 가속기 시스템.

청구항 14

청구항 13에 있어서,

상기 호스트 NDP 요청 제어 회로는 상기 필터 회로와 상기 메모리 컨트롤러 사이에 연결된 캐시 메모리 및 캐시 메모리 제어기를 더 포함하고,

상기 호스트 확장 제어 회로는 연산 동작이 필요하지 않은 단순 요청을 더 수신하고,

상기 필터 회로가 상기 단순 요청을 식별하는 경우 상기 단순 요청을 상기 캐시 메모리 제어기를 통해 상기 메모리 컨트롤러로 바이패스 하는 가속기 시스템.

청구항 15

청구항 14에 있어서, 상기 필터 회로는 주소 정보를 포함하는 테이블을 저장하고, 상기 필터 회로는 상기 주소 정보를 참조하여 NDP 요청과 단순 요청을 식별하는 가속기 시스템.

청구항 16

청구항 13에 있어서, 상기 NDP 회로는

상기 NDP 요청에 대응하는 연산 동작을 수행하는 연산 회로;

상기 연산 동작을 위한 인스트럭션과 상기 요청 처리 동작을 위한 요청을 저장하는 인스트럭션 저장 회로; 및

상기 연산 동작에 필요한 다수의 레지스터를 포함하는 레지스터 파일

을 포함하는 가속기 시스템.

청구항 17

청구항 16에 있어서, 상기 NDP 회로는 다수의 인스트럭션을 미리 저장하는 인스트럭션 캐시를 더 포함하되, 상기 인스트럭션 저장 회로는 상기 NDP 요청에 대응하는 인스트럭션을 상기 인스트럭션 캐시로부터 수신하여 저장하는 가속기 시스템.

청구항 18

청구항 17에 있어서, 상기 NDP 회로는 상기 NDP 요청에 포함된 정보를 이용하여 디코딩 동작을 수행하는 요청 디코더를 더 포함하되,

상기 요청 디코더는 NDP 요청과 인스트럭션 캐시 주소를 연관하여 저장하는 NDP 커널 테이블을 저장하는 가속기 시스템.

청구항 19

청구항 18에 있어서, 상기 NDP 회로는

NDP 요청과 연산 동작 시 사용할 레지스터의 시작 주소를 연관하여 저장하는 마이크로 컨텍스트 저장 회로; 및

상기 시작 주소를 참조하여 연산 동작 시 사용하는 레지스터의 주소를 생성하는 레지스터 주소 변환 회로

를 더 포함하는 가속기 시스템.

청구항 20

청구항 12에 있어서, 상기 호스트 확장 제어 회로는 상기 인터페이스 회로에 연결되어 NDP 요청을 생성하는 DMA 회로를 더 포함하고,

상기 DMA 회로에서 생성된 NDP 요청은 상기 인터페이스 회로를 통해 상기 NDP 요청 제어 회로에 제공되거나 외부로 제공되는 가속기 시스템.

발명의 설명

기 술 분 야

[0001] 본 기술은 NDP 기능을 포함하는 호스트 장치 및 이를 포함하는 가속기 시스템에 관한 것이다.

배 경 기 술

[0002] 심층 신경망(DNN: Deep Neural Network)의 파라미터 개수가 많아지고, 학습 데이터의 크기가 증가하며 학습 알고리즘의 반복적인 수행으로 인하여 가속기의 연산 성능을 효율화하는 것이 중요해지고 있다.

[0003] 심층 신경망을 이용한 동작은 연산 동작, 메모리 동작 및 통신 동작으로 구분될 수 있으며 컨벌루션 연산 등을 위해 수행되는 행렬 곱셈 연산은 연산 동작의 가장 큰 부분을 차지한다.

[0004] 연산 동작을 효율적으로 수행하기 위하여 텐서 코어, 매트릭스 코어와 같이 행렬 곱셈을 가속화하는 특수한 연산 유닛을 포함한 그래픽 프로세서(GPU)가 사용되고 있다.

[0005] 연산 동작에 비하여 메모리 동작과 통신 동작의 개선은 지체되고 있으며 이에 따라 최신 심층 신경망에서 메모리 동작과 통신 동작이 차지하는 비중이 증가하고 있다.

[0006] 최근 데이터 근접 연산 기술(NDP: Near Data Processing)이나 메모리 내 연산 기술(PIM: Processing In Memory) 기술이 도입되고 있으나 이는 메모리 장치 내부에 연산 회로를 추가함으로써 저장 공간을 희생하는 문제가 있다.

선행기술문헌

특허문헌

[0007] (특허문헌 0001) KR 10-2019-0018888 A

(특허문헌 0002) US 2021-0117131 A1

(특허문헌 0003) US 2021-0349837 A1

(특허문헌 0004) KR 10-2020-0018188 A

(특허문헌 0005) US 2021-0311739 A1

발명의 내용

해결하려는 과제

[0008] 본 기술은 다수의 호스트 장치와 다수의 메모리 확장 장치를 서로 연결하여 연산 성능과 메모리 저장 공간이 보다 확장된 가속기 시스템을 제공한다.

[0009] 본 기술은 메모리 동작 및 통신 동작을 연산 동작과 중첩하여 수행함으로써 처리 성능을 향상시킬 수 있는 가속기 시스템을 제공한다.

[0010] 본 기술은 NDP 기능을 포함하는 호스트 장치를 제공하여 원거리 메모리 장치에 대한 접근에 필요한 시간을 줄임으로써 연산 효율을 향상시킨다.

과제의 해결 수단

[0011] 본 발명의 일 실시예에 의한 호스트 장치는 NDP 요청을 생성하는 단위 프로세서, NDP 요청을 수신하는 호스트

확장 제어 회로, 및 호스트 확장 제어 회로의 제어에 따라 NDP 요청에 대응하는 데이터를 저장하는 로컬 메모리 장치를 포함하되, 호스트 확장 제어 회로는 NDP 요청에 대응하여 로컬 메모리 장치에 대한 읽기 또는 쓰기를 수행하는 요청 처리 동작과 요청된 데이터를 이용한 연산 동작을 함께 수행한다.

[0012] 본 발명의 일 실시예에 의한 가속기 시스템은 단위 프로세서를 포함하는 호스트 장치, 메모리 확장 장치, 및 호스트 장치와 메모리 확장 장치를 연결하는 인터커넥트 네트워크를 포함하되, 호스트 장치는 단위 프로세서에서 제공되는 NDP 요청을 수신하는 호스트 확장 제어 회로, 및 호스트 확장 제어 회로의 제어에 따라 NDP 요청에 대응하는 데이터를 저장하는 로컬 메모리 장치를 포함하고, 호스트 확장 제어 회로는 NDP 요청에 대응하여 로컬 메모리 장치에 대한 읽기 또는 쓰기를 수행하는 요청 처리 동작과 요청된 데이터를 이용한 연산 동작을 함께 수행한다.

발명의 효과

[0013] 본 기술에 의한 호스트 장치는 NDP 기능을 함께 수행하여 메모리 장치에 접근하는데 필요한 시간을 절약할 수 있다.

[0014] 본 기술에 의한 가속기 시스템에서는 NDP 요청을 통해 메모리 읽기 쓰기 동작과 함께 연산 동작을 중첩함으로써 심층 신경망 처리 동작의 성능을 향상시킬 수 있다.

도면의 간단한 설명

[0015] 도 1은 본 발명의 일 실시예에 의한 가속기 시스템을 나타내는 블록도.

도 2는 본 발명의 일 실시예에 의한 메모리 확장 장치를 나타낸 블록도.

도 3은 본 발명의 일 실시예에 의한 그래픽 처리 장치와 메모리 확장 장치에 대한 제어 과정을 나타낸 설명도.

도 4는 종래의 심층 신경망 연산 과정을 나타낸 설명도.

도 5는 본 발명의 일 실시예에 의한 심층 신경망 연산 과정을 나타낸 설명도.

도 6은 본 발명의 일 실시예에 의한 확장 제어 회로를 나타낸 블록도.

도 7은 본 발명의 일 실시예에 의한 NDP 회로를 나타낸 블록도.

도 8은 본 발명의 일 실시예에 의한 메모리 확장 장치에 사용되는 테이블 구조.

도 9는 본 발명의 일 실시예에 의한 NDP 커널의 동작을 나타내는 소프트웨어 코드.

도 10은 NDP 커널 개시 패킷에 의해 설정된 테이블의 일 예를 나타낸 도면.

도 11은 본 발명의 다른 실시예에 의한 가속기 시스템을 나타내는 블록도.

도 12는 본 발명의 다른 실시예에 의한 그래픽 처리 장치를 나타내는 블록도.

도 13은 본 발명의 다른 실시예에 의한 단위 프로세서를 나타내는 블록도.

도 14는 본 발명의 다른 실시예에 의한 그래픽 처리 장치와 메모리 확장 장치에 대한 제어 과정을 나타낸 설명도.

도 15는 본 발명의 다른 실시예에 의한 심층 신경망 연산 과정을 나타낸 설명도.

발명을 실시하기 위한 구체적인 내용

[0016] 이하에서는 첨부한 도면을 참조하여 본 발명의 실시예를 개시한다.

[0017] 도 1은 본 발명의 일 실시예에 의한 가속기 시스템(1000)을 나타내는 블록도이다.

[0018] 가속기 시스템(1000)은 다수의 호스트 장치(10), 다수의 메모리 확장 장치(100), 호스트 장치(10)와 메모리 확장 장치(100)를 연결하는 인터커넥트 네트워크(20)를 포함한다.

[0019] 본 실시예에서 호스트 장치(10), 인터커넥트 네트워크(20), 및 메모리 확장 장치(100) 사이에서 송수신되는 요청은 주소와 데이터가 미리 정해진 형태로 포맷된 패킷 구조를 가진다.

[0020] 호스트 장치(10)는 프로세서(11)와 메모리 장치(12)를 포함한다. 본 실시예에서 프로세서(11)는 그래픽 프로세

서(11, GPU)이며 이에 따라 호스트 장치(10)는 그래픽 처리 장치(10)로 지칭할 수 있다.

[0021] 메모리 장치(12)는 GPU(11)에서 전용으로 사용하는 메모리 장치로서 그래픽 메모리 장치(12) 또는 로컬 메모리 장치(12)로 지칭할 수 있다.

[0022] 그래픽 메모리 장치(12)는 특정한 종류의 메모리 장치로 한정되지 않으며 디램, 그래픽 디램, HBM 등 다양한 메모리 장치가 사용될 수 있다.

[0023] GPU(11) 또는 메모리 확장 장치(100)는 NDP 기능을 포함할 수 있는데 도 1은 메모리 확장 장치(100)에 NDP 기능을 포함하는 실시예에 대응하며, GPU에 NDP 기능이 포함되는 실시예는 도 11 이하를 참조하여 개시한다.

[0024] NDP 기능을 포함하는 메모리 확장 장치(100)를 NDPX(NDP Expander) 장치(100)로 지칭할 수 있다.

[0025] 도 2에 도시된 바와 같이 메모리 확장 장치(100)는 확장 제어 회로(110)와 다수의 메모리 장치(120)를 포함한다. 메모리 장치(120)를 확장 메모리 장치(120) 또는 원격 메모리 장치(120)로 지칭할 수 있다.

[0026] 확장 제어 회로(110)는 스위치 기능을 수행하여 인터커넥트 네트워크(20)를 통한 통신 기능을 지원할 수 있다.

[0027] 인터커넥트 네트워크(20)는 다수의 그래픽 처리 장치(10)와 다수의 메모리 확장 장치(100)를 완전 연결하는 형태의 네트워크이다.

[0028] 본 실시예에서 다수의 GPU(11)와 다수의 메모리 확장 장치(100)는 주소 공간을 공유한다.

[0029] 이에 따라 각각의 GPU(11)는 읽기 또는 쓰기 요청에 의해 원격 메모리 장치(120)에 접근할 수 있다. 또한 어느 하나의 메모리 확장 장치는 다른 메모리 확장 장치에 접근할 수 있다.

