



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2025년04월14일
(11) 등록번호 10-2795207
(24) 등록일자 2025년04월08일

(51) 국제특허분류(Int. Cl.)
H04L 69/10 (2022.01) G06F 15/173 (2006.01)
H04L 69/12 (2022.01) H04L 69/16 (2022.01)
H04L 69/22 (2022.01)
(52) CPC특허분류
H04L 69/10 (2022.05)
G06F 15/17331 (2013.01)
(21) 출원번호 10-2023-0179404
(22) 출원일자 2023년12월12일
심사청구일자 2023년12월12일
(56) 선행기술조사문헌
KR1020150146391 A
KR1020050011191 A

(73) 특허권자
울산과학기술원
울산광역시 울주군 언양읍 유니스트길 50
(72) 발명자
임영빈
울산광역시 울주군 언양읍 유니스트길 50
김현수
울산광역시 울주군 언양읍 유니스트길 50
(74) 대리인
특허법인 무한

전체 청구항 수 : 총 20 항

심사관 : 황철규

(54) 발명의 명칭 smart NIC을 통해 TCP 프로토콜을 가속화하는 장치 및 방법

(57) 요약

일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 애플리케이션 버퍼(application buffer)로부터 전달된 애플리케이션 데이터(application data)가 저장되는 애플리케이션 RDMA(Remote Direct Memory Access) 메모리 영역을 포함하는 애플리케이션을 실행하고, 프리 리스트(Free list)와 해시 테이블(Hash (뒷면에 계속))

대표도

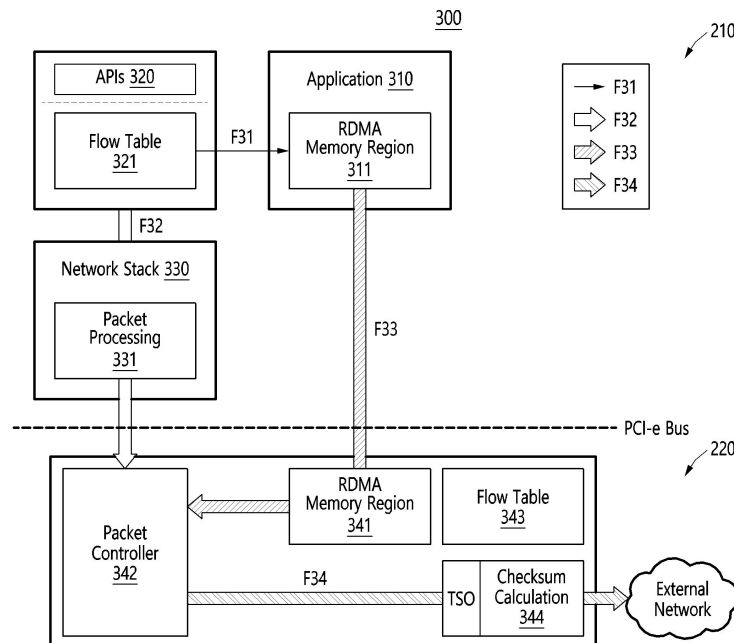


Table)을 포함하는 호스트 플로우 테이블(Flow Table)을 내부 메모리인 호스트 메모리(Host Memory) 내에 저장하여 상기 애플리케이션의 RDMA 메모리 영역의 애플리케이션 데이터를 관리하는 API(Application Programming Interface)들을 실행하며, 상기 API들로부터 전송된 헤더를 처리하는 호스트 CPU(Host Central Processing Unit) 및 상기 애플리케이션의 애플리케이션 RDMA 메모리 영역에서 읽어(read) 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역에 저장하고, 저장된 애플리케이션 데이터와 상기 호스트 CPU로부터 수신된 처리된 헤더에 기초하여 패킷을 형성하는 스마트 닉을 포함할 수 있다.

(52) CPC특허분류

H04L 69/12 (2022.05)

H04L 69/16 (2022.05)

H04L 69/22 (2022.05)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711193910
과제번호	2021-0-01817-003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보통신방송혁신인재양성
연구과제명	하이퍼-컴포저블 데이터센터를 위한 차세대 컴퓨팅 기술 개발
기 여 율	50/100
과제수행기관명	울산과학기술원
연구기간	2023.01.01 ~ 2023.12.31

이 발명을 지원한 국가연구개발사업

과제고유번호	1711184969
과제번호	2021R1F1A1061346
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	개인기초연구(과기정통부)
연구과제명	세부 성능 정보 측정을 통한 분산화, 가상화된 시스템의 성능 향상 연구
기 여 율	50/100
과제수행기관명	울산과학기술원
연구기간	2023.03.01 ~ 2024.02.29

명세서

청구범위

청구항 1

스마트 닉(Smart network interface controller, Smart NIC)을 통해 TCP 프로토콜을 가속화하는 장치에 있어서,

애플리케이션 버퍼(application buffer)로부터 전달된 애플리케이션 데이터(application data)가 저장되는 애플리케이션 RDMA(Remote Direct Memory Access) 메모리 영역을 포함하는 애플리케이션을 실행하고, 프리 리스트(Free list)와 해시 테이블(Hash Table)을 포함하는 호스트 플로우 테이블(Flow Table)을 내부 메모리인 호스트 메모리(Host Memory) 내에 저장하여 상기 애플리케이션의 RDMA 메모리 영역의 애플리케이션 데이터를 관리하는 API(Application Programming Interface)들을 실행하며, 상기 API들로부터 전송된 헤더를 처리하는 호스트 CPU(Host Central Processing Unit); 및

상기 애플리케이션의 애플리케이션 RDMA 메모리 영역에서 읽어(read) 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역에 저장하고, 저장된 애플리케이션 데이터와 상기 호스트 CPU로부터 수신된 처리된 헤더에 기초하여 패킷을 형성하는 스마트 닉

을 포함하는 장치.

청구항 2

제1항에 있어서,

상기 애플리케이션 RDMA 메모리 영역은,

고정된 크기로 형성된 복수의 메모리 블록(Memory Block)들을 포함하고,

상기 프리 리스트는,

상기 복수의 메모리 블록들 중 플로우에 할당되지 않은 블록들의 각각이 프리 리스트에서 다른 메모리 블록을 포인팅(point)하도록 생성되며,

상기 호스트 CPU는,

상기 프리 리스트를 이용하여 상기 플로우가 할당되지 않은 블록들을 관리하고, 각 플로우 별로 하나 이상의 블록을 할당하는,

장치.

청구항 3

제1항에 있어서,

상기 스마트 닉 RDMA 메모리 영역은,

고정된 크기로 형성된 복수의 메모리 블록들을 포함하고,

상기 스마트 닉은,

상기 복수의 메모리 블록들 중 플로우에 할당되지 않은 블록들의 각각이 다른 메모리 블록을 포인팅하도록 생성된 프리 리스트와 해시 테이블을 포함하는 스마트 닉 플로우 테이블을 내부 메모리인 스마트 닉 메모리 내에 저장하고, 상기 프리 리스트를 이용하여 상기 플로우가 할당되지 않은 블록들을 관리하며, 각 플로우 별로 하나 이상의 블록을 할당하는,

장치.

청구항 4

제1항에 있어서,

상기 호스트 CPU는,

고정된 크기의 어레이(array)인 각 플로우에 대한 정보를 포함하는 플로우 리스트(Flow List)를 이용하여 다수의 플로우들에 대응하는 메모리 블록에 대해 접근하기 위한 정보를 상기 해시 테이블에 기록하는,

장치.

청구항 5

제4항에 있어서,

상기 플로우 리스트는,

링크드 리스트(linked list) 형식으로 구성되어 충돌(Collision)을 방지하도록 사용되는 해시 평션(Hash Function) 중 두번째 해시 평션 정보, 블록 내에서 애플리케이션 데이터를 쓰기 시작하는 지점을 나타내는 스타트 포인트(Start Point) 정보, 플로우에 할당된 메모리 블록에 접근하기 위한 블록 인덱스 번호(Block Index) 정보, 상기 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트(Start Byte)와 엔드 바이트(End Byte) 정보, 하나의 어레이의 블록 인덱스 번호 저장 영역의 용량이 초과되는 경우 새로 할당된 어레이인 다음 플로우 리스트를 포인팅하는 넥스트 포인터(Next Pointer) 정보, 전체 애플리케이션 데이터 크기인 토탈 애플리케이션 바이트(Total App Byte) 정보 및 RDMA로 복사된 전체 데이터 크기인 토탈 RDMA 바이트(Total RDMA Byte) 정보

를 포함하는 장치.

청구항 6

제1항에 있어서,

상기 스마트 닉은,

미리 결정된 주기마다 상기 호스트 메모리 내 플로우 정보(flow information, flow info)에서 상기 애플리케이션 RDMA 메모리 영역으로부터 읽을(read) 애플리케이션 데이터가 있는지 확인하고 상기 호스트 메모리 내 플로우 별 읽기(read) 정보를 포함하는 호스트 RDMA 메모리 영역에서 상기 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어오며, 읽어 온 상기 읽기 정보에 기초하여 플로우 테이블을 구성하며, 상기 읽을 애플리케이션 데이터의 주소를 계산하여 읽어 온 상기 애플리케이션 데이터를 상기 스마트 닉 RDMA 메모리 영역으로 저장하는,

장치.

청구항 7

제6항에 있어서,

상기 플로우 정보는 수신할 데이터의 블록 인덱스 번호 정보를 포함하고,

상기 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그(Flag) 정보, TCP 플로우 관리를 위한 ISN(Initial Sequence Number) 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평션 정보, 상기 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보, 및 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함하는,

장치.

청구항 8

제1항에 있어서,

상기 호스트 CPU는,

상기 API들을 이용하여 상기 애플리케이션 RDMA 메모리 영역에서의 데이터 쓰기(data write)와 데이터 읽기(data read)를 시간차를 두고 수행하는,

장치.

청구항 9

제1항에 있어서,

상기 호스트 CPU는,

상기 애플리케이션 버퍼에 애플리케이션 데이터가 저장된 애플리케이션을 실행하고, 상기 애플리케이션 버퍼로부터 전달된 상기 애플리케이션 데이터가 저장되는 API들 RDMA 메모리 영역 및 상기 API들 RDMA 메모리 영역의 애플리케이션 데이터를 관리하기 위하여 프리 리스트를 관리하고 해시 테이블을 포함한 호스트 플로우 테이블을 포함하는 API들을 실행하며, 상기 API들로부터 전송된 헤더를 처리하고,

상기 스마트 닉은,

상기 API들의 API들 RDMA 메모리 영역에서 읽어 온 애플리케이션 데이터를 상기 스마트 닉 RDMA 메모리 영역에 저장하고 저장된 애플리케이션 데이터와 상기 호스트 CPU로부터 수신된 처리된 헤더에 기초하여 패킷을 형성하는,

장치.

청구항 10

제9항에 있어서,

상기 호스트 CPU는 상기 스마트 닉으로 통지 메시지(notification message) 및 읽기 메시지(Read message)를 전송하고,

상기 스마트 닉은 상기 읽기 메시지에 기초하여 상기 API들 RDMA 메모리 영역의 상기 애플리케이션 데이터를 읽어 오는,

장치.

청구항 11

제10항에 있어서,

상기 통지 메시지는 ISN 및 해시 평선 정보를 포함하고,

상기 스마트 닉은 상기 ISN 및 해시 평선 정보에 기초하여 스마트 닉 플로우 테이블을 생성하는,

장치.

청구항 12

제10항에 있어서,

상기 읽기 메시지는 전송할 데이터에 대한 주소인 토탈 플로우 넘버(Total Flow Number), 데이터 크기인 렱스 (Length), 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보를 포함하고,

상기 스마트 닉은 상기 토탈 플로우 넘버, 렱스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보에 기초하여 요청된 모든 애플리케이션 데이터를 상기 API들 RDMA 메모리 영역으로부터 한 번에 읽어서 상기 스마트 닉 RDMA 메모리 영역에 저장하는,

장치.

청구항 13

제12항에 있어서,

상기 스마트 닉은 상기 스마트 닉 RDMA 메모리 영역에 상기 애플리케이션 데이터가 저장된 후 상기 호스트 CPU에 읽기 완료 신호를 송신하고,

상기 호스트 CPU는 상기 API들 RDMA 메모리 영역 내 플래그 공간에 읽기 완료에 대한 쓰기를 수행하는,

장치.

청구항 14

스마트 닉에서 TCP 프로토콜을 가속화하는 방법에 있어서,

호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 단계;

상기 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역으로 저장하는 단계;

상기 호스트 CPU로부터 처리된 헤더를 수신하는 단계; 및

상기 스마트 닉 RDMA 메모리 영역에 저장된 상기 애플리케이션 데이터와 상기 처리된 헤더에 기초하여 패킷을 형성하는 단계

를 포함하는 방법.

청구항 15

제14항에 있어서,

상기 호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 단계는,

미리 결정된 주기마다 호스트 메모리 내 플로우 정보에 기초하여 애플리케이션 RDMA 메모리 영역으로부터 읽을 애플리케이션 데이터가 있는지 확인하는 단계;

상기 호스트 메모리 내 플로우 별 읽기 정보를 포함하는 호스트 RDMA 메모리 영역에서 상기 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어 오는 단계;

읽어 온 상기 읽기 정보에 기초하여 플로우 테이블을 구성하는 단계; 및

상기 읽을 애플리케이션 데이터의 주소를 계산하여 상기 애플리케이션 데이터를 읽어 오는 단계

를 포함하는 방법.

청구항 16

제15항에 있어서,

상기 플로우 정보는 읽을 데이터의 블록 인덱스 번호 정보를 포함하고,

상기 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그 정보, TCP 플로우 관리를 위한 ISN 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평선 정보, 상기 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보, 및 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함하는,

방법.

청구항 17

제14항에 있어서,

상기 호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 단계는,

상기 호스트 CPU로부터 수신된 통지 메시지에 기초하여 스마트 닉 플로우 테이블을 생성하는 단계; 및

상기 호스트 CPU로부터 수신된 읽기 메시지에 기초하여 API들 RDMA 메모리 영역에서의 애플리케이션 데이터를 읽어 오는 단계

를 포함하는 방법.

청구항 18

제17항에 있어서,

상기 통지 메시지는 ISN 및 해시 평선 정보를 포함하고,

상기 읽기 메시지는 전송할 데이터에 대한 주소인 토탈 플로우 넘버, 데이터 크기인 렱스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보를 포함하는,

방법.

청구항 19

제17항에 있어서,

상기 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역으로 저장하는 단계는,

상기 읽기 메시지에 기초하여 요청된 모든 애플리케이션 데이터를 상기 호스트 CPU의 API들 RDMA 메모리 영역에서 한 번에 읽어서 상기 스마트 닉 RDMA 메모리 영역에 저장하는 단계

를 포함하는 방법.

청구항 20

제14항에 있어서,

상기 애플리케이션 데이터를 통지 메시지 및 읽기 메시지에 기초하여 수신하는 경우, 상기 호스트 CPU가 API들 RDMA 메모리 영역 내 플래그 공간에 읽기 완료에 대한 쓰기를 수행하도록 상기 스마트 닉 RDMA 메모리 영역에 상기 애플리케이션 데이터를 저장한 후 상기 호스트 CPU에 읽기 완료 신호를 송신하는 단계

를 더 포함하는 방법.

발명의 설명

기술 분야

[0001] 이하, 호스트(Host) CPU가 처리하던 데이터를 스마트 닉(Smart network interface controller, Smart NIC)이 처리하도록 이동시켜 호스트 CPU에 대한 부담을 줄이는 기술이 제공된다.

배경 기술

[0003] 컴퓨터는 네트워크를 연결하기 위하여 하드웨어인 네트워크 인터페이스 컨트롤러(network interface controller, NIC)를 설치할 수 있다. 네트워크 인터페이스 컨트롤러(NIC)는 OSI 계층 1(물리 계층)과 계층 2(데이터 링크 계층)로 구성되어 계층에 따라 역할과 기능을 분배할 수 있다. 예를 들면, 네트워크 인터페이스 컨트롤러(NIC)는 계층 1을 통해 신호를 전송하고 계층 2에서 패킷을 전송할 수 있다. 그러나 이러한 표준적인 네트워크 인터페이스 컨트롤러(NIC)의 구조는 네트워크를 이해하고 개발하기에는 쉽지만 성능 구현이 제한되는 단점이 있을 수 있다.

[0004] 한편, 컴퓨터는 기술이 발달할수록 처리할 자원이 많아짐에 따라 한정된 컴퓨팅 자원의 한계를 극복하기 위하여 오프로딩(offloading)이라는 방식을 이용하고 있다. 오프로딩은 연산 능력과 저장 공간, 전력 등이 제한적인 장치(예: 코걸 컴퓨터, 모바일 장치)에서 수행하는 애플리케이션의 일부를 컴퓨팅 자원과 처리능력이 우수한 플랫폼(예: 원격지 컴퓨터)에 전달하여 처리한 후 결과를 반환받는 방식일 수 있다. 이러한 추세에 따라 전술한 단점을 극복하고자 현대 네트워크에서는 전송 계층의 기능을 연결 계층에 오프로딩하는 방식을 사용하고 있으며, 이는 오프로딩을 하지 않는 방식보다 뛰어난 성능을 보여주고 있다. 예를 들면, 전송 계층의 기능은 체크섬 계산, TCP Segmentation Offload(TSO) 등일 수 있고, 연결 계층은 네트워크 인터페이스 컨트롤러(NIC)일 수 있다. 그러나 이러한 오프로딩이 가능한 네트워크 인터페이스 컨트롤러(NIC)는 Application-Specific Integrated Circuit(ASIC)기반으로 개발되어 양산이 용이하고 가격이 저렴하다는 장점이 있으나 내부 로직을 변경하는 것이 불가능하고, 오프로딩 기능 또한 제한적인 단점이 있다.

[0005] 전술한 단점을 극복하기 위하여 최근에는 스마트 네트워크 인터페이스 카드(Smart NIC)이 개발되어 사용되고 있다. 스마트 네트워크 인터페이스 카드(Smart NIC)은 ASIC 뿐만 아니라 ARM, FPGA, SoC(system-on-chip) 및 이를 융합한 기술을 기반으로 개발되어 오프로딩이 가능한 네트워크 인터페이스 컨트롤러(NIC)보다 내부 로직을 자유롭게 구현할 수 있다. 이에 따라 스마트 네트워크 인터페이스 카드(Smart NIC)은 다양한 오프로딩이 가능할 수 있다.

[0006] 위에서 설명한 배경기술은 발명자가 본원의 개시 내용을 도출하는 과정에서 보유하거나 습득한 것으로서, 반드시 본 출원 전에 일반 공중에 공개된 공지기술이라고 할 수는 없다.

선행기술문헌

특허문헌

[0008] (특허문헌 0001) 한국 공개 특허공보 10-2023-0111512(2023년 07월 25일 공고)에는 클라우드 컴퓨팅 환경에서의 스마트 닉 기반 패킷 처리 가속화 및 그것의 운영 자동화 시스템이 제시된다.

(특허문헌 0002) 일본 공개 특허공보 2022-166934(2022년 11월 04일 공고)에는 정보처리 장치, 과부하 제어 프로그램 및 과부하 제어 방법이 제시된다.

발명의 내용

해결하려는 과제

[0009] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 애플리케이션 데이터의 헤더 부분을 호스트 CPU에서 처리하고, 애플리케이션 데이터의 데이터 부분을 스마트 닉에서 처리할 수 있다.

[0010] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 네트워크 구조에 있어서 애플리케이션이 애플리케이션 RDMA 메모리 영역을 포함할 수 있다.

[0011] 다만, 기술적 과제는 상술한 기술적 과제들로 한정되는 것은 아니며, 또 다른 기술적 과제들이 존재할 수 있다.

과제의 해결 수단

- [0012] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치는 애플리케이션 버퍼(application buffer)로부터 전달된 애플리케이션 데이터(application data)가 저장되는 애플리케이션 RDMA(Remote Direct Memory Access) 메모리 영역을 포함하는 애플리케이션을 실행하고, 프리 리스트(Free list)와 해시 테이블(Hash Table)을 포함하는 호스트 플로우 테이블(Flow Table)을 내부 메모리인 호스트 메모리(Host Memory) 내에 저장하여 상기 애플리케이션의 RDMA 메모리 영역의 애플리케이션 데이터를 관리하는 API(Application Programming Interface)들을 실행하며, 상기 API들로부터 전송된 헤더를 처리하는 호스트 CPU(Host Central Processing Unit) 및 상기 애플리케이션의 애플리케이션 RDMA 메모리 영역에서 읽어(read) 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역에 저장하고, 저장된 애플리케이션 데이터와 상기 호스트 CPU로부터 수신된 처리된 헤더에 기초하여 패킷을 형성하는 스마트 닉을 포함할 수 있다.
- [0013] 상기 애플리케이션 RDMA 메모리 영역은, 고정된 크기로 형성된 복수의 메모리 블록(Memory Block)들을 포함하고, 상기 프리 리스트는, 상기 복수의 메모리 블록들 중 플로우에 할당되지 않은 블록들의 각각이 프리 리스트에서 다른 메모리 블록을 포인팅(point)하도록 생성되며, 상기 호스트 CPU는, 상기 프리 리스트를 이용하여 상기 플로우가 할당되지 않은 블록들을 관리하고, 각 플로우 별로 하나 이상의 블록을 할당할 수 있다.
- [0014] 상기 스마트 닉 RDMA 메모리 영역은, 고정된 크기로 형성된 복수의 메모리 블록들을 포함하고, 상기 스마트 닉은, 상기 복수의 메모리 블록들 중 플로우에 할당되지 않은 블록들의 각각이 다른 메모리 블록을 포인팅하도록 생성된 프리 리스트와 해시 테이블을 포함하는 스마트 닉 플로우 테이블을 내부 메모리인 스마트 닉 메모리 내에 저장하고, 상기 프리 리스트를 이용하여 상기 플로우가 할당되지 않은 블록들을 관리하며, 각 플로우 별로 하나 이상의 블록을 할당할 수 있다.
- [0015] 상기 호스트 CPU는, 고정된 크기의 어레이(array)인 각 플로우에 대한 정보를 포함하는 플로우 리스트(Flow List)를 이용하여 다수의 플로우들에 대응하는 메모리 블록에 대해 접근하기 위한 정보를 상기 해시 테이블에 기록할 수 있다.
- [0016] 상기 플로우 리스트는, 링크드 리스트(linked list) 형식으로 구성되어 충돌(Collision)을 방지하도록 사용되는 해시 평선(Hash Function) 중 두번째 해시 평선 정보, 블록 내에서 애플리케이션 데이터를 쓰기 시작하는 지점을 나타내는 스타트 포인트(Start Point) 정보, 플로우에 할당된 메모리 블록에 접근하기 위한 블록 인덱스 번호(Block Index) 정보, 상기 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트(Start Byte)와 엔드 바이트(End Byte) 정보, 하나의 어레이의 블록 인덱스 번호 저장 영역의 용량이 초과되는 경우 새로 할당된 어레이인 다음 플로우 리스트를 포인팅하는 넥스트 포인터(Next Pointer) 정보, 전체 애플리케이션 데이터 크기인 토탈 애플리케이션 바이트(Total App Byte) 정보 및 RDMA로 복사된 전체 데이터 크기인 토탈 RDMA 바이트(Total RDMA Byte) 정보를 포함할 수 있다.
- [0017] 상기 스마트 닉은, 미리 결정된 주기마다 상기 호스트 메모리 내 플로우 정보(flow information, flow info)에서 상기 애플리케이션 RDMA 메모리 영역으로부터 읽을(read) 애플리케이션 데이터가 있는지 확인하고 상기 호스트 메모리 내 플로우 별 읽기(read) 정보를 포함하는 호스트 RDMA 메모리 영역에서 상기 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어오며, 읽어 온 상기 읽기 정보에 기초하여 플로우 테이블을 구성하며, 상기 읽을 애플리케이션 데이터의 주소를 계산하여 읽어 온 상기 애플리케이션 데이터를 상기 스마트 닉 RDMA 메모리 영역으로 저장할 수 있다.
- [0018] 상기 플로우 정보는 수신할 데이터의 블록 인덱스 번호 정보를 포함하고, 상기 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그(Flag) 정보, TCP 플로우 관리를 위한 ISN 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평선 정보, 상기 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보, 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0019] 상기 호스트 CPU는, 상기 API들을 이용하여 상기 애플리케이션 RDMA 메모리 영역에서의 데이터 쓰기(data write)와 데이터 읽기(data read)를 시간차 두고 수행할 수 있다.
- [0020] 상기 호스트 CPU는, 상기 애플리케이션 버퍼에 애플리케이션 데이터가 저장된 애플리케이션을 실행하고, 상기 애플리케이션 버퍼로부터 전달된 상기 애플리케이션 데이터가 저장되는 API들 RDMA 메모리 영역 및 상기 API들 RDMA 메모리 영역의 애플리케이션 데이터를 관리하기 위하여 프리 리스트와 해시 테이블을 포함한 호스트 플로우 테이블을 포함하는 API들을 실행하며, 상기 API들로부터 전송된 헤더를 처리하고, 상기 스마트 닉은, 상기 API들의 API들 RDMA 메모리 영역에서 읽어 온 애플리케이션 데이터를 상기 스마트 닉 RDMA 메모리 영역에 저장하고 저장된 애플리케이션 데이터와 상기 호스트 CPU로부터 수신된 처리된 헤더에 기초하여 패킷을 형성할 수