[0030] 도 3은 본 발명의 일 실시예에 의한 그래픽 처리 장치(10)와 메모리 확장 장치(100)에 대한 제어 과정을 나타낸 설명도이다.

[0031] 심층 신경망(DNN) 응용 프로그램(1)은 메모리 확장 장치(100)를 지원하는 컴파일러(2)에 의해 컴파일된다.

[0032] 컴파일러(2)는 그래픽 처리 장치(10)에서 수행하는 GPU 커널(3)과 메모리 확장 장치(100)에서 수행하는 NDP 커널(4)을 생성한다.

[0033] 컴퓨터 과학 분야에서 커널은 다양한 의미로 해석될 수 있는데 본 실시예에서 커널은 함수와 같은 의미로 해석한다.

[0034] 그래픽 처리 장치(10)에서 GPU 커널(3)을 수행하는 중에 메모리 확장 장치(100)에 대한 읽기 또는 쓰기 요청이 발생할 수 있고, 각 요청에 대응하여 메모리 확장 장치(100)는 대응하는 NDP 커널(4)을 수행할 수 있다.

[0035] GPU 커널(3)에서 발생한 요청과 이에 대응하는 NDP 커널(4)은 컴파일러(2)에서 미리 결정될 수 있다.

[0036] 도 4는 종래의 기술에 의한 심층 신경망 연산 방식을 나타낸다.

[0037] 도 4는 "컨벌루션(CONV) 동작 -> 배치 정규화(BN) 동작 -> ReLU 동작 -> 컨벌루션(CONV) 동작" 순서로 신경망 연산이 진행되는 경우를 예시한다.

[0038] 이하에서 심층 신경망에 입력되거나, 심층 신경망의 각 레이어에서 출력되는 데이터를 텐서 데이터로 지칭한다.

[0039] 도 4에서 행렬 곱셈을 위하여 입력되는 텐서 데이터는 GPU(11)에 미리 제공된 것으로 가정한다.

[0040] 먼저 컨벌루션 동작을 위해 GPU(11)에서 행렬 곱셈을 수행하고(S1), 곱셈 결과를 로컬 메모리 장치(12)에 저장한다(S2).

[0041] 이후, 전체 곱셈 결과를 로컬 메모리 장치(12)에서 다시 읽고(S3), GPU(11)에서 누적 동작을 수행하며(S4), 평균과 표준 편차를 계산한다(S5).

[0042] 이후, 곱셈 결과를 로컬 메모리 장치(12)에서 다시 읽고(S6), 정규화 동작과 ReLU 연산을 수행한 후(S7), 연산 결과인 텐서 데이터를 로컬 메모리 장치(12)에 저장한다(S8).

[0043] 마지막으로 다음 컨벌루션 동작을 위한 행렬 곱셈(S10)을 위해 로컬 메모리 장치(12)에서 텐서 데이터를 읽어온다(S9).

[0044] 이와 같이 종래에는 GPU(11)에서 모든 연산 동작을 수행하고 GPU(11) 내부 버퍼 부족으로 인해 로컬 메모리 장

치(12)와의 사이에서 읽기, 쓰기 동작이 빈번하게 수행된다.

[0045] 또한 연산 동작과 메모리 동작이 순차적으로 진행되어 신경망 연산에 많은 시간이 걸린다.

[0046] 도 5는 본 발명의 일 실시예에 의한 심층 신경망 연산 방식을 나타낸다.

[0047] 도 5 역시 "컨벌루션(CONV) 동작 -> 배치 정규화(BN) 동작 -> ReLU 동작 -> 컨벌루션(CONV) 동작" 순서로 신경망 연산이 진행되는 경우를 예시한다.

[0048] 본 실시예에서는 컨벌루션을 위한 행렬 곱셈 동작은 GPU(11)에서 수행되나 정규화 동작과 ReLU 동작은 NDP(Near Data Processing) 기능이 포함된 메모리 확장 장치(100) 내부에서 수행할 수 있다.

[0049] 정규화 동작에 필요한 누적 연산, 평균/표준 편차 계산 역시 메모리 확장 장치(100) 내부에서 수행할 수 있다.

[0050] 도 5에서 행렬 곱셈을 위한 입력 텐서 데이터는 GPU(11)에 미리 제공된 것으로 가정한다.

[0051] 먼저 GPU(11)에서 행렬 곱셈을 수행한다(S11). 곱셈 결과는 쓰기 요청 패킷을 이용해 메모리 확장 장치(100)로 전송되어 원격 메모리 장치(120)에 저장된다(S12).

[0052] 본 실시예에서는 원격 메모리 장치(120)에 데이터를 저장하는 쓰기 동작과 메모리 확장 장치(100) 내부의 NDP 회로에서 NDP 연산 동작이 동시에 수행될 수 있다. 이를 온더플라이(ON-THE-FLY) NDP 연산 동작으로 지칭할 수 있다.

[0053] 본 실시예에서 원격 메모리 장치(120)에 대한 요청은 온더플라이 NDP 연산 동작이 함께 수행되는 요청과 그렇지 않은 요청으로 구분될 수 있다.

[0054] 이하에서 온더플라이 NDP 연산 동작이 수행되는 요청을 NDP 요청으로 지칭하고 그렇지 않은 요청을 단순 요청으로 지칭한다.

[0055] 이에 따라 GPU(11)에서 메모리 확장 장치(100)로의 쓰기 요청은 NDP 쓰기 요청 또는 단순 쓰기 요청이며, GPU(11)에서 메모리 확장 장치(100)로의 읽기 요청은 NDP 읽기 요청 또는 단순 읽기 요청이다. 도 5의 쓰기 요청과 읽기 요청은 NDP 쓰기 요청 및 NDP 읽기 요청이다.

[0056] 도 5에서 쓰기 동작(S12)과 누적 동작(S13)은 각각 다수 번(n 회, n은 자연수) 수행될 수 있다. 이때 n은 텐서 데이터의 크기와 패킷의 크기에 의존한다.

[0057] 예를 들어 텐서 데이터가 n 개의 쓰기 요청 패킷을 통해 제공되는 경우, n 회의 쓰기 요청(S12)과 각 쓰기 요청에 대응하는 NDP 동작으로서 n 회의 누적 동작(S13)이 수행될 수 있다.

[0058] 이후 평균 및 표준 편차 계산 동작(S14)을 수행한다.

[0059] 본 실시예에서 n 회의 누적 동작(S13) 및 평균 및 표준 편차 계산 동작(S14)은 하나의 NDP 커널을 통해 실행될 수 있다. 이에 대해서는 아래에서 구체적으로 개시한다.

[0060] 이후 GPU(11)는 원격 메모리 장치(120)에서 두 번째 컨벌루션 연산을 위해 텐서 데이터를 읽어 온다(S15).

[0061] 원격 메모리 장치에서 텐서 데이터를 읽어 오는 동안 온더플라이 NDP 연산 동작이 수행될 수 있다. 본 실시예에서는 정규화 및 ReLU 계산 동작(S16)이 온더플라이 NDP 연산 동작으로 수행된다.

[0062] 도 5에서는 n 회의 읽기 요청(S15)이 발생하고 각 읽기 요청에 대응하는 NDP 동작으로서 정규화 및 ReLU 계산 동작(S16)이 수행될 수 있다.

[0063] 이후 정규화 및 ReLU 계산 결과를 이용하여 다음 행렬 곱셈(S17)을 수행한다.

[0064] 본 실시예에서는 인터커넥트 네트워크(20)를 통해 GPU(11)와 메모리 확장 장치(100) 사이에서 데이터를 전송하므로 통신 동작에 필요한 시간이 추가로 필요할 수 있다.

[0065] 그러나 메모리 확장 장치(100)의 온더플라이 NDP 연산 동작을 메모리 읽기/쓰기 동작을 중첩시킬 수 있으며 이에 따라 더 많은 시간 절약이 가능하게 되며 결과적으로 전체 심층 신경망 연산 시간을 크게 줄일 수 있다.

[0066] 도 5에서는 GPU(11)에서 수행한 행렬 곱셈 결과를 메모리 확장 장치(100)에서 이용하므로 GPU 연산 동작과 NDP 연산 동작 사이에 의존 관계가 존재한다.

[0067] GPU 연산 동작과 NDP 연산 동작 사이에 의존 관계가 존재하지 않는 경우 경우에는 GPU 연산 동작과 NDP 연산 동

작도 중첩적으로 진행될 수 있으며 이 경우 더 많은 시간 절약이 가능할 수 있다.

[0068] 도 6은 본 발명의 일 실시예에 의한 확장 제어 회로(110)를 나타낸 블록도이다.

[0069] 확장 제어 회로(110)는 인터페이스 회로(111), DMA 회로(112), 다수의 NDP 요청 제어 회로(200)를 포함한다.

[0070] 인터페이스 회로(111)는 다수의 NDP 요청 제어 회로(200)와 인터커넥트 네트워크(20) 사이에서 패킷을 송수신한다.

[0071] 각각의 NDP 요청 제어 회로(200)는 대응하는 확장 메모리 장치(120)에 따라 주소 범위가 지정되며 인터페이스 회로(111)는 입력된 요청 패킷의 주소를 판단하여 대응하는 NDP 요청 제어 회로(200)에 전달할 수 있다.

[0072] DMA 회로(112)는 메모리 확장 장치(100) 내부에서 DMA 기술을 통해 요청 패킷을 생성할 수 있으며 인터페이스 회로(111)에 연결될 수 있다.

[0073] 예를 들어 DMA 회로(112)에서 생성되는 요청 패킷은 호스트 장치(10)에서 제공되는 요청 패킷과 동일한 형태를 가질 수 있다.

[0074] 이에 따라 하나의 메모리 확장 장치(100)에서 생성된 요청은 내부에서 처리될 수도 있고, 다른 메모리 확장 장치로 전송될 수도 있다.

[0075] 다수의 NDP 요청 제어 회로(200)는 인터페이스 회로(111)와 다수의 원격 메모리 장치(120) 사이에 연결되어 메모리 읽기 쓰기 동작 및 NDP 연산 동작을 수행한다.

[0076] NDP 요청 제어 회로(200)는 필터 회로(210), NDP 회로(300) 및 메모리 컨트롤러(220)를 포함한다.

[0077] 필터 회로(210)는 인터페이스 회로(111)를 통해 제공된 요청 패킷이 NDP 요청 패킷인지 단순 요청 패킷인지 식별한다. 필터 회로(210)의 구체적인 동작 내용은 이하에서 다시 개시한다.

[0078] 도 7은 본 발명의 일 실시예에 의한 NDP 회로(300)를 나타낸 블록도이다.

[0079] NDP 회로(300)는 요청 디코더(310), 요청 버퍼(320), 인스트럭션 저장 회로(330), 연산 회로(340), 인스트럭션 캐시(350), 레지스터 주소 변환 회로(360), 레지스터 파일(370), 마이크로 컨텍스트 저장 회로(380)를 포함한다.

[0080] 요청 디코더(310)는 필터 회로(210)에서 전달된 요청에서 NDP 동작에 필요한 정보가 포함되도록 요청의 포맷을 변경한다

[0081] 요청 버퍼(320)는 디코딩된 요청을 저장한다.

[0082] 인스트럭션 저장 회로(330)는 요청에 대응하는 인스트럭션을 저장한다.

[0083] 인스트럭션은 인스트럭션 캐시(350)에 미리 저장되는데 인스트럭션 캐시(350)를 참조하여 요청에 대응하는 인스트럭션을 인스트럭션 저장 회로(330)에 저장된다.

[0084] 요청에 대응하는 인스트럭션의 위치는 미리 지정될 수 있으며 이에 대해서는 아래에서 구체적으로 개시한다.

[0085] 인스트럭션 저장 회로(330)는 다수의 인스트럭션 큐(331)를 포함하며 각각의 인스트럭션 큐는 대응하는 NDP 커널을 위한 인스트럭션을 순차적으로 저장한다.

[0086] 인스트럭션 큐(331)에 저장된 인스트럭션은 연산 회로(340)에 제공되어 연산 동작에 사용된다.