있다.

- [0021] 상기 호스트 CPU는 상기 스마트 닉으로 통지 메시지(notification message) 및 읽기 메시지(Read message)를 전송하고, 상기 스마트 닉은 상기 읽기 메시지에 기초하여 상기 API들 RDMA 메모리 영역의 상기 애플리케이션 데이터를 읽어 올 수 있다.
- [0022] 상기 통지 메시지는 ISN 및 해시 평선 정보를 포함하고, 상기 스마트 닉은 상기 ISN 및 해시 평선 정보에 기초하여 스마트 닉 플로우 테이블을 생성할 수 있다.
- [0023] 상기 읽기 메시지는 전송할 데이터에 대한 주소인 토탈 플로우 넘버(Total Flow Number), 데이터 크기인 렉스(Length), 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보를 포함하고, 상기 스마트 닉은 상기 토탈 플로우 넘버, 렉스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보에 기초하여 요청된 모든 애플리케이션 데이터를 상기 API들 RDMA 메모리 영역으로부터 한 번에 읽어서 상기 스마트 닉 RDMA 메모리 영역에 저장할 수 있다.
- [0024] 상기 스마트 닉은 상기 스마트 닉 RDMA 메모리 영역에 상기 애플리케이션 데이터가 저장된 후 상기 호스트 CPU에 읽기 완료 신호를 송신하고, 상기 호스트 CPU는 상기 API들 RDMA 메모리 영역 내 플래그 공간에 읽기 완료에 대한 쓰기를 수행할 수 있다.
- [0025] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 방법은 호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 단계, 상기 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역으로 저장하는 단계, 상기 호스트 CPU로부터 처리된 헤더를 수신하는 단계 및 상기 스마트 닉 RDMA 메모리 영역에 저장된 상기 애플리케이션 데이터와 상기 처리된 헤더에 기초하여 패킷을 형성하는 단계를 포함할 수 있다.
- [0026] 상기 애플리케이션 데이터를 수신하는 단계는, 미리 결정된 주기마다 호스트 메모리 내 플로우 정보에 기초하여 애플리케이션 RDMA 메모리 영역으로부터 읽을 애플리케이션 데이터가 있는지 확인하는 단계, 상기 호스트 메모리 내 플로우 별 읽기 정보를 포함하는 호스트 RDMA 메모리 영역에서 상기 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어 오는 단계, 읽어 온 상기 읽기 정보에 기초하여 플로우 테이블을 구성하는 단계 및 상기 읽을 애플리케이션 데이터의 주소를 계산하여 상기 애플리케이션 데이터를 읽어 오는 단계를 포함할 수 있다.
- [0027] 상기 플로우 정보는 읽을 데이터의 블록 인덱스 번호 정보를 포함하고, 상기 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그 정보, TCP 플로우 관리를 위한 ISN 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평선 정보, 상기 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보, 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0028] 상기 애플리케이션 데이터를 수신하는 단계는, 상기 호스트 CPU로부터 수신된 통지 메시지에 기초하여 스마트 닉 플로우 테이블을 생성하는 단계 및 상기 호스트 CPU로부터 수신된 읽기 메시지에 기초하여 API들 RDMA 메모리 영역에서의 애플리케이션 데이터를 읽어 오는 단계를 포함할 수 있다.
- [0029] 상기 통지 메시지는 ISN 및 해시 평선 정보를 포함하고, 상기 읽기 메시지는 전송할 데이터에 대한 주소인 토탈 플로우 넘버, 데이터 크기인 렉스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0030] 상기 스마트 닉 RDMA 메모리 영역으로 저장하는 단계는, 상기 읽기 메시지에 기초하여 요청된 모든 애플리케이션 데이터를 상기 호스트 CPU의 API들 RDMA 메모리 영역에서 한 번에 읽어서 상기 스마트 닉 RDMA 메모리 영역에 저장하는 단계를 포함할 수 있다.
- [0031] 상기 방법은 상기 애플리케이션 데이터를 통지 메시지 및 읽기 메시지에 기초하여 수신하는 경우, 상기 호스트 CPU가 API들 RDMA 메모리 영역 내 플래그 공간에 읽기 완료에 대한 쓰기를 수행하도록 상기 스마트 닉 RDMA 메모리 영역에 상기 애플리케이션 데이터를 저장한 후 상기 호스트 CPU에 읽기 완료 신호를 송신하는 단계를 더 포함할 수 있다.

발명의 효과

- [0033] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 애플리케이션 데이터의 헤더 부분을 호스트 CPU에서 처리하고, 애플리케이션 데이터의 데이터 부분을 스마트 닉에서 처리함에 따라 데이터 처리 속도가 증가하여 전체적인 네트워크 속도 또한 향상되고, 호스트 CPU가 애플리케이션 데이터 전체를 처리하는 것보다 호스트 CPU의 부하가 감소되며 이에 따른 다른 애플리케이션들에 대한 성능 또한 향상될 수 있다.

[0034] 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 네트워크 구조에 있어서 애플리케이션이 애플리케이션 RDMA 메모리 영역을 포함하여 애플리케이션 데이터를 애플리케이션 버퍼로부터 애플리케이션 RDMA 메모리 영역으로 전달하기 위한 별도의 복사(copy)를 수행하는 작업이 필요하지 않아 호스트 CPU의 부하를 감소시킬 수 있다.

도면의 간단한 설명

[0036] 도 1(a)는 딜레이 없이 호스트에서 스마트 닉으로의 패킷 전송 결과를 측정한 그래프를 나타내고, 도 1(b)는 10초의 딜레이가 있는 호스트에서 스마트 닉으로의 패킷 전송 결과를 측정한 그래프를 나타낸다.

도 2는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 블록도를 도시한다.

도 3은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 네트워크 구조를 예시적으로 도시한다.

도 4는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역의 구조를 예시적으로 도시한다.

도 5는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 플로우 리스트를 포함하는 해시 테이블의 구조를 예시적으로 도시한다.

도 6은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 새로운 플로우 리스트가 할당되는 경우를 예시적으로 도시한다.

도 7은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 빈 플로우 리스트에 이어지는 블록 인덱스 번호를 복사하는 경우를 예시적으로 도시한다.

도 8은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 스마트 닉의 읽기 동작을 위한 호스트 메모리와의 구조를 예시적으로 도시한다.

도 9는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 또 다른 네트워크 구조를 예시적으로 도시한다.

도 10은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 또 다른 네트워크 구조에 기초한 호스트 CPU가 스마트 닉으로 보내는 통지 메시지 및 읽기 메시지를 예시적으로 도시한다.

도 11은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 방법의 흐름도를 예시적으로 도시한다.

발명을 실시하기 위한 구체적인 내용

[0037] 실시예들에 대한 특정한 구조적 또는 기능적 설명들은 단지 예시를 위한 목적으로 개시된 것으로서, 다양한 형태로 변경되어 구현될 수 있다. 따라서, 실제 구현되는 형태는 개시된 특정 실시예로만 한정되는 것이 아니며, 본 명세서의 범위는 실시예들로 설명한 기술적 사상에 포함되는 변경, 균등물, 또는 대체물을 포함한다.

[0038] 제1 또는 제2 등의 용어를 다양한 구성요소들을 설명하는데 사용될 수 있지만, 이런 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 해석되어야 한다. 예를 들어, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소는 제1 구성요소로도 명명될 수 있다.

[0039] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다.

[0040] 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 설명된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이 존재함으로써 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.

[0041] 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.

- [0042] 이하, 실시예들을 첨부된 도면들을 참조하여 상세하게 설명한다. 첨부 도면을 참조하여 설명함에 있어, 도면 부호에 관계없이 동일한 구성 요소는 동일한 참조 부호를 부여하고, 이에 대한 중복되는 설명은 생략하기로 한다.
- [0043] 무어의 법칙(Moore's law)은 1965년 페어차일드(Fairchild)의 연구원으로 있던 고든 무어(Gordon Moore)가 마이크로칩(예: CPU)의 용량이 1~2년마다 2배가 될 것으로 예측하며 만든 법칙으로, 90년대는 무어의 법칙에 기초하여 기술 개발을 수행하였다. 그러나 근래에 들어서는 작은 기판에 더 많은 회로가 집적됨에 따라 복잡한 아키텍처와 이에 의한 열처리 문제로 인해 무어의 법칙이 한계를 맞이하며 CPU 성능의 향상 속도 또한 느려지고 있다. 반면에, 네트워크의 대역폭(bandwidth)과 데이터(data)의 성장 속도는 느려지지 않았으며 오히려 지속적으로 성장하고 있으므로, 이러한 데이터를 처리하는 기술의 지속적인 성장을 필요로 한다.
- [0044] 또한, 네트워크 기능(Network function)은 전용 하드웨어에서 소프트웨어(software)와 가상 어플라이언스(virtual appliances)로 전환됨에 따라 CPU의 부담이 더욱 커지게 되었다. 이러한 전환으로 제한적인 CPU 자원 내에서의 네트워크 처리량은 더욱 늘어남에 따라 애플리케이션(application)을 위한 자원은 상대적으로 줄어들게 되었다. 이에 현재 이러한 CPU의 부담을 줄이기 위하여 네트워크 기능을 오프로드(offload)하는 기술들이 활발히 연구되고 있다.
- [0045] 기본적인 네트워크 스택은 유저(user) 영역, 커널(kernel) 영역 및 디바이스(device) 영역으로 나뉘어져 있다. 유저(user) 영역 및 커널(kernel) 영역은 호스트(host)라 하여 CPU에서 업무를 처리함에 따라 디바이스(device) 영역을 담당하는 네트워크 인터페이스 카드(NIC)와 구분될 수 있다. 유저 영역의 애플리케이션에서 전송할 사용자 데이터(user data)가 생성되면 커널 영역을 거쳐 디바이스 영역의 NIC으로 사용자 데이터가 전달될 수 있다. 커널 영역은 복수의 레이어로 구성될 수 있는데, 간략하게 소켓 레이어, TCP 레이어, IP 레이어 및 드라이버 레이어로 구성되어 순차적으로 각 레이어에 데이터가 통과하면서 전달될 수 있다.
- [0046] RDMA(Remote Direct Memory Access)는, 고속 네트워크를 통한 메모리 간 데이터 전송용 기술로, NIC(network interface card)에서 오프로드 기능을 수행하기 위하여 커널(kernel) 영역 내 CPU를 사용하지 않고 직접 원격 데이터를 전송할 수 있다. 그러나 이러한 RDMA 기술 또한 지속적으로 증가하는 처리해야 될 데이터 양에 대응하여 최적화하는 방안이 필요하다.
- [0047] 도 1(a)는 딜레이 없이 호스트에서 스마트 닉으로의 패킷 전송 결과를 측정한 그래프를 나타내고, 도 1(b)는 10초의 딜레이가 있는 호스트에서 스마트 닉으로의 패킷 전송 결과를 측정한 그래프를 나타낸다.
- [0048] 도 1에서, 일반적으로(110) 호스트에서 스마트 닉으로 전송되는 오브젝트의 크기가 커질수록(예: 도 1에서 우측으로) 초당 전송된 작은 사이즈(예를 들어, 헤더 크기의)의 패킷 개수가 점차 감소한다. 그러나, 오브젝트의 크기를 작은 크기(예를 들어, 12B)로(120) 쪼개서 전송되는 경우에는, 오브젝트의 크기와 관계없이 일정한 패킷 개수가 전송될 수 있다. 예를 들어, 도 1(a)와 같이 (예를 들어, 10초의) 딜레이가 있는 경우, 일반적으로(110) 전송되는 패킷 개수는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커질수록 각각 49620, 34071, 24151, 13489개로 점차 작아진다. 반면에, 작은 크기로(120) 전송되는 패킷 개수는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커져도 일정하게 92500개를 유지할 수 있다. 이는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커짐에 따라 약 2배, 약 3배, 약 4배, 약 7배의 패킷 개수의 차이를 가지는 것을 의미할 수 있다. 또한, 도 1(b)와 같이 딜레이가 없는 경우, 일반적으로(110) 전송되는 패킷 개수는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커질수록 각각 59690, 31010, 26041, 15756개로 점차 작아진다. 반면에, 작은 크기로(120) 전송되는 패킷 개수는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커져도 딜레이가 있는 경우보다 더 많은 25333개를 일정하게 유지할 수 있다. 이는 오브젝트의 크기가 4K, 8K, 16K, 32K로 커짐에 따라 약 4배, 약 8배, 약 10배, 약 16배의 패킷 개수의 차이를 가지는 것을 의미할 수 있다. 도 1의 결과에 따라, 데이터 전송은 일반 TCP 통신을 통하여 큰 크기의 데이터를 보내는 것보다 작은 크기의 데이터를 보내는 것이 더 효율적임을 알 수 있다. 이에 일 실시예에 따른 스마트 닉(Smart NIC, Smart Network Interface Card)을 통해 TCP 프로토콜을 가속화하는 장치는 RDMA를 이용하여 애플리케이션 데이터(application data)를 처리하고 나머지 헤더를 일반 TCP를 이용하여 처리함에 따라 훨씬 많은 패킷을 처리할 수 있다.
- [0049] 도 2는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 블록도를 도시한다.
- [0050] 도 3은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 네트워크 구조를 예시적으로 도시한다.

- [0051] 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치(예: 장치(200))는 호스트 CPU(Host Central Processing Unit)(210), 스마트 닉(Smart Network Interface Card, Smart NIC)(220) 호스트 메모리(Host Memory)(230)를 포함할 수 있다. 예시적으로, 호스트 CPU(210)는 호스트 메모리(230) 내에 저장된 컴퓨터로 읽을 수 있는 코드(예를 들어, 소프트웨어) 및 인스트럭션(instruction)들을 실행할 수 있는 프로세서(processor)일 수 있고, 호스트 메모리(230)는 RAM(random access memory)일 수 있다. 스마트 닉(220)은 오프로딩(off-load)된 동작(operation) 및/또는 기능(function)을 수행 가능한 내부 프로세서를 포함할 수 있다.
- [0052] 일 실시예에 따른 장치(200)에서의 네트워크 구조(300)는 호스트 CPU(210)가 구성하는 호스트 영역 내 애플리케이션(application)(310), API(Application Programming Interface)들(320) 및 네트워크 스택(Network Stack)(330)과 스마트 닉(220)이 구성하는 디바이스 영역을 포함할 수 있다.
- [0053] 호스트 CPU(210)는 애플리케이션 버퍼(application buffer)로부터 전달된 애플리케이션 데이터(application data)가 저장되는 애플리케이션 RDMA(Remote Direct Memory Access) 메모리 영역(311)을 포함하는 애플리케이션(310)을 실행하고, 프리 리스트(Free list)와 해시 테이블(Hash Table)을 포함하는 호스트 플로우 테이블(Flow Table)(321)을 내부 메모리인 호스트 메모리(Host Memory) 내에 저장하여 애플리케이션의 RDMA 메모리 영역(311)의 애플리케이션 데이터를 관리(F1)하는 API들(320)을 실행하며, API들(320)로부터 전송(F2)된 헤더를 처리할 수 있다. 이에 호스트 CPU(210)는 네트워크 구조에 있어서 애플리케이션(310)에 애플리케이션 RDMA 메모리 영역(311)이 포함됨에 따라 애플리케이션 데이터를 애플리케이션 버퍼로부터 애플리케이션 RDMA 메모리 영역(311)으로 전달하기 위한 별도의 복사(copy)를 수행하는 것 없이 바로 내려보냄에 따라 호스트 CPU(210)의 부하가 감소될 수 있다.
- [0054] 스마트 닉(220)은 애플리케이션으로부터 읽어 온(F3) 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역(341)에 저장하고, 저장된 애플리케이션 데이터와 호스트 CPU(210)로부터 수신된 처리된 헤더에 기초하여 패킷 컨트롤러(Packet Controller)(342)를 통해 패킷을 형성할 수 있다(F4). 형성된 패킷은 TSO(TCP Segmentaion Offload)(344)를 통해 패킷 사이즈를 조절하고, 계산된 체크섬(checksum)(344)을 포함하여 외부 장치 또는 서버에 송신될 수 있다. TSO(344)는 패킷을 여러 개로 분할하여 패킷 사이즈를 조절할 수 있다. 스마트 닉(220)은 분할된 패킷에 기초하여 TCP 시퀀스 넘버(sequence number)와 IP 렘스(length)등을 업데이트할 수 있다. 이에 호스트 CPU(210)는 애플리케이션 데이터의 헤더 부분을 호스트 CPU(210)에서 처리하고, 애플리케이션 데이터의 데이터 부분을 스마트 닉(220)에서 처리함에 따라 데이터 처리 속도가 증가하여 전체적인 네트워크 속도 또한 향상되고, 호스트 CPU가 애플리케이션 데이터 전체를 처리하는 것보다 호스트 CPU의 부하가 감소되며 이에 따른 다른 애플리케이션들에 대한 성능 또한 향상될 수 있다. 또한, 이전에 NVIDIA에서 스마트 닉을 사용함에 따라 주로 CPU 발열을 막기 위해 사용되는 에너지가 약29%로 감소되는 결과에 기초하여 더욱 효율적인 분산된 데이터 처리를 수행하는 일 실시예에 따른 호스트 CPU(210)와 스마트 닉(220)은 사용 에너지가 29%보다 더 절약됨과 동시에 탄소 배출량 또한 감소될 수 있다.
- [0055] 호스트 CPU(210)와 스마트 닉(220) 간의 데이터 송/수신은 PCI-e(Peripheral Component Interconnect-express) Bus를 이용하여 수행될 수 있다. PCI 버스는 컴퓨팅 시스템의 주변기기를 장착하기 위하여 정한 버스 규격일 수 있다. 또한, PCI-e 버스는 PCI 버스의 후속 인터페이스로, 두 쌍의 와이어로 구성된 하나 이상의 전송 라인으로 구성되어 PCI보다 두배 빠른 전송 속도를 가지고 있다.
- [0056] 호스트 CPU(210)는 애플리케이션(310)을 이용하여 애플리케이션 버퍼에 애플리케이션 데이터를 생성할(또는 쓸(write)) 수 있다. 호스트 CPU(210)는 애플리케이션 버퍼로부터 애플리케이션 RDMA 메모리 영역(311)으로 외부 장치로 전송할 애플리케이션 데이터를 저장할 수 있다. 애플리케이션 RDMA 메모리 영역(311)의 구조는 스마트 닉 RDMA 메모리 영역(341)의 구조와 함께 도 4에서 후술한다.
- [0057] 호스트 CPU(210)는 API들(320)을 이용하여 애플리케이션 RDMA 메모리 영역(311)에 저장된 애플리케이션 데이터를 외부 장치로 송신할 수 있도록 관리할 수 있다. API들(320)은 해시 테이블을 포함하는 호스트 플로우 테이블(321)을 포함하며, 호스트 CPU(210)는 애플리케이션 RDMA 메모리 영역(311)의 애플리케이션 데이터에 기초하여 호스트 플로우 테이블(321)을 생성할 수 있다. 이에 호스트 CPU(210)는 호스트 플로우 테이블(321)을 이용하여 스마트 닉(220)에 전달될 헤더를 생성할 수 있다. API들(320)은 호스트 메모리(230) 내에 별개로 존재하는 프리 리스트를 관리하고 API들(320)의 호스트 플로우 테이블(321)은 해시 테이블을 관리할 수 있다. 프리 리스트는 도 4에서 후술하고, 해시 테이블은 도 5에서 후술한다. 또한, 호스트 CPU(210)는 API들(320)을 이용하여 애플리케이션 RDMA 메모리 영역(311)에서의 데이터 쓰기(data write)와 데이터 읽기(data read)를 시간차 두고 수행할 수 있다. 예를 들어, 호스트 CPU(210)는 API들(320)을 이용하여 데이터 읽기 시 애플리케이션

RDMA 메모리 영역(311)에 블록(block)을 걸어 데이터 쓰기로 인한 데이터 변조 발생을 방지할 수 있다. 호스트 CPU(210)는 데이터 읽기가 종료된 후, API들(320)을 이용하여 RDMA 메모리 영역(311)의 블록(block)을 해제하여 데이터 쓰기를 수행할 수 있다.