[0087] 인스트럭션 저장 회로(330)는 요청 큐(332)를 더 포함한다.

[0088] 요청 큐(332)는 NDP 쓰기 요청 또는 NDP 읽기 요청에 대응하는 쓰기 요청과 읽기 요청을 저장한다.

[0089] 요청 큐(332)에 저장된 읽기 요청 및 쓰기 요청은 메모리 컨트롤러(220)에 제공되어 원격 메모리 장치(120)에 대한 읽기 쓰기 동작이 수행된다.

[0090] 예를 들어 도 5에서 쓰기 동작(S12)을 위한 쓰기 요청은 요청 큐(332)에 저장되고 누적 동작(S13), 평균 및 표준 편차 계산 동작(S14)을 위한 인스트럭션은 인스트럭션 큐(331)에 저장된다.

[0091] 연산 회로(340)는 인스트럭션 큐(331)에서 제공되는 인스트럭션에 대응하는 연산 동작을 수행한다.

[0092] 본 실시예에서는 스칼라 및 벡터를 이용한 연산, 제곱근 연산 등을 지원하는데 연산의 종류가 이에 제한되는 것

은 아니며 지원되는 연산의 종류는 실시예에 따라 다양하게 설계 변경될 수 있다.

[0093] 연산의 종류에 따른 구체적인 회로 설계는 종래에 알려진 것을 사용할 수 있으므로 구체적인 개시는 생략한다.

[0094] 인스트럭션 캐시(350)는 요청에 대응하는 인스트럭션을 미리 저장하는 회로이다.

[0095] 레지스터 파일(370)은 연산 동작시 사용하는 다수의 벡터 레지스터 및 스칼라 레지스터를 포함한다.

[0096] 레지스터 주소 변환 회로(360)는 NDP 커널에서 사용하는 논리적인 레지스터 주소와 레지스터 파일(370)에 포함된 레지스터의 물리 주소를 변환하는 역할을 수행한다.

[0097] 마이크로 컨텍스트 저장회로(380)는 마이크로 컨텍스트 테이블을 저장한다. 마이크로 컨텍스트에 대해서는 아래에서 구체적으로 개시한다..

[0098] 필터 회로(210)는 필터 테이블을 저장하여 필터링 동작에 참조할 수 있고 NDP 회로(300)는 NDP 커널 테이블과 마이크로 컨텍스트 테이블을 저장하여 NDP 커널의 실행 과정에서 필요한 정보를 관리할 수 있다.

[0099] 본 실시예에서 NDP 커널 테이블은 요청 디코더(310)에 저장되며 마이크로 컨텍스트 테이블은 마이크로 컨텍스트 저장 회로(380)에 저장된다.

[0100] 도 8(A)는 필터 테이블의 구조를 나타내고, 도 8(B)는 NDP 커널 테이블의 구조를 나타내며, 도 8(C)는 마이크로 컨텍스트 테이블의 구조를 나타낸다.

[0101] 필터 테이블은 베이스 주소 필드, 주소 바운드 필드, 피벗 차원 필드, 텐서 형태 필드, NDP 커널 ID 필드, 필터 인자 필드를 포함한다.

[0102] NDP 커널 테이블은 NDP 커널 ID 필드, 코드 위치 필드, 정적 레지스터 개수 필드, 동적 레지스터 개수 필드, 마이크로 컨텍스트 당 요청 개수 필드, 나머지 마이크로 컨텍스트 필드를 포함한다.

[0103] 마이크로 컨텍스트 테이블은 NDP 커널 ID 필드, 피벗 인덱스 필드, 정적 레지스터 베이스 ID 필드, 나머지 패킷 개수 필드를 포함한다.

[0104] 각 테이블에 포함된 필드의 의미에 대해서는 아래에서 다시 구체적으로 개시한다.

[0105] NDP 요청 패킷이 전송되었을 때 NDP 커널이 정상적으로 수행되도록 하기 위해서는 도 8에 포함된 테이블의 정보를 미리 설정하는 것이 필요하다.

[0106] 본 실시예에서 GPU(11)는 NDP 요청 패킷을 전송하기 전에 NDP 개시 패킷을 메모리 확장 장치(100)에 전송하여 필터 테이블, NDP 커널 테이블, 마이크로 컨텍스트 테이블을 미리 설정한다.

[0107] 도 9는 메모리 확장 장치(100)에서 수행되는 NDP 커널의 일 예를 나타내는 소프트웨어 코드이다.

[0108] 예시된 NDP 커널은 도 5의 누적 동작(S13)과 평균 및 표준 편차 계산 동작(S14)에 대응하는 것이다.

[0109] NDP 커널은 초기화 동작, 요청 당 함수 연산 동작, 및 완료 동작을 순차적으로 수행한다.

[0110] 도 9(A)는 초기화 동작을 나타낸다.

[0111] 초기화 동작에서는 필요한 레지스터를 초기화하는 동작을 수행할 수 있으며 NDP 개시 패킷이 수신되는 경우에 수행될 수 있다.

[0112] 도 9(A)의 코드는 벡터 레지스터(v0, v1)의 값을 각각 0으로 초기화하는 것을 나타낸다.

[0113] 도 9(B)의 코드는 요청 당 함수 연산 동작을 나타낸다. 요청 당 함수 연산 동작은 각각의 NDP 요청이 수신될 때마다 수행된다.

[0114] 예를 들어서 도 5에서 쓰기 동작(S12)과 누적 동작(S13)은 n번의 NDP 요청 패킷을 전달하여 수행되는데 이에 따라 도 9(B)의 코드는 NDP 요청 패킷이 수신될 때마다 수행되어 총 n번 수행될 수 있다.

[0115] 도 9(B)의 코드에서 REQDATA, REQADDR은 특수 목적 레지스터로서 각각 요청된 데이터와 요청된 주소를 저장한다.

[0116] 도 9(B)의 코드는 먼저 요청 데이터(REQDATA)를 벡터 레지스터(v2)에 로드하는 동작, 벡터 레지스터(v2)의 각 원소를 벡터 레지스터(v0)에 누적하는 동작, 벡터 레지스터(v2)의 각 원소를 제곱하여 벡터 레지스터(v1)에 누

적하는 동작, 요청된 주소(REQADDR)에 벡터 레지스터(V2)의 값을 저장하는 동작을 포함한다.

[0117] 도 9(C)는 완료 동작을 나타내며 도 5에서 평균 및 표준 편차 계산 동작(S14)을 수행한다.

[0118] 도 9(C)에서 FILTERARG는 필터 인자를 나타내는 특수 목적 레지스터를 나타낸다.

[0119] 먼저 필터 인자(FILTERARG)를 레지스터(r1)에 저장한다. 이때 필터 인자는 계산된 평균과 표준 편차를 저장할 주소에 대응한다.

[0120] 도 9(C)의 코드는 레지스터(v0, v1)의 각 원소에 1/4을 곱한다. 이때 4는 누적하고자 하는 행 벡터의 개수를 나타낸다.

[0121] 이후 레지스터(v0)에 저장된 평균값을 필터 인자로 지정된 주소에 저장한다.

[0122] 다음으로 레지스터(v0)의 각 원소를 제곱하여 레지스터(v0)를 갱신하고, 레지스터(v0)에서 레지스터(v1)의 값을 뺀 값(분산)을 레지스터(v1)에 저장한다.

[0123] 이후 레지스터(v1)의 각 원소에 대해서 제곱근(표준 편차)을 계산하여 레지스터(v1)의 값을 갱신한다.

[0124] 마지막으로 레지스터(v1)에 저장된 값을 필터 인자로 지정된 주소에 옵셋(0x400)을 더한 주소에 표준 편차를 저장한다.

[0125] 이하에서는 GPU(11)에서 다수의 NDP 쓰기 요청을 전송하여 메모리 확장 장치(100)에서 도 9의 NDP 커널을 수행하는 기술을 개시한다.

[0126] 본 실시예에서 GPU(11)는 NDP 쓰기 요청을 통해 2차원 텐서 데이터(A)를 메모리 확장 장치(100)에 저장한다.

[0127] 텐서 데이터는 행(X) 개수가 4, 열(Y) 개수가 32인 형태를 가지는 2차원 행렬 데이터이다. 텐서 원소($A_{x,y}$)에서 x는 행 번호, y는 열 번호를 나타낸다.

[0128] 텐서 데이터의 크기는 256 바이트이며 이에 따라 텐서 데이터의 각 텐서 원소($A_{x,y}$)는 2 바이트의 크기를 가진다.

[0129] 텐서 데이터의 베이스 주소는 0x100이고, 주소 바운드는 0x100인 것으로 가정한다. 즉 GPU(11)가 0x000에서 0x100까지의 주소 범위에 쓰기 요청을 전송하는 경우 필터 회로(210)는 해당 요청을 NDP 쓰기 요청으로 식별할 수 있다.

[0130] GPU(11)에서 메모리 확장 장치(100)로 전송하는 쓰기 요청 패킷에 저장될 수 있는 정보의 크기는 32바이트인 것으로 가정한다. 이에 따라 하나의 요청 패킷은 텐서 데이터 중 16개의 원소에 대한 쓰기 요청을 전송할 수 있으며, 하나의 텐서 데이터를 모두 전송하기 위하여 총 8개의 쓰기 요청 패킷이 전송된다.

[0131] 본 실시예에서는 하나의 행을 전송할 때 상위 열 그룹과 하위 열 그룹으로 나누어 전송하며, 상위 열 그룹에 대응하는 행 벡터를 상위 행 벡터, 하위 열 그룹에 대응하는 행 벡터를 하위 행 벡터로 지칭한다.

[0132] 이에 따라 하나의 요청 패킷에 포함되는 텐서 원소는 $A_{x,0}$ ~ $A_{x,15}$ 또는 $A_{x,16}$ ~ $A_{x,31}$인 것으로 가정한다.

[0133] 본 실시예에서 상위 행 벡터에 대한 다수의 NDP 요청과 하위 행 벡터에 대한 다수의 NDP 요청은 서로 다른 마이크로 컨텍스트에 속한다.

[0134] NDP 커널에 대한 코드는 캐시 메모리 주소 0x300부터 저장되는 것으로 가정한다. 이때 캐시 메모리 주소는 인스트럭션 캐시(350)의 주소를 나타낸다.

[0135] 전술한 바와 같이 REQDATA, REQADDR, FILTERARG는 NDP 커널에서 사용하는 특수 레지스터를 나타내며 이들은 레지스터 파일(370)에 포함될 수 있다.

[0136] REQDATA는 32 바이트의 쓰기 데이터가 저장되는 레지스터이고, REQADDR는 쓰기 요청된 주소가 저장되는 레지스터이고, FILTERARG는 필터 인자가 저장되는 레지스터인데 본 실시예에서는 계산 결과를 저장하는 주소로서 0x200인 것으로 가정한다.

[0137] 전술한 바와 같이 쓰기 동작을 수행하기 전에 GPU(11)는 메모리 확장 장치(100)에 NDP 개시 패킷을 전송하여 도 8에 도시된 테이블에 정보를 설정한다.

[0138] NDP 개시 패킷은 미리 정해진 포맷을 사용함으로써 필터 회로(210)와 NDP 회로(300)가 이를 식별할 수 있으며

NDP 개시 패킷에 포함된 정보를 디코딩하여 도 8의 테이블에 정보를 설정할 수 있다.

[0139] 본 실시예에서 NDP 개시 패킷은 베이스 주소, 주소 바운드, 피벗 차원, 텐서 형태, 필터 인자, 코드 위치, 정적 레지스터 개수, 동적 레지스터 개수 정보를 포함하며 다른 정보는 이들로부터 계산될 수 있다.

[0140] 도 10은 NDP 개시 패킷에 의해서 설정된 테이블의 정보를 나타낸다.

[0141] 도 10(A)의 필터 테이블에 베이스 주소는 0x000, 주소 바운드는 0x100, 피벗 차원은 0, 텐서 형태는(4, 32), NDP 커널 ID는 0, 필터 인자는 0x200인 행이 추가된다.