- [0058] 호스트 CPU(210)는 API들(320)을 이용하여 호스트 플로우 테이블(321)에 기초한 헤더를 형성할 수 있다. 형성된 헤더는 실제 데이터를 포함하지 않은 실제 데이터 길이만을 포함할 수 있다. 이후, 스마트 닉(220)은 스마트 닉 플로우 테이블(343) 내 정보에 기초하여 실제 데이터를 스마트 닉 RDMA 메모리 영역(341)으로부터 가져와 포함시킬 수 있다.
- [0059] 호스트 CPU(210)는 네트워크 스택(330)을 통해 API들(320)로부터 수신된 헤더를 처리할 수 있다. 네트워크 스택(330)은 패킷 처리(Packet Processing) 구성(331)으로 TCP 레이어, IP 레이어 및 드라이버 레이어를 포함할 수 있다. 패킷 처리는 패킷이 외부 장치로 송신될 수 있도록 헤더에 대한 정보를 적절히 변형하는 것을 의미할 수 있다.
- [0060] 스마트 닉(220)은 호스트 CPU(210)와 유사하게 프리 리스트와 해시 테이블을 포함하는 스마트 닉 플로우 테이블(343)을 내부 메모리인 스마트 닉 메모리 내에 저장할 수 있다. 또한, 스마트 닉(220)은 네트워크 스택(330)에 기초한 방식과 RDMA를 이용한 방식에 기초하여 데이터를 획득할 수 있다. 스마트 닉 플로우 테이블(343)과 전술한 호스트 플로우 테이블(321)은 동일한 정보를 포함할 수 있지만 호스트 CPU(210)와 스마트 닉(220)의 동작에 의해 각기 다른 정보를 포함할 수 있다.
- [0061] 스마트 닉(220)에서의 네트워크 스택(330)에 기초한 방식은 전술한 바와 같이 호스트 CPU(210)의 제어에 의해 데이터가 커널 영역의 TCP 레이어, IP 레이어 및 드라이버 레이어를 통과함에 따라 송/수신할 수 있다.
- [0062] 스마트 닉(220)에서의 RDMA에 기초한 방식은 스마트 닉(220)이 호스트 CPU(210)로부터 권한을 받아 호스트 CPU(210)의 개입없이 호스트 메모리로부터 원격으로 직접 데이터를 읽거나(read) 쓸(write) 수 있다. RDMA에 기초한 방식은 RDMA를 수행하기 위하여 호스트 영역과 스마트 닉에 지정된 RDMA 메모리 영역을 필요로 할 수 있다. 스마트 닉(220)은 RDMA 수행 시 허용된 RDMA 메모리 영역 내에서만 애플리케이션 데이터를 읽거나 쓸 수 있다.
- [0063] 도 4는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역의 구조를 예시적으로 도시한다.
- [0064] 도 4(a)는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역에 블록이 할당된 경우를 예시적으로 도시하고, 도 4(b)는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역에 할당된 블록 중 하나의 블록이 할당 해제된 경우를 예시적으로 도시할 수 있다. 또한, 도 4(c)는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역에 블록이 할당된 경우에 프리 리스트의 구조를 예시적으로 도시하고, 도 4(d)는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 메모리 영역에 할당된 블록 중 하나의 블록이 할당 해제된 경우에 프리 리스트의 구조를 예시적으로 도시할 수 있다.
- [0065] 애플리케이션 RDMA 메모리 영역(311) 및 스마트 닉 RDMA 메모리 영역(341)(예: 메모리 영역(410))은 메모리 관리를 용이하게 하기 위하여 고정된 크기로 형성된(예를 들어, 나뉘어진) 복수의 메모리 블록(Memory Block)(411)들을 포함할 수 있다. 호스트 CPU(210)와 스마트 닉(220)은 각 플로우(flow) 별로 하나 이상의 블록을 할당하여 사용할 수 있으며, 하나의 플로우에 대한 둘 이상의 할당은 연속적인 블록에 할당될 수 있지만 연속적이지 않은 블록에도 할당될 수 있다. 이는 플로우의 상황(예: 네트워크 상황과 요청)에 따라 필요한 블록 개수가 변동될 수 있기 때문에 발생할 수 있다. 예를 들어, 호스트 CPU(210)는 제1 플로우(421)에 처음에 세개의 블록(도 4(a)에서 메모리 영역(410)의 좌측부터 1 블록, 1 블록, 2 블록)을 할당할 수 있다. 다만, 이후 제1 플로우(421)에서 한개의 블록이 필요하지 않아 호스트 CPU(210)는 한 개의 블록(예: 2 블록)을 반환(예: 프리(free))시키고, 해당 블록은 프리 리스트에 추가될 수 있다. 이후, 제2 플로우(422)에서 블록 할당을 요청하는 경우, 호스트 CPU(210)는 제1 플로우(421)에서 할당되었다 반환된 블록(예: 제2 블록)을 제2 플로우(422)로 할당할 수 있다. 이와 같은 방식을 이용하여 호스트 CPU(210)는 제1 플로우(421)에 세개의 블록을 할당하고, 제2 플로우(422)에 다섯개의 블록을 할당하며, 제3 플로우(423)에 두개의 블록을 할당할 수 있다. 또한, 호스트 CPU(210)와 스마트 닉(220)은 메모리 영역에 할당된 블록 중 하나의 블록이 할당 해제된 경우 빈 메모리 블록으로 형성될 수 있다.
- [0066] 호스트 CPU(210)와 스마트 닉(220)은 프리 리스트(440)를 이용하여 플로우가 할당되지 않은 블록(예: 빈 메모리

블록)들을 관리할 수 있다. 블록에 플로우가 할당되지 않은 것은 사용 중인 플로우가 없어 빈 블록 상태이거나 플로우가 블록 사용을 완료하여 해당 블록이 RDMA 메모리 영역으로 반환됨을 의미할 수 있다. 블록에 플로우가 할당되지 않은 경우, 호스트 CPU(210)와 스마트 닉(220)은 프리 리스트(440)에 해당 블록을 추가하여 다른 플로우에 의한 블록 할당을 요청할 때까지 대기시킬 수 있다. 또한, 호스트 CPU(210)와 스마트 닉(220)은 복수의 메모리 블록들 중 플로우에 할당되지 않은 블록들의 각각이 프리 리스트(440)에서 다른 메모리 블록을 포인팅(point)하도록 생성할 수 있다.

[0067] 예를 들어, 도 4(c)에서의 프리 리스트(440)는 머리(Head) 부분(441)을 시작으로 꼬리(Tail) 부분(442)을 끝으로 하여 각 메모리 블록이 다음 빈 메모리 블록을 포인팅하는 다섯 개의 블록(431)을 포함할 수 있다. 도 4(d)에서의 프리 리스트(440)는 하나의 빈 메모리 블록(432)이 추가되는 경우로 기존의 다섯 개의 블록(431)에서 꼬리(Tail) 부분(442) 쪽으로 하나의 메모리 블록(432)이 추가될 수 있다.

[0068] 또한, 호스트 CPU(210)와 스마트 닉(220)은 해시 테이블(Hash Table)을 이용하여 복수의 네트워크 플로우를 관리할 수 있다. 호스트 CPU(210)와 스마트 닉(220)은 각 플로우의 데이터를 키(key)로 하여 각 플로우에 접근하여 관리할 수 있다. 호스트 CPU(210)는 고정된 크기의 어레이(array)인 각 플로우에 대한 정보를 포함하는 플로우 리스트(Flow List)를 이용하여 다수의 플로우들에 대응하는 메모리 블록에 대해 접근하기 위한 정보를 해시 테이블에 기록할 수 있다.

[0069] 도 5는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 플로우 리스트를 포함하는 해시 테이블의 구조를 예시적으로 도시한다.

[0070] 호스트 CPU(210)와 스마트 닉(220)은 플로우 리스트를 이용하여 메모리 블록에 접근하여 관리할 수 있다. 해시 테이블(520)은 각 플로우 별 플로우 리스트(521, 522 및 523)를 가질 수 있다. 해시 테이블(520)은 플로우 자체가 직접적으로 저장된 것이 아닌 각 플로우의 데이터(예: src_ip, dst_ip, src_port 및 dst_port)에 기초하여 생성된 해시 평선(Hash Function)의 값이 저장될 수 있다. 호스트 CPU(210)와 스마트 닉(220)은 두 개의 해시 평선(Hash Function)을 이용하여 각 플로우에 접근할 수 있다. 또한, 호스트 CPU(210)와 스마트 닉(220)은 해시 충돌(Hash Collision)이 발생하는 경우, 제2 플로우의 플로우 리스트(522) 및 제3 플로우의 플로우 리스트(523)와 같이 뒤에 연결될 수 있다. 각 블록은 RDMA 메모리 영역(510)과 같이 인덱스(index) 번호가 매겨져 있어, 호스트 CPU(210)와 스마트 닉(220)은 블록의 인덱스 번호에 기초하여 접근할 수 있다. 예를 들어, 제1 플로우의 플로우 리스트(521)는 메모리 블록 인덱스 0,1,3을 포함하고, 제2 플로우의 플로우 리스트(522)는 메모리 블록 인덱스 2,4,5,6,7을 포함하며, 제3 플로우의 플로우 리스트(523)는 메모리 블록 인덱스 8,9를 포함할 수 있다.

[0071] 또한, 플로우 리스트(523)는 플로우 리스트 간의 구분을 위해 링크드 리스트(linked list) 형식으로 구성되어 충돌(Collision)을 방지하도록 사용되는 해시 평선(Hash Function) 중 두번째 해시 평선 정보, 블록 내에서 애플리케이션 데이터를 쓰기 시작하는 지점을 나타내는 스타트 포인트(Start Point) 정보, 플로우에 할당된(예: 사용 중인) 메모리 블록에 접근하기 위한 블록 인덱스 번호(Block Index) 정보, 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트(Start Byte)와 엔드 바이트(End Byte) 정보, 하나의 어레이의 블록 인덱스 번호 저장 영역의 용량이 초과되는 경우 새로 할당된 어레이인 다음 플로우 리스트를 포인팅하는 넥스트 포인터(Next Pointer) 정보, 전체 애플리케이션 데이터 크기인 토탈 애플리케이션 바이트(Total App Byte) 정보 및 RDMA로 복사된 전체 데이터 크기인 토탈 RDMA 바이트(Total RDMA Byte) 정보를 포함할 수 있다.

[0072] 예를 들어, 스마트 닉(220)이 RDMA를 수행하기 위하여 호스트 CPU(210)가 해시 테이블을 생성하는 경우, 호스트 CPU(210)는 충돌을 방지하도록 두개의 해시 평선을 사용하여 해시 테이블(520)을 링크드 리스트 형식으로 구성할 수 있다. 스마트 닉(220)은 블록 인덱스 번호 정보들을 이용하여 애플리케이션 RDMA 메모리 영역(311) 내 해당 메모리 블록들에 접근함에 따라 RDMA 읽기 동작을 수행할 수 있다. 스타트 바이트와 엔드 바이트 정보는 플로우에 인덱스 번호를 통해 할당된 메모리 블록에 접근 데이터를 쓰기 시작한 위치와 끝 위치를 나타낼 수 있다. 제1 플로우의 플로우 리스트(521)는 메모리 블록 인덱스 0에서 시작하고 3에서 끝나는 스타트 바이트와 엔드 바이트 정보를 포함하고, 제2 플로우의 플로우 리스트(522)는 메모리 블록 인덱스 2에서 시작하고 7에서 끝나는 스타트 바이트와 엔드 바이트 정보를 포함하며, 제3 플로우의 플로우 리스트(523)는 메모리 블록 인덱스 8에서 시작하고 9에서 끝나는 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다. 이에 호스트 CPU(210)와 스마트 닉(220)은 스타트 바이트와 엔드 바이트 정보를 확인하여 해당 플로우 리스트를 통해 어떠한 데이터가 전달되었는지를 확인하고 앞으로 전달되어야 할 데이터를 관리할 수 있다.

[0073] 또한, 호스트 CPU(210)와 스마트 닉(220)은 하나의 플로우 정보를 메모리 블록으로 저장함과 동시에 플로우 리

스트를 생성할 수 있다. 예를 들어, 호스트 CPU(210)와 스마트 닉(220)은 제3 플로우의 데이터 부분이 메모리 블록 8, 9를 구성하고 추가적인 메모리 블록이 필요하여 이를 추가하면 제3 플로우의 플로우 리스트(523)에 추가된 메모리 블록에 대한 정보를 이어서 저장할 수 있다(예: Get By Free List).

- [0074] 도 6은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 새로운 플로우 리스트가 할당되는 경우를 예시적으로 도시한다.
- [0075] 호스트 CPU(210)는 하나의 어레이의 블록 인덱스 번호 저장 영역의 용량이 초과되는 경우, 첫번째 플로우 리스트(611)에 이어 새로운(예: 두번째) 플로우 리스트(612)를 할당할 수 있다. 이에 스마트 닉(220)은 넥스트 포인터 정보를 이용하여 하나의 플로우(610)의 첫번째 플로우 리스트(611)에 이어진 두번째 플로우 리스트(612)를 확인할 수 있다. 두번째 플로우 리스트(612)는 첫번째 플로우 리스트(611)에 이어져 있기 때문에 두번째 해시 평션 정보 및 스타트 포인트 정보를 포함하지 않을 수 있다.
- [0076] 도 7은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 빈 플로우 리스트에 이어지는 블록 인덱스 번호를 복사하는 경우를 예시적으로 도시한다.
- [0077] 예를 들어, 하나의 플로우에 제1 플로우 리스트(710), 제2 플로우 리스트(720) 및 제3 플로우 리스트(730)가 있는 경우, 스마트 닉(220)은 RDMA 읽기를 수행함에 따라 제1 플로우 리스트(710)의 제1 블록 인덱스 번호(711)에 해당되는 데이터를 모두 읽을 수 있다. 이에 제1 플로우 리스트(710)에서의 제1 블록 인덱스 번호(711)는 삭제되어 해당 영역이 비어 있을 수 있다. 이러한 경우, 호스트 CPU(210)는 제1 블록 인덱스 번호(711)가 있었던 영역에 제2 플로우 리스트(720)의 제2 블록 인덱스 번호(721)를 복사(copy)할 수 있다. 또한, 호스트 CPU(210)는 넥스트 포인터 정보까지 복사할 수 있다. 이 후, 호스트 CPU(210)는 비어진 제2 플로우 리스트(720)를 제거함에 따라 불필요한 메모리 사용을 줄일 수 있다. 또한, 제2 블록 인덱스 번호(721) 복사 시 제1 플로우 리스트(710)의 스타트 포인트의 수정이 발생할 수 있다. 또한, 스타트 바이트와 엔드 바이트 및 토탈 RDMA 바이트는 읽기 시 수정될 수 있다.
- [0078] 도 8은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 스마트 닉의 읽기 동작을 위한 호스트 메모리와의 구조를 예시적으로 도시한다.
- [0079] 스마트 닉(220)은 미리 결정된 주기마다 호스트 메모리 내 플로우 정보(flow information, flow info)(810)에서 애플리케이션 RDMA 메모리 영역(311)으로부터 수신할 애플리케이션 데이터가 있는지 확인하고(F81) 호스트 메모리(800) 내 플로우 별 읽기 정보를 포함하는 호스트 RDMA 메모리 영역(820)에서 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어오며(F82), 읽어 온 읽기 정보에 기초하여 플로우 테이블(343)을 구성하며, 읽을 애플리케이션 데이터의 주소를 계산하여 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역(341)으로 저장할 수 있다.
- [0080] 호스트 영역의 메모리(800)는 플로우 정보(810) 및 호스트 RDMA 메모리 영역(820)을 포함할 수 있다. 호스트 RDMA 메모리 영역(820)은 애플리케이션의 RDMA 메모리 영역(311)과 별개의 RDMA 메모리 영역일 수 있다. 플로우 정보(810)는 수신할 데이터의 플로우 번호 정보를 포함할 수 있고, 예를 들어, 1, 3, 4의 플로우 번호가 기재될 수 있다. 호스트 RDMA 메모리 영역(820)은 모든 플로우의 데이터(예: 메타(meta) 정보)를 담고 있어, 스마트 닉(220)은 1, 3, 4의 플로우 번호에 대응되는 읽기 정보를 호스트 RDMA 메모리 영역(820)으로부터 읽어 올 수 있다. 메타 정보는 고정적인 공간에 저장될 수 있다. 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그(Flag) 정보, TCP 플로우 관리를 위한 ISN 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평션 정보 (Hash Func1, 2), 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보(예: 2 내지 7), 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0081] 도 9는 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 또 다른 네트워크 구조를 예시적으로 도시한다.
- [0082] 일 실시예에 따른 장치(200)에서의 또 다른 네트워크 구조(900)는 일 실시예에 따른 장치(200)에서의 네트워크 구조(300)의 구성과 동일한 호스트 CPU(210)가 구성하는 호스트 영역 내 애플리케이션(910), API들(920) 및 네트워크 스택(930)과 스마트 닉(220)이 구성하는 디바이스 영역을 포함할 수 있다. 다만, 애플리케이션(910)과 API들(920)의 구성에서 일 실시예에 따른 장치(200)에서의 네트워크 구조(300)와 차이가 있다.
- [0083] 호스트 CPU(210)는 애플리케이션 버퍼에 애플리케이션 데이터가 저장된 애플리케이션(910)을 실행하고, 애플리케이션 버퍼로부터 전달(F91)된 애플리케이션 데이터가 저장되는 API들 RDMA 메모리 영역(921) 및 API들 RDMA 메모리 영역(921)의 애플리케이션 데이터를 관리하기 위하여 프리 리스트를 관리하고 해시 테이블을 포함한 호

스트 플로우 테이블(922)을 포함하는 API들(920)을 실행하며, 네트워크 스택(930)을 이용하여 API들로부터 전송된 헤더를 처리(F92)할 수 있다.

- [0084] 스마트 닉(220)은 API들의 API들 RDMA 메모리 영역(921)에서 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역(941)에 저장(F91)하고 저장된 애플리케이션 데이터와 호스트 CPU(210)로부터 수신된 처리된 헤더에 기초하여 패킷을 형성할 수 있다(F94). 이 외의 패킷 컨트롤러(942) 및 TSO(944)는 전술한 일 실시예에 따른 장치(200)에서의 네트워크 구조(300)에 사용되는 것과 동일/유사하게 사용될 수 있다. 호스트 CPU(210)는 전술한 일 실시예에 따른 장치(200)에서와 동일하게 패킷의 헤더와 데이터를 완전히 분리하여 전송할 수 있다. 이를 통해 Cloud환경에서는 TCP 처리에 대한 부담을 줄이고 이를 테넌트 증가를 위한 자원으로 사용할 수 있다. 또한, 비디오 스트리밍 시장은 지원하고 요구되는 비디오 품질(예: 480p, 720p, 등)이 높아짐에 따라 데이터 사이즈 또한 증가하여(예: 4K에서는 시간당 11.25GB의 데이터가 사용되어), 일 실시예에 따른 장치(200)가 더욱 비디오 스트리밍 분야의 성능을 향상시킬 수 있다.
- [0085] 도 10은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치의 또 다른 네트워크 구조에 기초한 호스트 CPU가 스마트 닉으로 보내는 통지 메시지 및 읽기 메시지를 예시적으로 도시한다.
- [0086] 호스트 CPU(210)는 스마트 닉(220)이 데이터를 읽도록(예: 읽기 동작을 수행하도록) 스마트 닉(220)으로 통지 메시지(notification message)(1010)(예: Type: 0) 및 읽기 메시지(Read message)(1020)(예: Type: 1)를 전송할 수 있다. 통지 메시지는 처음 구성할 때의 ISN 및 해시 평선 정보(Hash Func1, 2)(예: 해시 테이블 정보)(1011)를 포함할 수 있고, 스마트 닉(220)은 ISN 및 해시 평선 정보(1011)에 기초하여 스마트 닉 플로우 테이블(943)을 생성할 수 있다. 호스트 CPU(210)는 플로우 생성 시 통지 메시지(1010)를 전송하고, 스마트 닉(220)은 스마트 닉 플로우 테이블(943)을 이용하여 읽기 메시지(1020)에 기초하여 읽어 온 애플리케이션 데이터를 관리할 수 있다.
- [0087] 스마트 닉(220)은 읽기 메시지(1020)에 기초하여 API들 RDMA 메모리 영역(921)의 애플리케이션 데이터를 읽어 올 수 있다. 읽기 메시지(1020)는 전송할 데이터에 대한 주소인 토탈 플로우 넘버(Total Flow Number), 데이터 크기인 렘스(Length), 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보(1021)를 포함할 수 있다. 토탈 플로우 넘버는 총 보내지는 플로우 개수를 의미하고, 렘스는 메시지 길이를 의미할 수 있다.
- [0088] 스마트 닉(220)은 토탈 플로우 넘버, 렘스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보(1021)에 기초하여 요청된 모든 애플리케이션 데이터를 API들 RDMA 메모리 영역(921)으로부터 (예를 들어 scatter-gathering 기술을 이용하여) 한 번에 읽어서 스마트 닉 RDMA 메모리 영역(941)에 저장할 수 있다. 이는 스마트 닉(220)이 복수의 메모리 블록 내 애플리케이션 데이터를 한 번에 읽어 온 후 저장하는 것을 의미할 수 있다. 또한, 한 번에 읽어 오는 것은 읽기 메시지(1020)에 읽어 올 모든 블록에 대한 정보를 포함하기 때문에 가능할 수 있다. 스마트 닉(220)은 블록 인덱스 번호로부터 API들 RDMA 메모리 영역(921)의 주소를 계산할 수 있다. 스마트 닉(220)은 스마트 닉 RDMA 메모리 영역(예: 해시 테이블 내)에 애플리케이션 데이터(예: 읽어 온 데이터들)가 저장된 후, 호스트 CPU(210)에 읽기 완료 신호(예: 플래그 메시지(Flag message))를 송신할 수 있다. 호스트 CPU(210)는 읽기 완료 신호를 수신함에 따라 읽기 작업이 완료됨을 알 수 있다. 또 다른 예로, 스마트 닉(220)은 직접 API들 RDMA 메모리 영역(921) 내 플래그 공간에 읽기 완료를 쓸 수 있다. 스마트 닉(220)의 해시 테이블(예: 플로우 리스트)은 호스트 CPU(210)의 해시 테이블(520) 구성에서 Wait ACK 정보 영역이 추가될 수 있다. Wait ACK 정보는 ISN과 ack 메시지를 어디까지 받았는지를 확인할 수 있는 정보를 의미할 수 있다. ack 메시지는 메시지 성공여부를 확인하는 메시지일 수 있다.
- [0089] 도 11은 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 방법의 흐름도를 예시적으로 도시한다.
- [0090] 단계(1110)에서, 스마트 닉은 호스트 CPU로부터 애플리케이션 데이터를 읽어 올 수 있다. 스마트 닉이 호스트 CPU로부터 애플리케이션 데이터를 읽어오는 방식은 두 가지로 수행될 수 있다.
- [0091] 단계(1120)에서, 스마트 닉은 읽어 온 애플리케이션 데이터를 스마트 닉 RDMA 메모리 영역으로 저장할 수 있다.
- [0092] 단계(1130)에서, 스마트 닉은 호스트 CPU로부터 처리된 헤더를 수신할 수 있다.
- [0093] 단계(1140)에서, 스마트 닉은 스마트 닉 RDMA 메모리 영역에 저장된 애플리케이션 데이터와 처리된 헤더에 기초하여 패킷을 형성할 수 있다.
- [0094] 스마트 닉이 호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 첫번째 방식은 우선, 스마트 닉이 미리 결정된 주기마다 호스트 메모리 내 플로우 정보에 기초하여 애플리케이션 RDMA 메모리 영역으로부터 읽을 애플리케이션

데이터가 있는지 확인할 수 있다. 플로우 정보는 읽을 데이터의 블록 인덱스 번호 정보를 포함할 수 있다.