[0142] 피벗 차원은 텐서 데이터가 2차원인 것을 나타낸다. 전술한 바와 같이 필터 인자는 평균과 표준 편차가 저장될 주소를 나타낸다.

[0143] 도 10(B)의 NDP 커널 테이블에 NDP 커널 ID는 0, 코드 위치는 0x300, 정적 레지스터 개수는 2, 동적 레지스터 개수는 2, 마이크로 컨텍스트 당 요청 개수는 4, 나머지 마이크로 컨텍스트 개수는 2인 행이 저장된다.

[0144] 마이크로 컨텍스트 당 요청 개수와 나머지 마이크로 컨텍스트 개수는 계산되어 저장된 것이다.

[0145] 전술한 바와 같이 하위 행 벡터에 대응하는 16개의 원소($A_{x,0}$ ~ $A_{x,15}$)에 대한 요청과 상위 행 벡터에 대응하는 16개의 원소($A_{x,16}$ ~ $A_{x,31}$)에 대한 요청은 각각 별개의 마이크로 컨텍스트에 해당한다.

[0146] 이에 따라 총 마이크로 컨텍스트 개수는 2개가 되고, 텐서 데이터에 총 4개의 행이 있으므로 마이크로 컨텍스트 당 요청의 개수는 4개가 된다.

[0147] 도 10(C)의 마이크로 컨텍스트 테이블에 NDP 커널 ID는 0, 피벗 인덱스는 0, 정적 레지스터 베이스 ID는 0, 나머지 패킷 개수는 4인 행과 NDP 커널 ID는 0, 피벗 인덱스는 1, 정적 레지스터 베이스 ID는 2, 나머지 패킷 개수는 4인 행이 저장된다.

[0148] 피벗 인덱스는 마이크로 컨텍스트를 식별하는 정보이다. NDP 커널 테이블에서 정적 레지스터 개수는 마이크로 컨텍스트 당 할당될 수 있는 정적 레지스터를 나타낸다.

[0149] 마이크로 컨텍스트 테이블에서 피벗 인덱스 0에 대응하는 정적 레지스터 베이스 ID가 0으로 설정되고 피벗 인덱스 1에 대응하는 정적 레지스터 베이스 ID가 2로 설정된다.

[0150] 이와 같이 NDP 개시 패킷이 전송되어 테이블에 필요한 정보가 설정되면서 도 9(A)와 같이 NDP 커널의 초기화부가 동작한다.

[0151] 이후 0번 마이크로 컨텍스트에 대한 첫 번째 쓰기 요청 패킷이 전송되는 경우를 가정한다. 이때 첫 번째 쓰기 요청의 주소는 0x000으로 가정한다.

[0152] 필터 회로(210)는 필터 테이블을 참조하여 쓰기 요청된 주소가 NDP 커널 ID 0번에 대응하는 패킷으로 인정하고 해당 요청을 NDP 회로(300)로 전송한다.

[0153] 요청 디코더(310)는 NDP 커널 테이블과 마이크로 컨텍스트 테이블을 참조하여 전달된 요청을 디코딩하여 요청 버퍼(320)에 저장한다.

[0154] NDP 커널 테이블에서 코드 위치를 참조하여 해당 NDP 커널 ID에 대응하는 인스트럭션을 인스트럭션 캐시(350)에서 로드하여 인스트럭션 큐(331)와 요청 큐(332)에 저장한다.

[0155] 인스트럭션 큐(331)에 저장된 인스트럭션은 연산 회로(340)에 전달되어 누적 동작이 수행되고, 요청 큐(332)에 저장된 쓰기 요청은 메모리 컨트롤러(220)에 제공된다.

[0156] 첫 번째 쓰기 요청 패킷이 처리되는 경우 마이크로 컨텍스트 테이블에서 0번 NDP 커널 ID 및 0번 피벗 인덱스에 대응하는 나머지 패킷 개수는 1 감소하여 3으로 설정된다.

[0157] 같은 방식으로 0번 마이크로 컨텍스트에 대한 두 번째 및 세 번째 쓰기 요청 패킷이 처리될 수 있다.

[0158] 두 번째 쓰기 요청 패킷에 대응하는 쓰기 주소는 0x040, 세 번째 쓰기 요청 패킷에 대응하는 쓰기 주소는 0x080인 것으로 가정한다.

[0159] 두 번째 쓰기 요청 패킷이 처리되는 경우 마이크로 컨텍스트 테이블에서 0번 NDP 커널 ID 및 0번 피벗 인덱스에 대응하는 나머지 패킷 개수는 1 감소하여 2로 설정된다.

[0160] 세 번째 쓰기 요청 패킷이 처리되는 경우 마이크로 컨텍스트 테이블에서 0번 NDP 커널 ID 및 0번 피벗 인덱스에 대응하는 나머지 패킷 개수는 1 감소하여 1로 설정된다.

[0161] 마지막으로 0번 마이크로 컨텍스트에 대한 네 번째 쓰기 요청 패킷이 처리될 수 있으며 이때 쓰기 주소는 0x0C0인 것으로 가정한다.

[0162] 네 번째 쓰기 요청 패킷도 마찬가지 방식으로 처리될 수 있으며 마이크로 컨텍스트 테이블에서 0번 NDP 커널 ID 및 0번 피벗 인덱스에 대응하는 나머지 패킷 개수는 1 감소하여 0으로 설정된다.

[0163] 또한 NDP 커널 테이블에서 0번 NDP 커널 ID에 대응하는 나머지 마이크로 컨텍스트 개수는 1 감소하여 1로 설정된다.

[0164] 이후 1번 마이크로 컨텍스트에 대응하는 4개의 쓰기 요청 패킷이 유사한 방식으로 처리될 수 있다.

[0165] 각각의 쓰기 요청 패킷에 대응하여 도 9(B)의 요청당 함수 연산 동작이 수행되어 최종적으로 텐서 데이터의 행 벡터들을 이용한 연산 결과가 2개의 정적 레지스터에 저장된다.

[0166] 도 9(B)의 코드에 포함된 정적 레지스터 번호는 논리적인 번호를 나타낸다.

[0167] 프로그램이 컴파일되는 경우 정적 레지스터의 논리 번호와 마이크로 컨텍스트 테이블의 정적 레지스터 베이스 ID를 참조하여 정적 레지스터의 물리 번호로 변환될 수 있으며 이러한 동작은 레지스터 주소 변환 회로(360)에서 수행될 수 있다.

[0168] 예를 들어 피벗 인덱스 0에 대응하는 NDP 커널 동작을 수행할 때 도 9의 레지스터(v0, v1)는 물리적인 레지스터(v0, v1)를 나타낸다고 가정하면, 피벗 인덱스 1에 대응하는 NDP 커널 동작을 수행할 때 도 9의 레지스터(v0, v1)는 물리적인 레지스터(v2, v3)를 나타낸다.

[0169] 본 실시예에서 도 9(C)의 완료 동작 코드는 마이크로 컨텍스트당 한 번씩 수행되며 4개의 요청 패킷이 전송된 후 수행된다.

[0170] 이에 따라 도 9(C)의 완료 동작 코드는 하위 행 벡터에 대한 계산 값을 이용하여 하위 행 벡터에 대응하는 평균과 표준 편차를 계산하고 이를 원격 메모리 장치(120)의 지정된 주소에 저장하는 동작을 수행한다.

[0171] 또한 도 9(C)의 완료 동작 코드는 상위 행 벡터에 대한 계산 값을 이용하여 상위 행 벡터에 대응하는 평균과 표준 편차를 계산하고 이를 원격 메모리 장치(120)의 지정된 주소에 저장하는 동작을 수행한다.

[0172] 도 11은 본 발명의 다른 실시예에 의한 가속기 시스템(2000)을 나타내는 블록도이다.

[0173] 본 실시예에서는 그래픽 처리 장치(10-1)에서 NDP 기능을 수행한다.

[0174] 그래픽 처리 장치(10-1)에서 수행하는 NDP 기능과 메모리 확장 장치(100)에서 수행하는 NDP 기능은 서로 독립적이다.

[0175] 이에 따라 본 실시예에서 메모리 확장 장치(100)는 도 1의 실시예와 같이 NDP 기능을 수행할 수 있다. 이 경우 GPU(400)에서 생성된 NDP 요청을 메모리 확장 장치(100)에서 처리하는 것은 전술한 바와 같으므로 설명을 생략한다.

[0176] 그래픽 처리 장치(10-1)는 GPU(400)와 로컬 메모리 장치(500)를 포함하며 본 실시예에서 GPU(400)는 NDP 기능을 함께 수행한다.

[0177] 도 12는 그래픽 처리 장치(10-1)를 구체적으로 나타낸 블록도이다.

[0178] GPU(400)는 다수의 단위 프로세서(410), 다수의 GPU 확장 제어 회로(430) 및 다수의 단위 프로세서(410)와 다수의 호스트 확장 제어 회로(430)를 연결하는 인터커넥트 네트워크(420)를 포함한다. 이하에서, GPU 확장 제어 회로(430)를 호스트 확장 제어 회로(430)로 지칭할 수 있다.

[0179] 단위 프로세서(410)는 일반적으로 GPU에 포함되는 하위 프로세서로서 예를 들어 스트리밍 멀티프로세서(SM)가 본 실시예의 단위 연산 회로(420)에 대응할 수 있다.

[0180] 인터커넥트 네트워크(420)는 다수의 단위 프로세서(410)와 다수의 GPU 확장 제어 회로(430)를 완전 연결 방식으로 연결할 수 있다.

[0181] 도 13은 GPU 확장 제어 회로(430)와 로컬 메모리 장치(500)를 구체적으로 나타낸 블록도이다.

[0182] GPU 확장 제어 회로(430)는 도 6의 확장 제어 회로(110)에 대응하며 세부 구성 및 기능도 이와 유사하다.

[0183] 즉, GPU 확장 제어 회로(430)는 인터페이스 회로(113), DMA 회로(114), 다수의 GPU NDP 요청 제어 회로(600)를 포함한다. 이하에서 GPU NDP 요청 제어 회로(600)를 호스트 NDP 요청 제어 회로(600)로 지칭할 수 있다.

[0184] 이들은 각각 도 6의 인터페이스 회로(111), DMA 회로(114) 및 다수의 NDP 요청 제어 회로(200)에 대응한다. 대응하는 구성의 동작은 실질적으로 동일하다.

[0185] 로컬 메모리 장치(500)는 다수의 단위 로컬 메모리 장치(501)를 포함한다.

[0186] 본 실시예에서 하나의 GPU NDP 요청 제어 회로(600)는 하나의 단위 로컬 메모리 장치(501)와 연결된다.

[0187] GPU NDP 요청 제어 회로(600)는 대응하는 단위 로컬 메모리 장치(501)에 따라 주소 범위가 지정되며 인터페이스 회로(113)는 입력된 요청 패킷의 주소를 판단하여 대응하는 GPU NDP 요청 제어 회로(600)에 전달할 수 있다.

[0188] 인터페이스 회로(113)는 다수의 GPU NDP 요청 제어 회로(600)와 인터커넥트 네트워크(420) 사이에서 패킷을 송수신한다.

[0189] DMA 회로(114)는 DMA 기술을 통해 요청 패킷을 생성할 수 있으며 인터페이스 회로(113)에 연결될 수 있다.

[0190] **예를 들어 DMA 회로(114)에서 생성되는 요청 패킷은 단위 프로세서(410)에서 생성되는 요청 패킷과 동일한 형태를 가질 수 있다.**

[0191] **이에 따라 하나의 GPU 확장 제어 회로(430)에서 생성된 요청은 내부에서 처리될 수도 있고, 다른 GPU 확장 제어 회로(430) 또는 다른 메모리 확장 장치(100)로 전송될 수도 있다.**

[0192] GPU NDP 요청 제어 회로(600)는 인터페이스 회로(113)와 단위 로컬 메모리 장치(120) 사이에 연결되어 메모리 읽기 쓰기 동작 및 NDP 연산 동작을 수행한다.

[0193] GPU NDP 요청 제어 회로(600)는 도 6의 NDP 요청 제어 회로(200)에 대응하며 구성 및 동작이 실질적으로 동일하다.