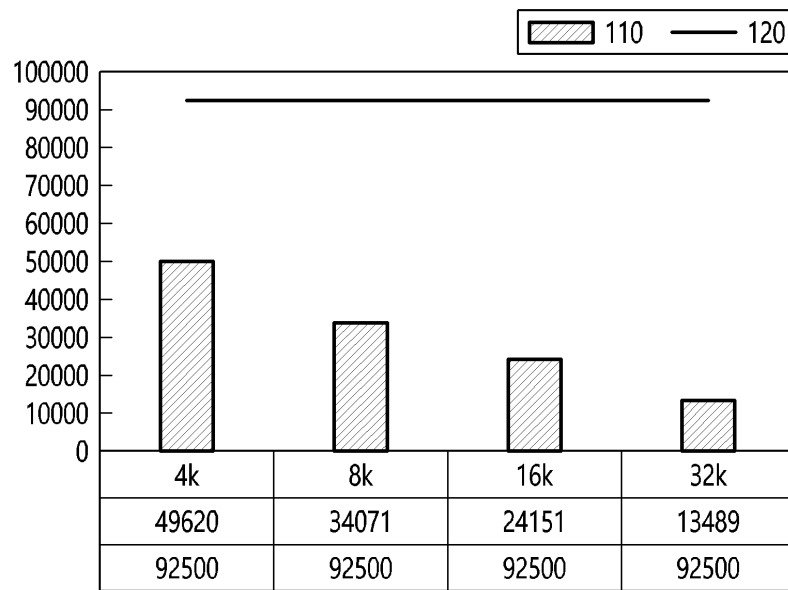
- [0095] 그런 후, 스마트 닉은 호스트 메모리 내 플로우 별 읽기 정보를 포함하는 호스트 RDMA 메모리 영역에서 읽을 애플리케이션 데이터에 기초한 읽기 정보를 읽어 올 수 있다. 스마트 닉은 읽어 온 상기 읽기 정보에 기초하여 플로우 테이블을 구성하고, 읽을 애플리케이션 데이터의 주소를 계산하여 애플리케이션 데이터를 읽어 올 수 있다. 플로우 별 읽기 정보는 각 플로우 별로 각 플로우의 읽기가 끝났는지 확인을 위한 플래그 정보, TCP 플로우 관리를 위한 ISN 정보, 해시 테이블 관리를 위한 제1 및 제2 해시 평선 정보, 상기 읽을 애플리케이션 데이터의 블록 인덱스 번호 정보, 메모리 블록 내 데이터의 시작점과 끝점을 가리키는 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0096] 스마트 닉이 호스트 CPU로부터 애플리케이션 데이터를 읽어 오는 두번째 방식은 스마트 닉이 호스트 CPU로부터 통지 메시지와 읽기 메시지를 수신함에 따라 애플리케이션 데이터 읽기가 수행될 수 있다.
- [0097] 먼저, 스마트 닉은 호스트 CPU로부터 수신된 통지 메시지에 기초하여 스마트 닉 플로우 테이블을 생성할 수 있다. 통지 메시지는 ISN 및 해시 평선 정보를 포함할 수 있다.
- [0098] 그런 후, 스마트 닉은 호스트 CPU로부터 수신된 읽기 메시지에 기초하여 API들 RDMA 메모리 영역에서의 애플리케이션 데이터를 읽어 올 수 있다. 읽기 메시지는 전송할 데이터에 대한 주소인 토탈 플로우 넘버, 데이터 크기인 렱스, 해시 평선, 블록 인덱스 번호 및 스타트 바이트와 엔드 바이트 정보를 포함할 수 있다.
- [0099] 또한, 스마트 닉은 읽기 메시지에 기초하여 요청된 모든 애플리케이션 데이터를 호스트 CPU의 API들 RDMA 메모리 영역에서 한 번에 읽어서 스마트 닉 RDMA 메모리 영역에 저장할 수 있다. 그런 후, 스마트 닉은 애플리케이션 데이터를 통지 메시지 및 읽기 메시지에 기초하여 수신하는 경우, 호스트 CPU가 API들 RDMA 메모리 영역 내 플래그 공간에 읽기 완료에 대한 쓰기를 수행하도록 스마트 닉 RDMA 메모리 영역에 상기 애플리케이션 데이터를 저장한 후 호스트 CPU에 읽기 완료 신호를 송신할 수 있다. 이러한 RDMA 기술을 응용한 방식은 커널 스택뿐만 아니라 QUIC 및 user level TCP 등 다양한 플랫폼(platform)에 적용 가능할 수 있다.
- [0100] 스마트 닉을 이용한 CPU 향상 결과는 NSDI' 14에 발표된 mTCP, NSDI' 20에 발표된 AccelTCP, NSDI ' 22에 발표된 FlexTOE 등에 존재하지만, 전체 적으로 데이터 크기가 작은 경우에만 좋은 성능을 보여주고 있다. 예를 들어, AccelTCP를 이용한 결과는 2048B와 4096B로 증가할 때 거의 성능에 변화가 없었으며, FlexTOE를 이용한 결과 또한 2M에서 32M로 크기가 증가할 때 성능의 변화가 거의 없는 것을 보여주었다. 반면에, 일 실시예에 따른 스마트 닉을 통해 TCP 프로토콜을 가속화하는 장치 및 방법은 데이터가 커질수록 호스트 CPU가 처리할 연산이 줄어들기 때문에 데이터 크기가 큰 상황에서도 유리한 이점을 가지고 있다. 이에 실기간으로 대량의 데이터를 주고받으며 빠르게 처리해주어야 하는 스마트 시티(Smart City), 각 유저별로 다양하고 많은 데이터를 처리하는 오픈 월드(open World), Cloud storage (drop box, google drive, etc) 및 VR 등과 같은 거대하고 많은 데이터를 처리해야 하는 대부분의 산업에서 적용할 수 있으며 성능적 향상 또한 기대할 수 있다.
- [0101] 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 상기 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.
- [0102] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 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 장치에 저장될 수 있다. 소프트웨어는 네트워크로 연결된 컴

퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

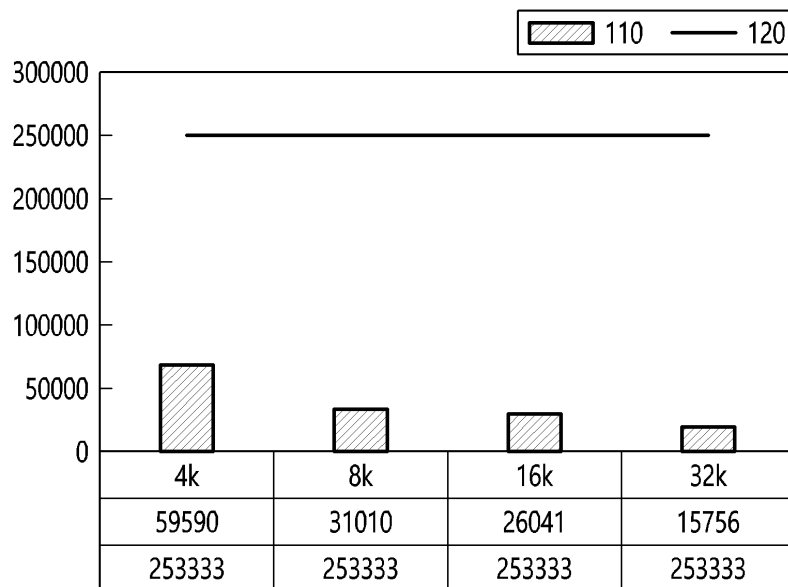
- [0103] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있으며 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), 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.
- [0104] 위에서 설명한 하드웨어 장치는 실시예의 동작을 수행하기 위해 하나 또는 복수의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.
- [0105] 이상과 같이 실시예들이 비록 한정된 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 이를 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.
- [0106] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

도면1

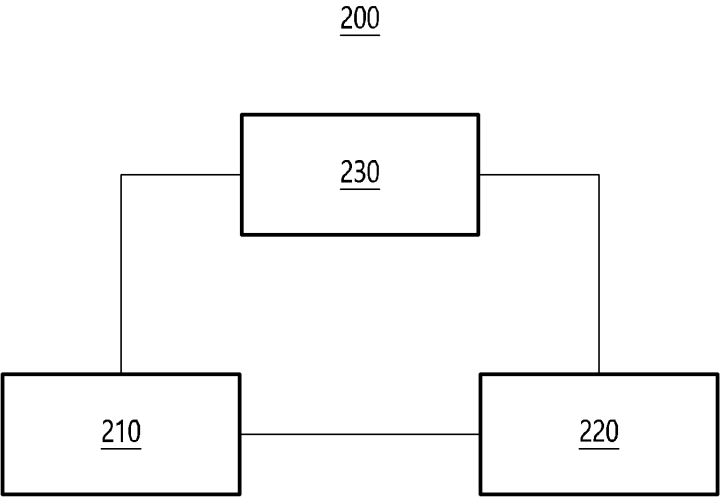


(a)

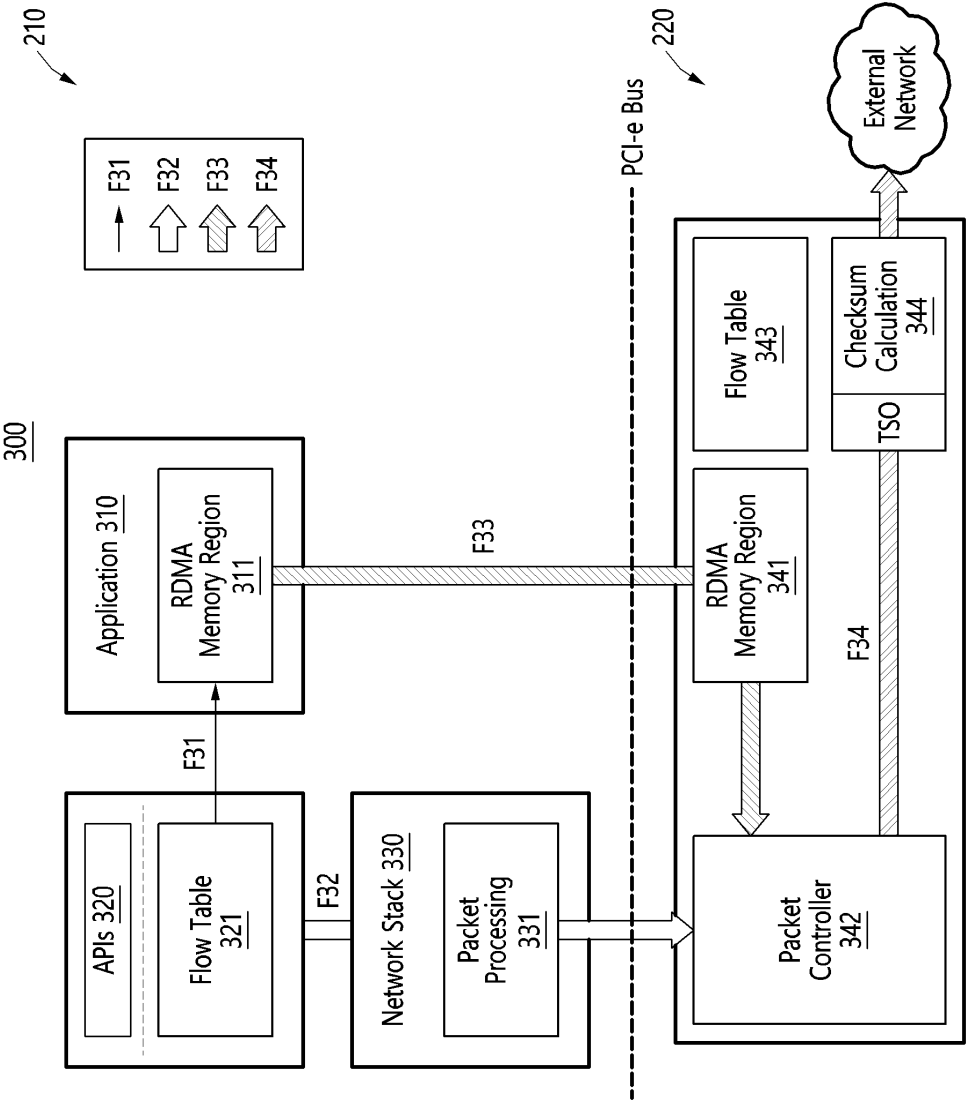


(b)