[0194] GPU NDP 요청 제어 회로(600)는 필터 회로(610), NDP 회로(700), 및 메모리 컨트롤러(620)를 포함하며 이들은 도 6의 필터 회로(210), NDP 회로(300), 및 메모리 컨트롤러(220)에 대응한다. 대응하는 구성의 동작은 실질적으로 동일한다.

[0195] 이에 따라 NDP 회로(700)는 도 7 내지 도 10을 이용하여 개시한 것과 동일한 세부 구성을 가지고 동작 방식도 동일다.

[0196] 이에 따라 GPU NDP 요청 제어 회로(600)의 세부 구성 및 동작에 대해서는 개시를 생략한다.

[0197] 일반적으로 GPU는 그 내부에 캐시 메모리를 포함한다. 이를 위하여 본 실시예에서는 GPU NDP 요청 제어 회로(600)의 필터 회로(610)와 메모리 컨트롤러(620) 사이에 캐시 메모리 및 캐시 제어 회로(630)가 더 포함된다.

[0198] 이에 따라, 필터 회로(610)에서 필터링된 단순 요청은 상위 레벨의 캐시 메모리(630)와 로우 레벨의 단위 로컬 메모리 장치(501)로 구성되는 메모리 계층 구조에 따라 처리될 수 있다.

[0199] 이러한 메모리 계층 구조에서 읽기 쓰기 요청을 처리하는 것은 통상의 기술자에게 잘 알려진 것으로서 구체적인 설명은 생략한다.

[0200] 본 실시예에서 단위 프로세서(410)는 NDP 요청을 생성하고 이를 GPU 확장 제어 회로(430)와 로컬 메모리 장치(500)를 이용하여 처리할 수 있다.

[0201] GPU 확장 제어 회로(430)에서 NDP 요청을 처리하는 방식 자체는 메모리 확장 장치(100)에서 확장 제어 회로(110)를 이용하여 NDP 요청을 처리하는 방식과 동일하므로 구체적인 설명은 생략한다.

[0202] 도 14는 본 발명의 다른 실시예에 의한 그래픽 처리 장치(10-1)와 메모리 확장 장치(100)에 대한 제어 과정을 나타낸 설명도이다.

[0203] 도 14는 그래픽 처리 장치(10-1)와 메모리 확장 장치(100) 모두에서 NDP 기능을 수행하는 실시예에 대응한다.

[0204] 도 14에서 컴파일러(2-1)는 GPU(400) 내부에서 수행하는 NDP 기능을 추가로 지원하는 점에서 도 3과 차이가 있으며, 이를 위하여 컴파일러(2-1)는 그래픽 처리 장치(10-1)에서 수행하는 GPU 커널(3-1)과 GPU NDP 커널(3-

2)을 생성한다.

[0205] **그래픽 처리 장치(10-1)에서 GPU 커널(3-1)을 수행하는 중에 그래픽 처리 장치(10-1) 또는 메모리 확장 장치(100)에 대한 읽기 또는 쓰기 요청이 발생할 수 있고, 각 요청에 대응하여 그래픽 처리 장치(10-1) 또는 메모리 확장 장치(100)는 대응하는 GPU NDP 커널(3-2) 또는 NDP 커널(4)을 수행할 수 있다.**

[0206] **GPU 커널(3-1)에서 발생한 요청과 이에 대응하는 GPU NDP 커널(3-2) 또는 NDP 커널(4)은 컴파일러(2-1)에서 미리 결정될 수 있다.**

[0207] 도 15는 그래픽 처리 장치(10-1)에서 수행되는 심층 신경망 연산 과정을 나타낸 설명도이다.

[0208] 도 15의 설명도는 도 5의 설명도에 대응한다.

[0209] 즉, 도 15의 단위 프로세서(410), NDP 회로(700), 단위 로컬 메모리 장치(501)는 도 5의 GPU(11), NDP 회로(300), 원격 메모리 장치(120)에 대응한다.

[0210] 동작의 주체를 제외한 동작 방식은 도 5에 개시된 것과 실질적으로 동일하므로 반복적인 설명은 생략한다.

[0211] 본 발명의 권리범위는 이상의 개시로 한정되는 것은 아니다. 본 발명의 권리범위는 청구범위에 문언적으로 기재된 범위와 그 균등범위를 기준으로 해석되어야 한다.

부호의 설명

[0212] 1000: 가속기 시스템

10, 10-1: 호스트 장치, 그래픽 처리 장치

11, 400: 프로세서, GPU

410: 단위 프로세서

12, 500: 로컬 메모리 장치, 그래픽 메모리 장치

20, 420: 인터커넥트 네트워크

100: 메모리 확장 장치

110: 확장 제어 회로

430: 호스트 확장 제어 회로

120: 원격 메모리 장치, 확장 메모리 장치

111, 113: 인터페이스 회로

112, 114: DMA

200: NDP 요청 제어 회로

600: 호스트 NDP 요청 제어 회로

210, 610: 필터 회로

220, 620: 메모리 컨트롤러

630: 캐시 메모리

300, 700: NDP 회로

310: 요청 디코더

320: 요청 버퍼

330: 인스트럭션 저장 회로

331: 인스트럭션 큐

332: 요청 큐

340: 연산 회로

350: 인스트럭션 캐시

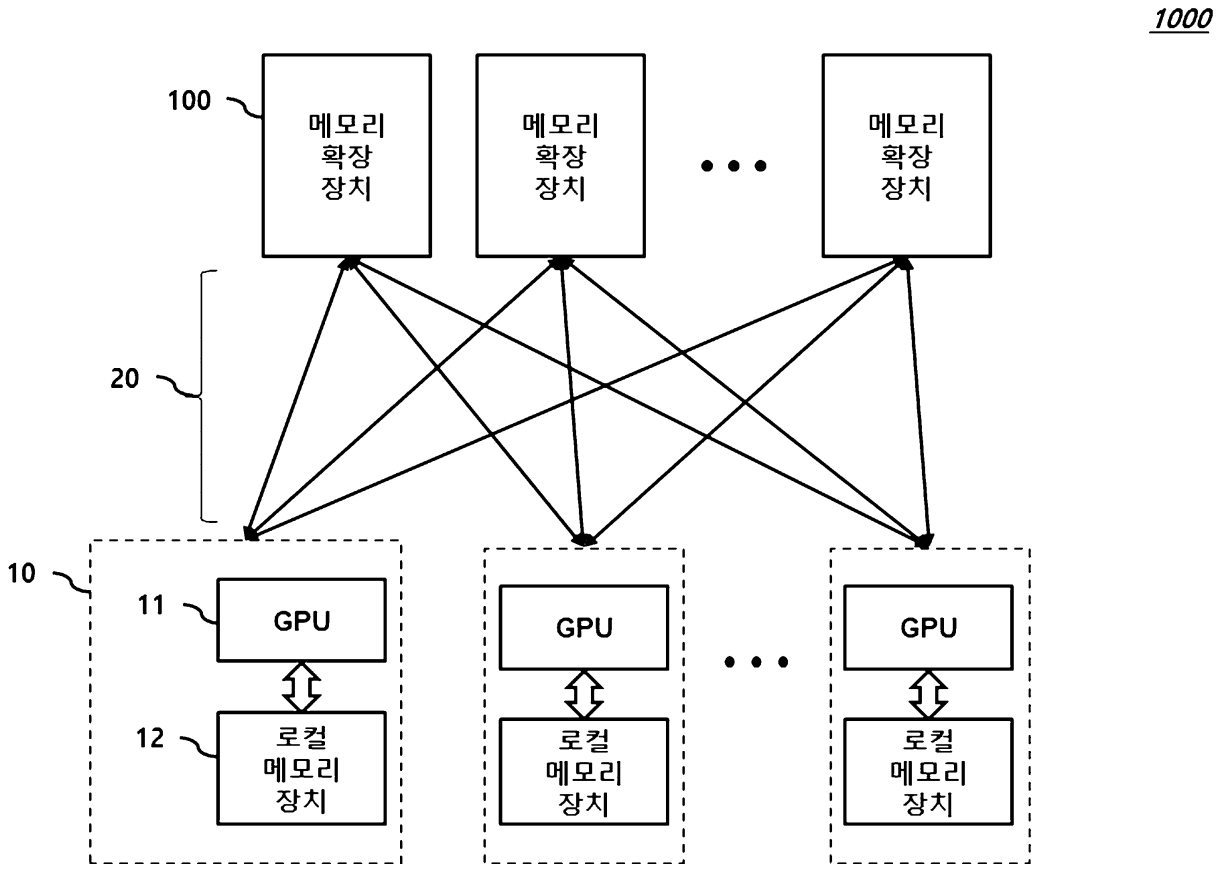
360: 레지스터 주소 변환 회로

370: 레지스터 파일

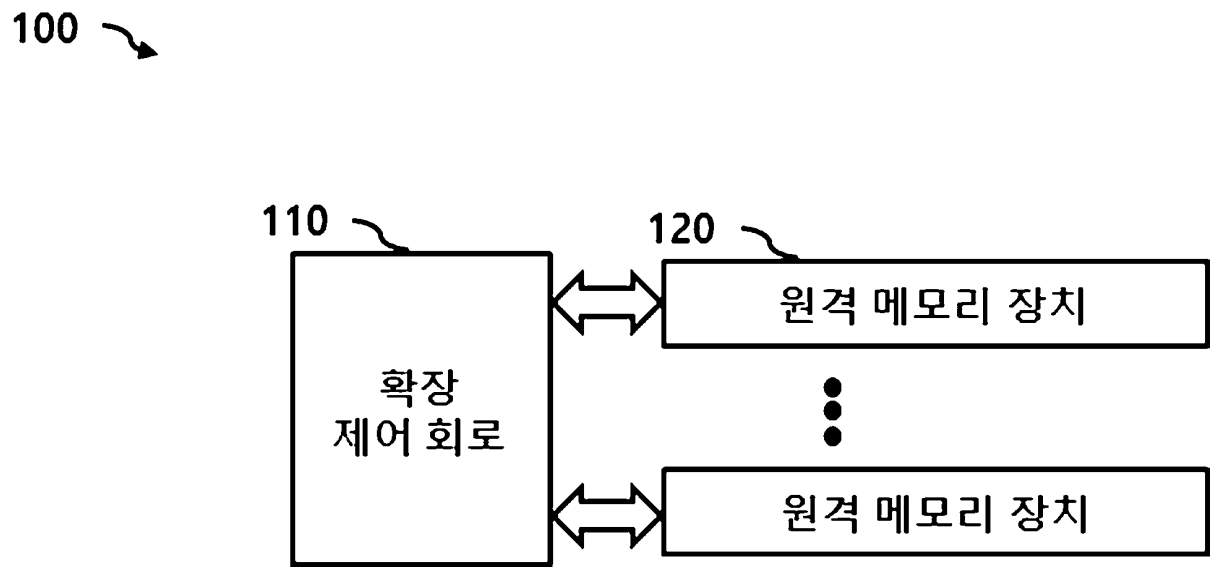
380: 마이크로 컨텍스트 저장 회로

도면

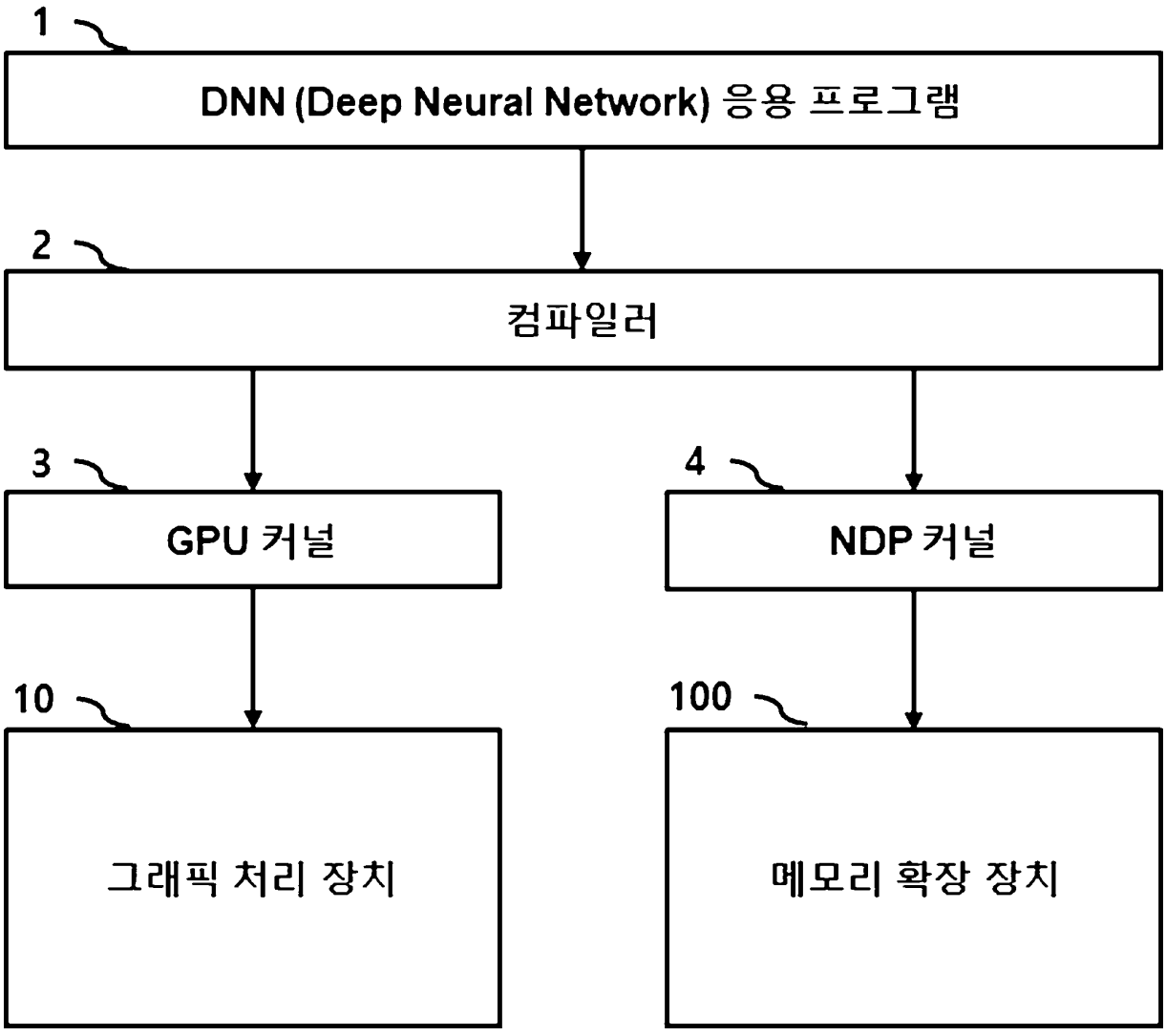
도면1



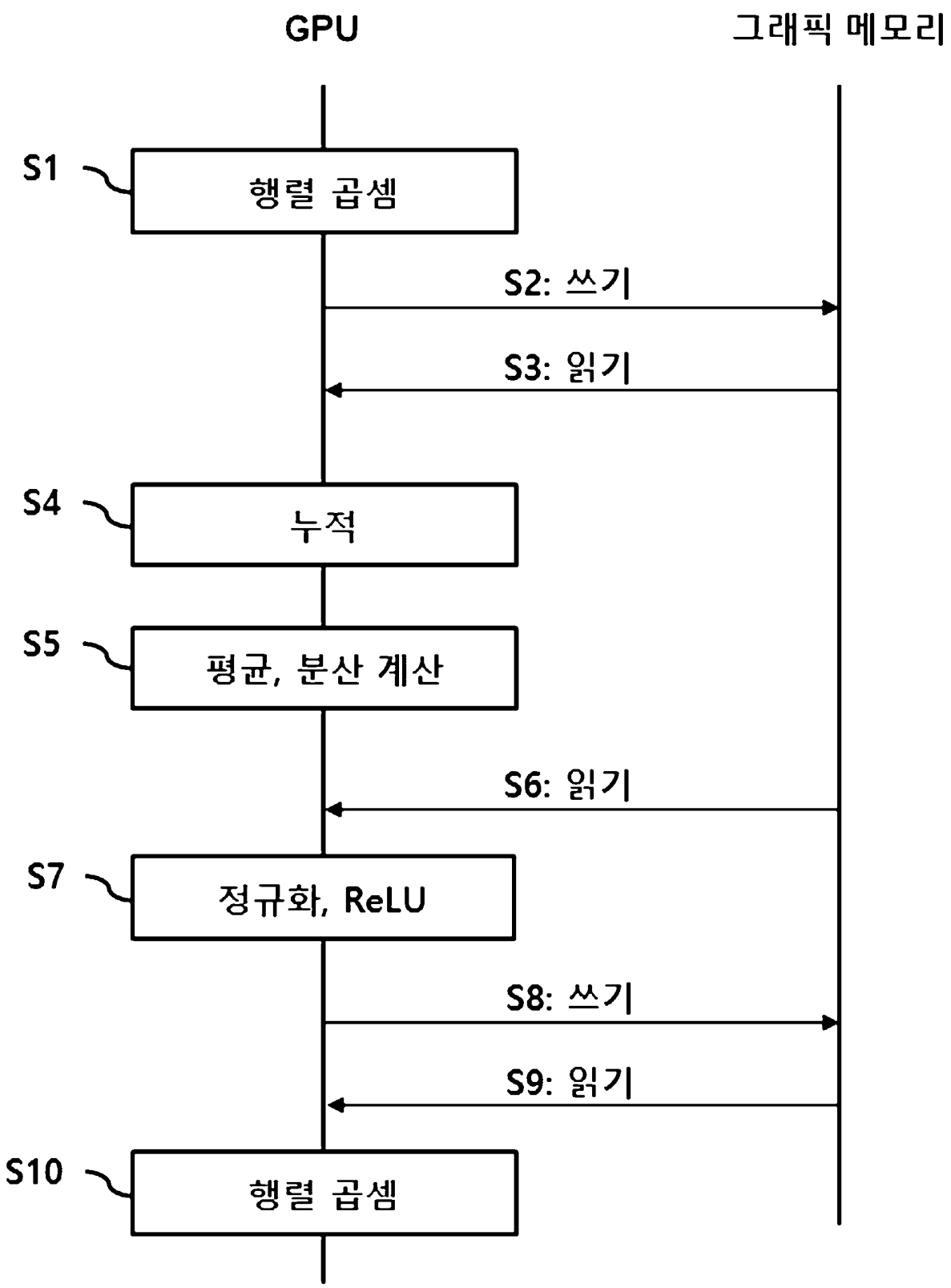
도면2


100
110
120
확장
제어 회로
원격 메모리 장치
원격 메모리 장치

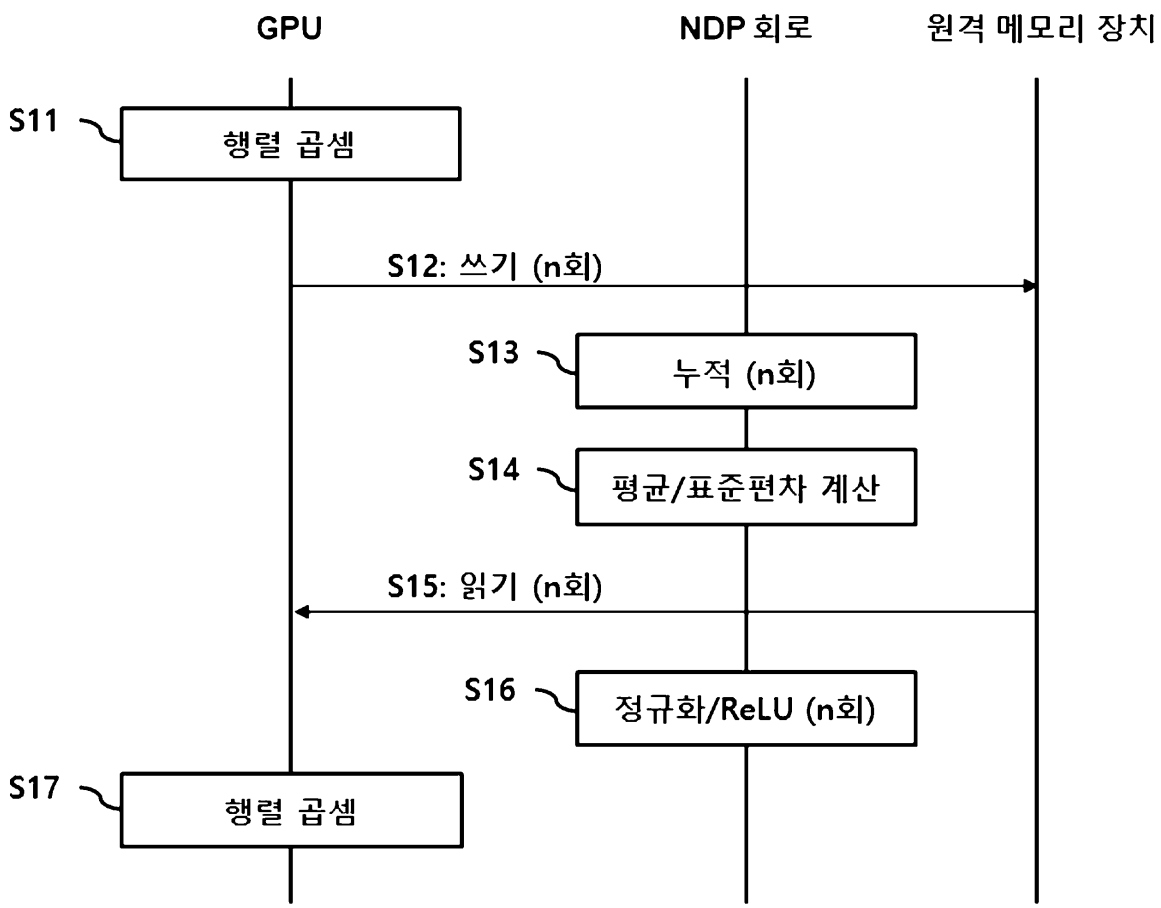
도면3


1
DNN (Deep Neural Network) 응용 프로그램
2
컴파일러
3
GPU 커널
4
NDP 커널
10
그래픽 처리 장치
100
메모리 확장 장치

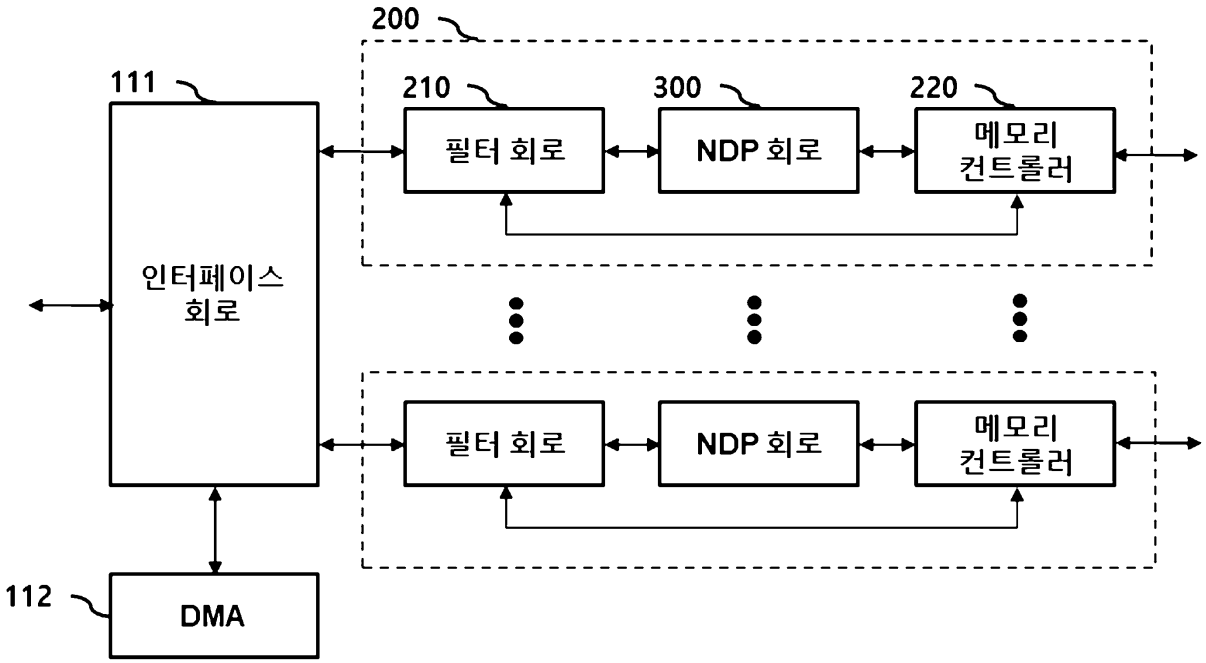
도면4


GPU
그래픽 메모리
S1
행렬 곱셈
S2: 쓰기
S3: 읽기
S4
누적
S5
평균, 분산 계산
S6: 읽기
S7
정규화, ReLU
S8: 쓰기
S9: 읽기
S10
행렬 곱셈

도면5

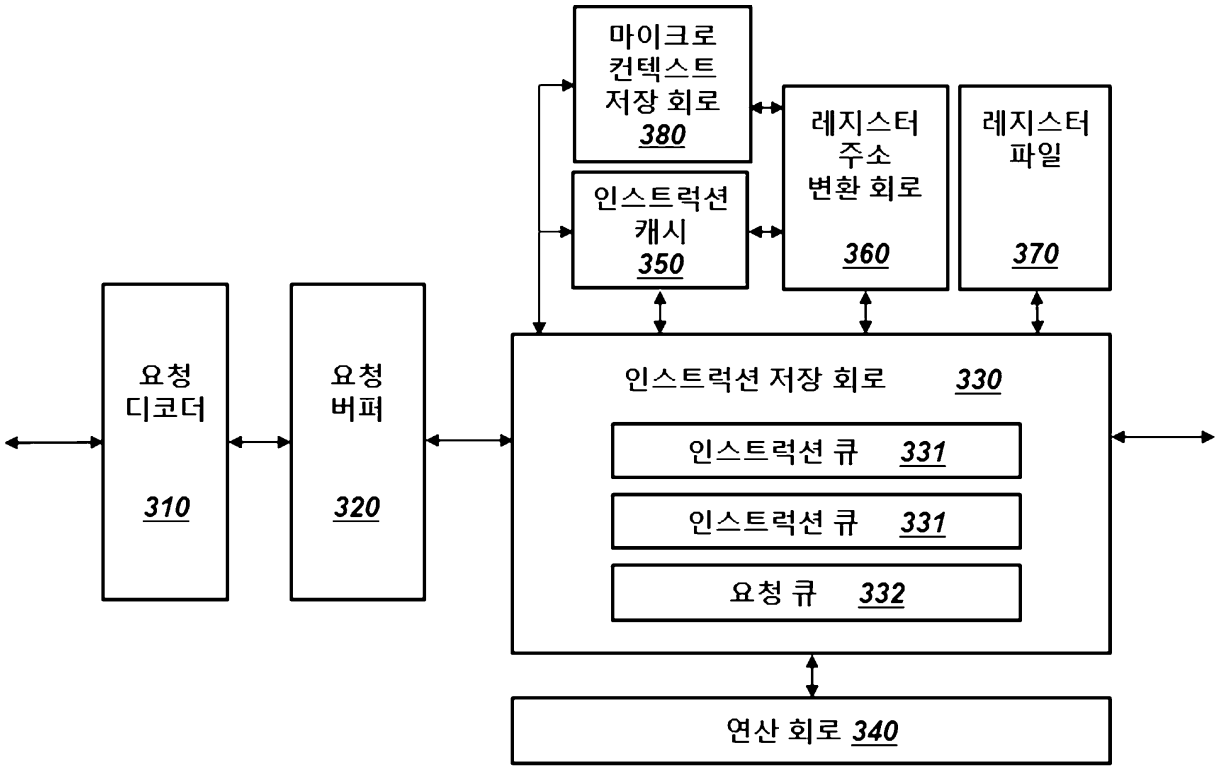

GPU
NDP 회로
원격 메모리 장치
S11
행렬 곱셈
S12: 쓰기 (n회)
S13
누적 (n회)
S14
평균/표준편차 계산
S15: 읽기 (n회)
S16
정규화/ReLU (n회)
S17
행렬 곱셈

도면6


200
111
210
300
220
필터 회로
NDP 회로
메모리
컨트롤러
인터페이스
회로
필터 회로
NDP 회로
메모리
컨트롤러
112
DMA

110

도면7



300

도면8

(A)

베이스 주소	주소 바운드	피벗 차원	텐서 형태	NDP 커널 ID	필터 인자

(B)

NDP 커널 ID	코드위치	정적 레지스터 개수	동적 레지스터 개수	마이크로 컨텍스트 당 요청 개수	나머지 마이크로 컨텍스트 개수

(C)

NDP 커널 ID	피벗 인덱스	정적 레지스터 베이스 ID	나머지 패킷 개수

도면9

(A)

```
// initialize reg.
VLI v0 0
VLI v1 0
```

(B)

```
// load req's data,x
VLS v2 REQDATA

// accumulate x, x2
VADD v0 v0 v2
VFMA v1 v1 v2 v2

// store req's data, x
LS r1 REQADDR
VST v2 0(r1)
```

(C)

```
LS r1 FILTERARG
// divide by # of x's
VMULI v0 v0 0.25
VMULI v1 v1 0.25

// store μ
VST v0 0(r1)

VMUL v0 v0 v0
VSUB v1 v1 v0
VSQRT v1 v1

// store σ
VST v1 0x400(r1)
```

도면10

(A)

베이스 주소	주소 바운드	피벗 차원	텐서 형태	NDP 커널 ID	필터 인자
0x000	0x100	0	(4, 32)	0	0x200

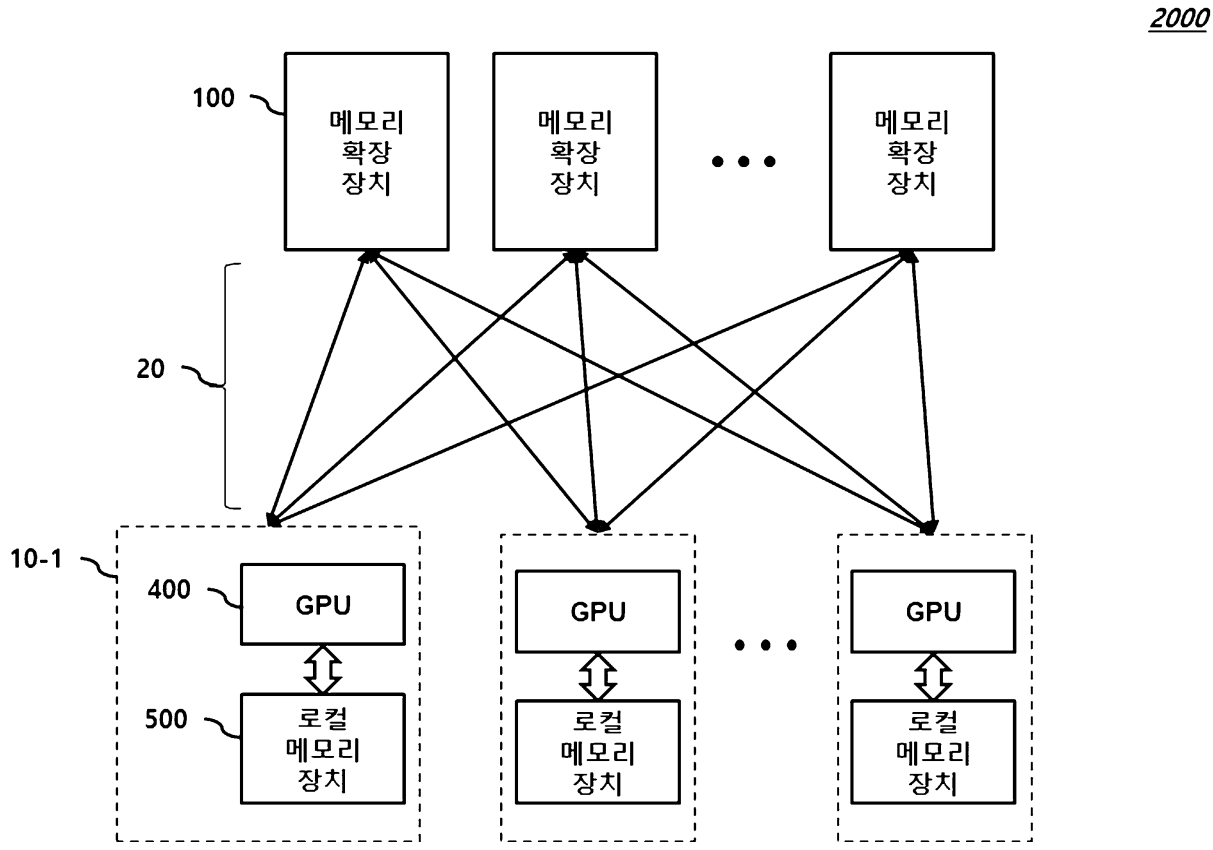
(B)

NDP 커널 ID	코드위치	정적 레지스터 개수	동적 레지스터 개수	마이크로 컨텍스트 당 요청 개수	나머지 마이크로 컨텍스트 개수
0	0x300	2	2	4	2

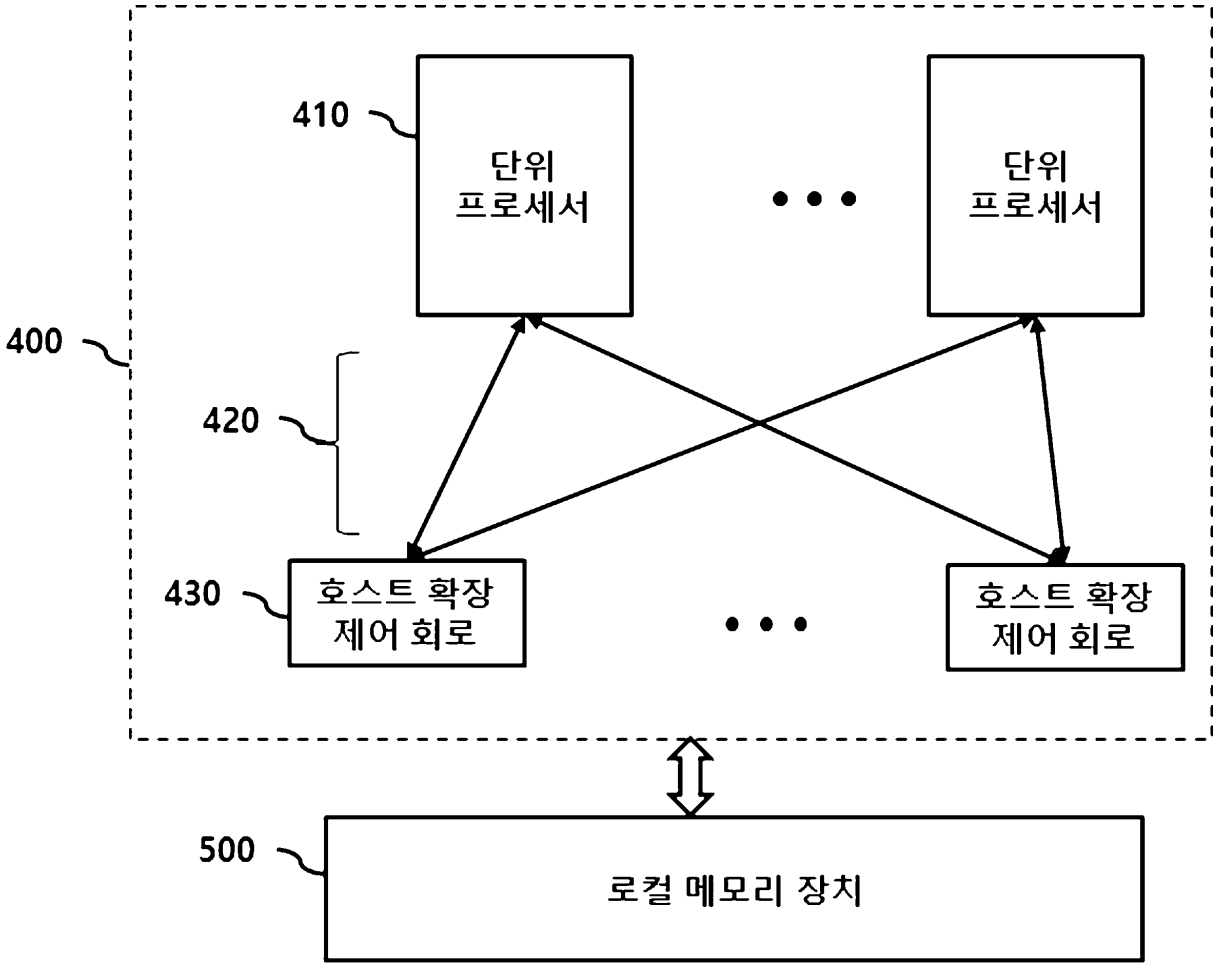
(C)

NDP 커널 ID	피벗 인덱스	정적 레지스터 베이스 ID	나머지 패킷 개수
0	0	0	4
0	1	2	4

도면11

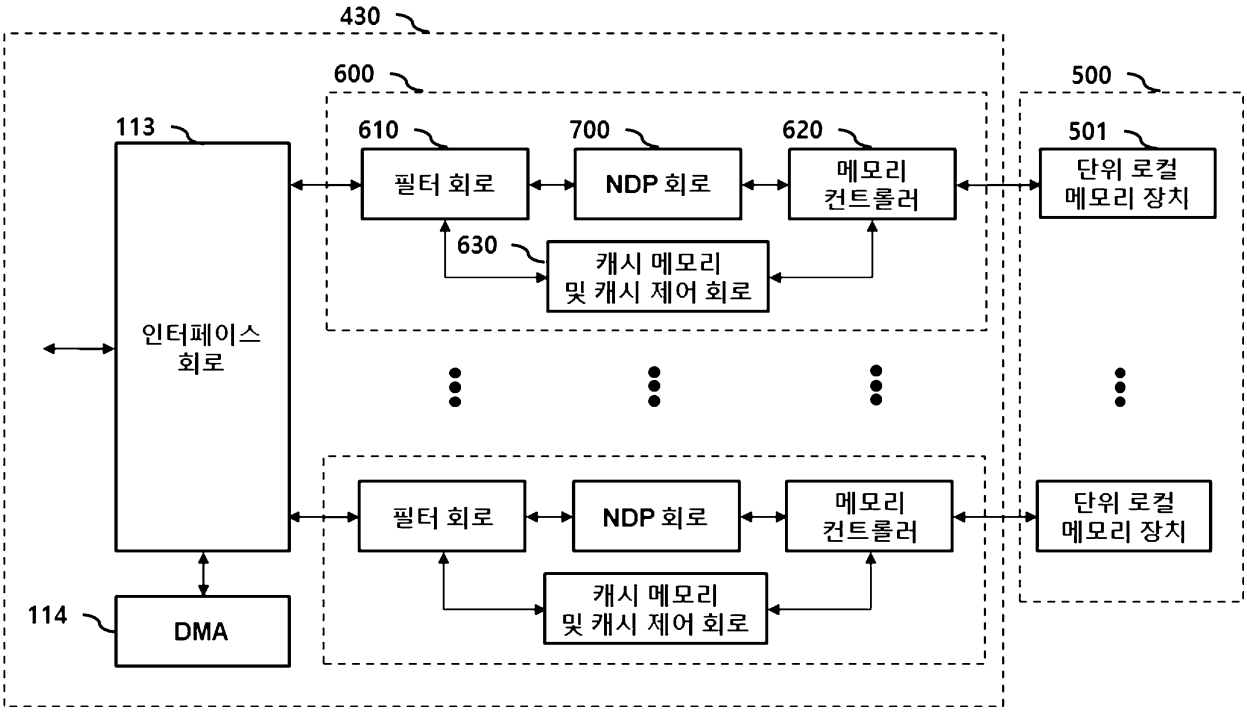


도면12

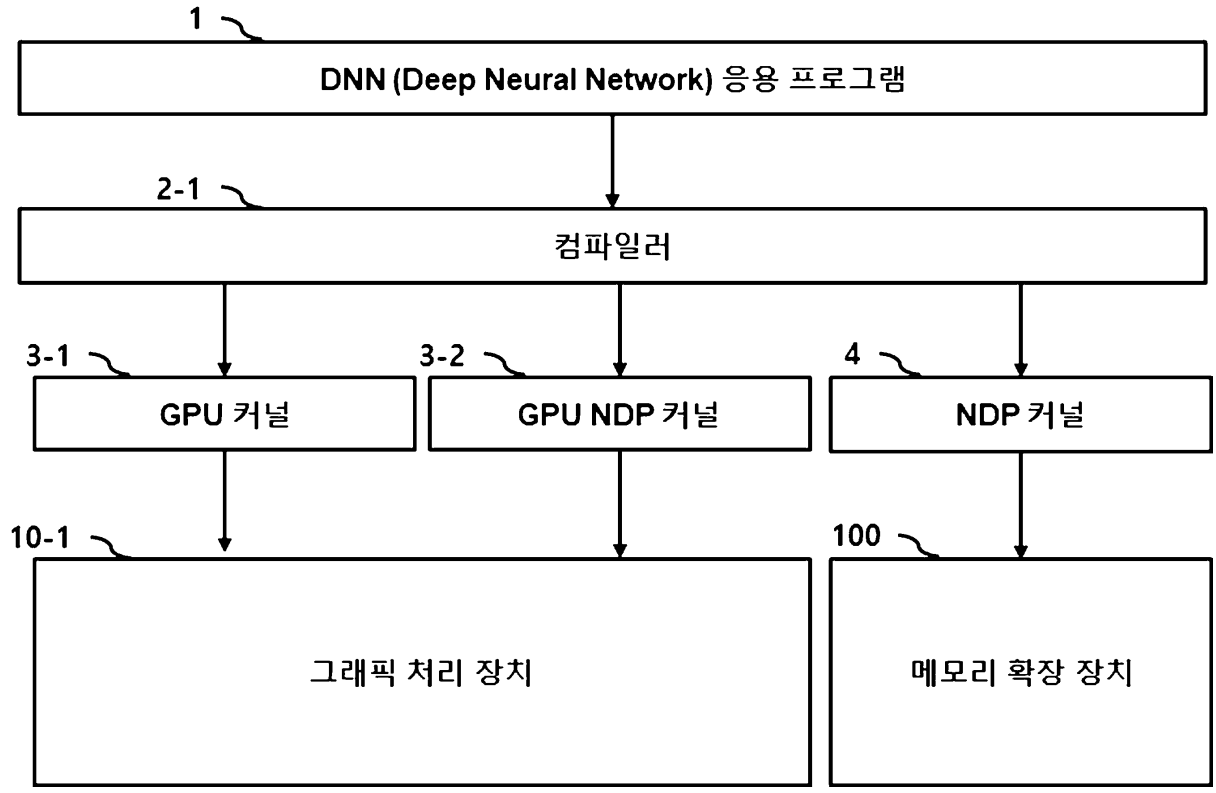

410
단위
프로세서
단위
프로세서
400
420
430
호스트 확장
제어 회로
호스트 확장
제어 회로
500
로컬 메모리 장치

10-1

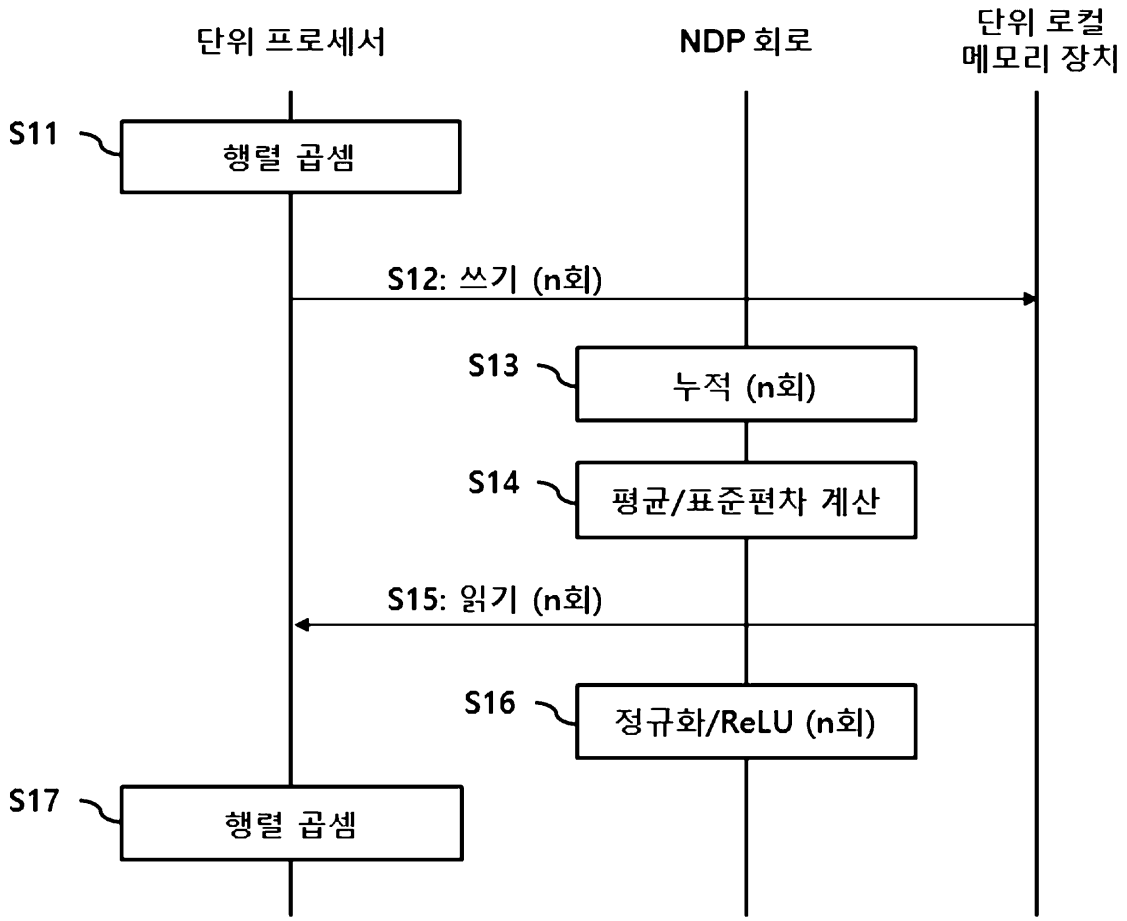
도면13


430
600
500
113
610
700
620
501
필터 회로
NDP 회로
메모리
컨트롤러
단위 로컬
메모리 장치
630
캐시 메모리
및 캐시 제어 회로
인터페이스
회로
필터 회로
NDP 회로
메모리
컨트롤러
단위 로컬
메모리 장치
캐시 메모리
및 캐시 제어 회로
114
DMA

도면14


1
DNN (Deep Neural Network) 응용 프로그램
2-1
컴파일러
3-1
3-2
4
GPU 커널
GPU NDP 커널
NDP 커널
10-1
100
그래픽 처리 장치
메모리 확장 장치

도면15


단위 프로세서
NDP 회로
단위 로컬
메모리 장치
S11
행렬 곱셈
S12: 쓰기 (n회)
S13
누적 (n회)
S14
평균/표준편차 계산
S15: 읽기 (n회)
S16
정규화/ReLU (n회)
S17
행렬 곱셈