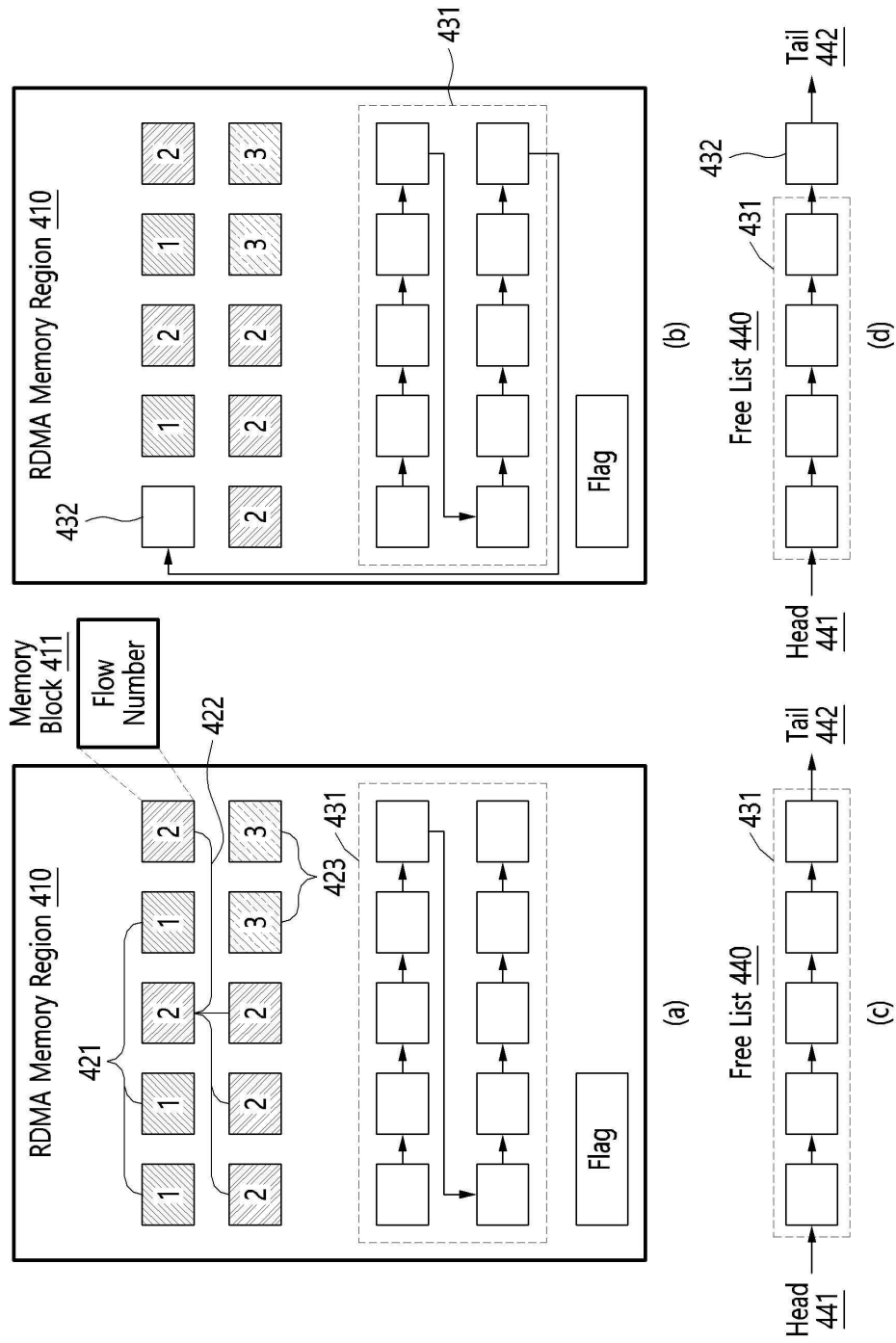
도면2



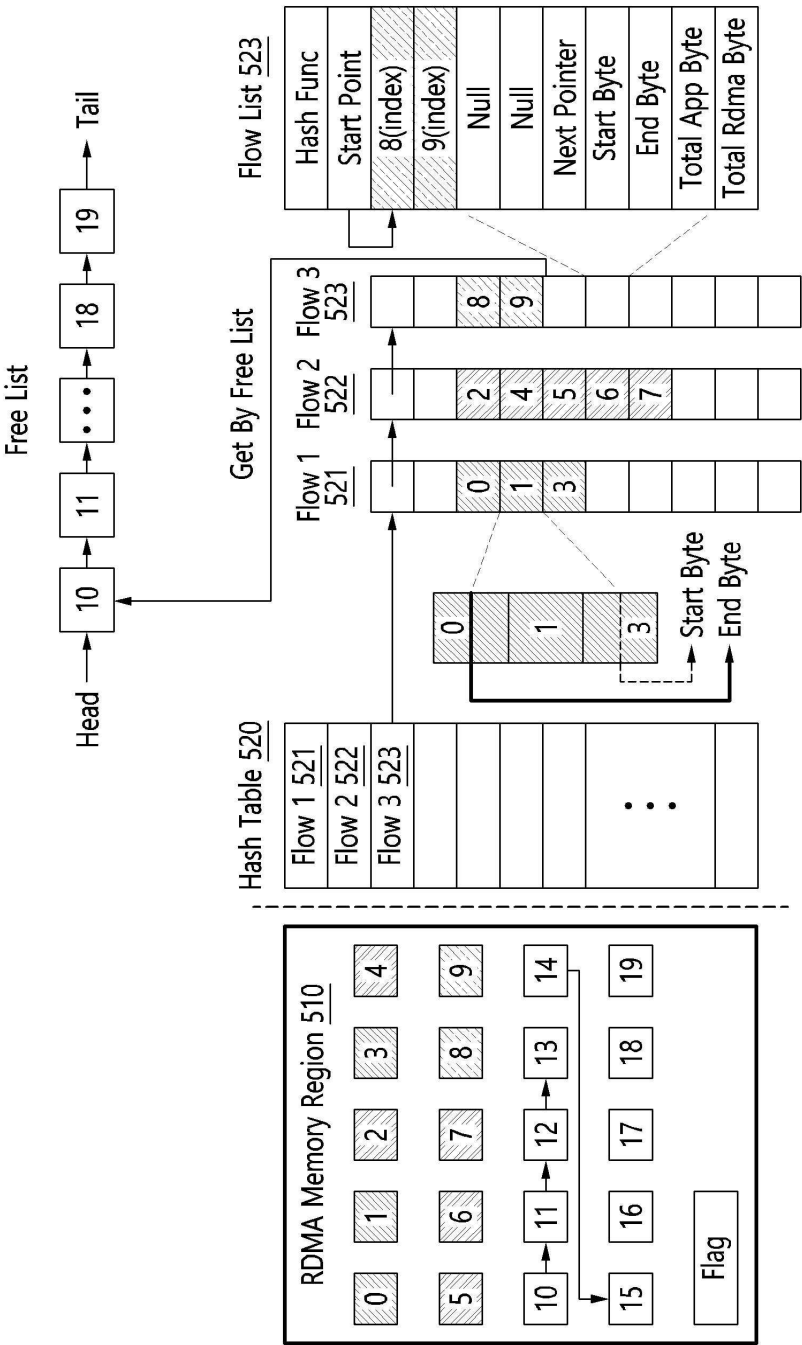
도면3



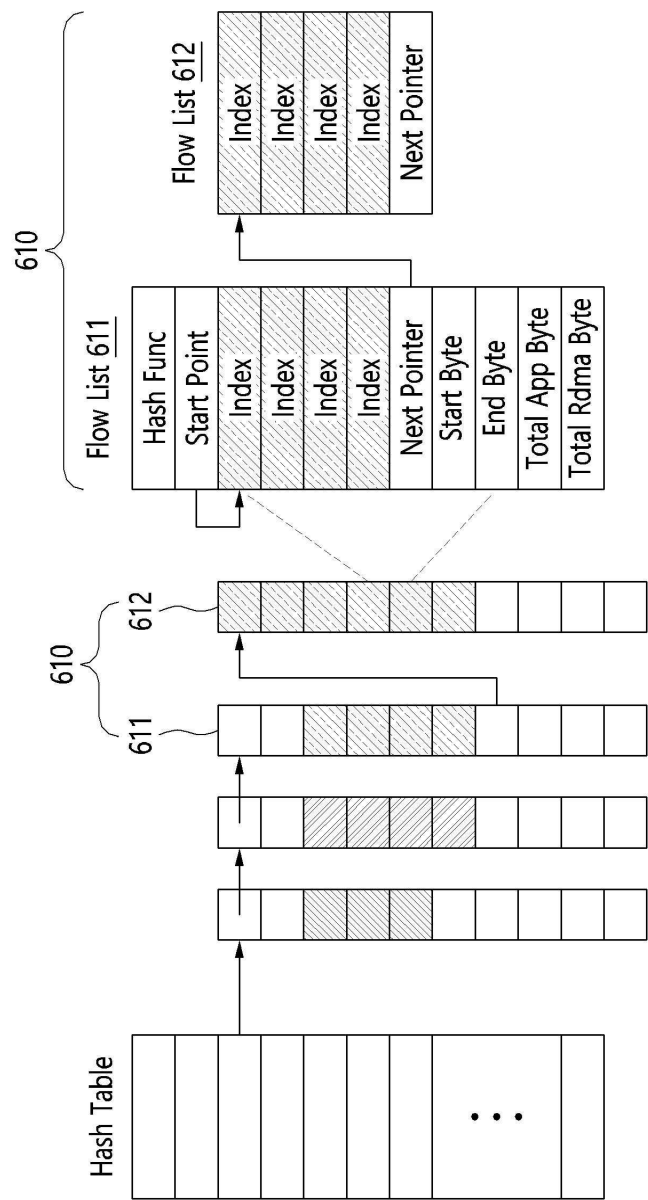
도면4



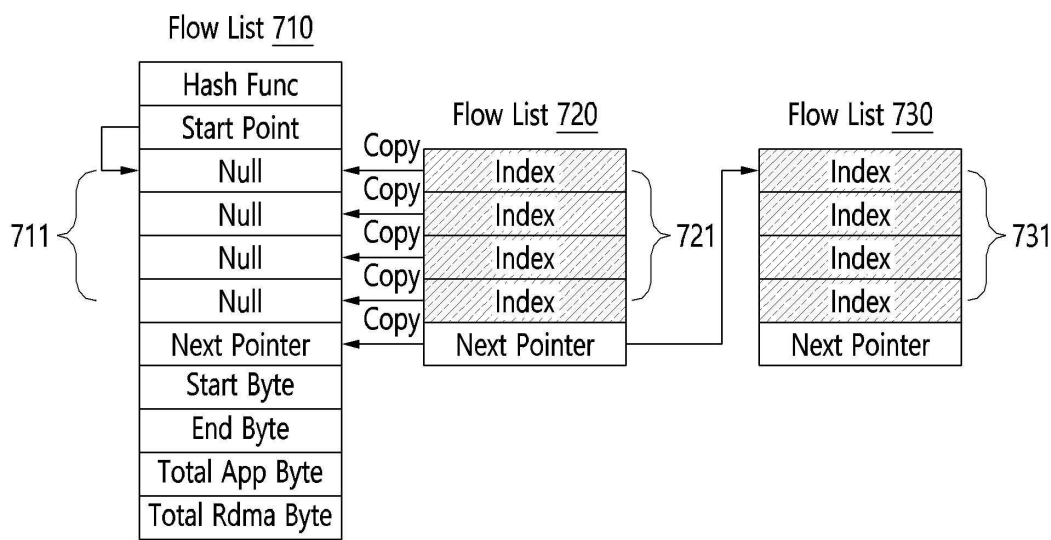
도면5



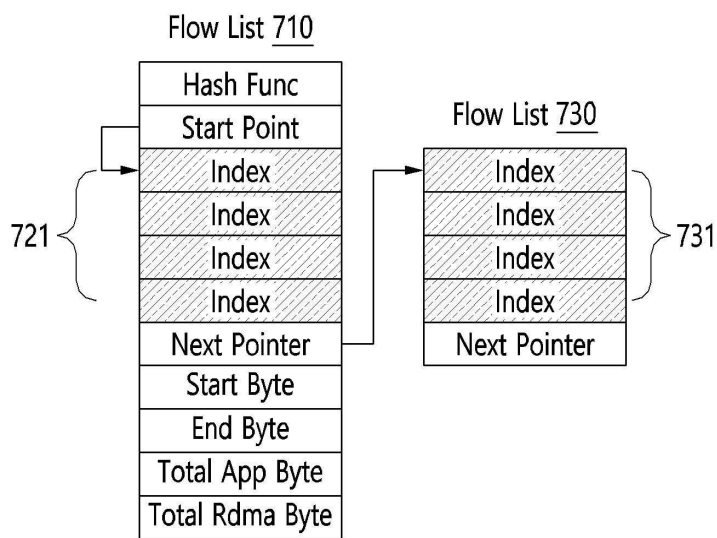
도면6



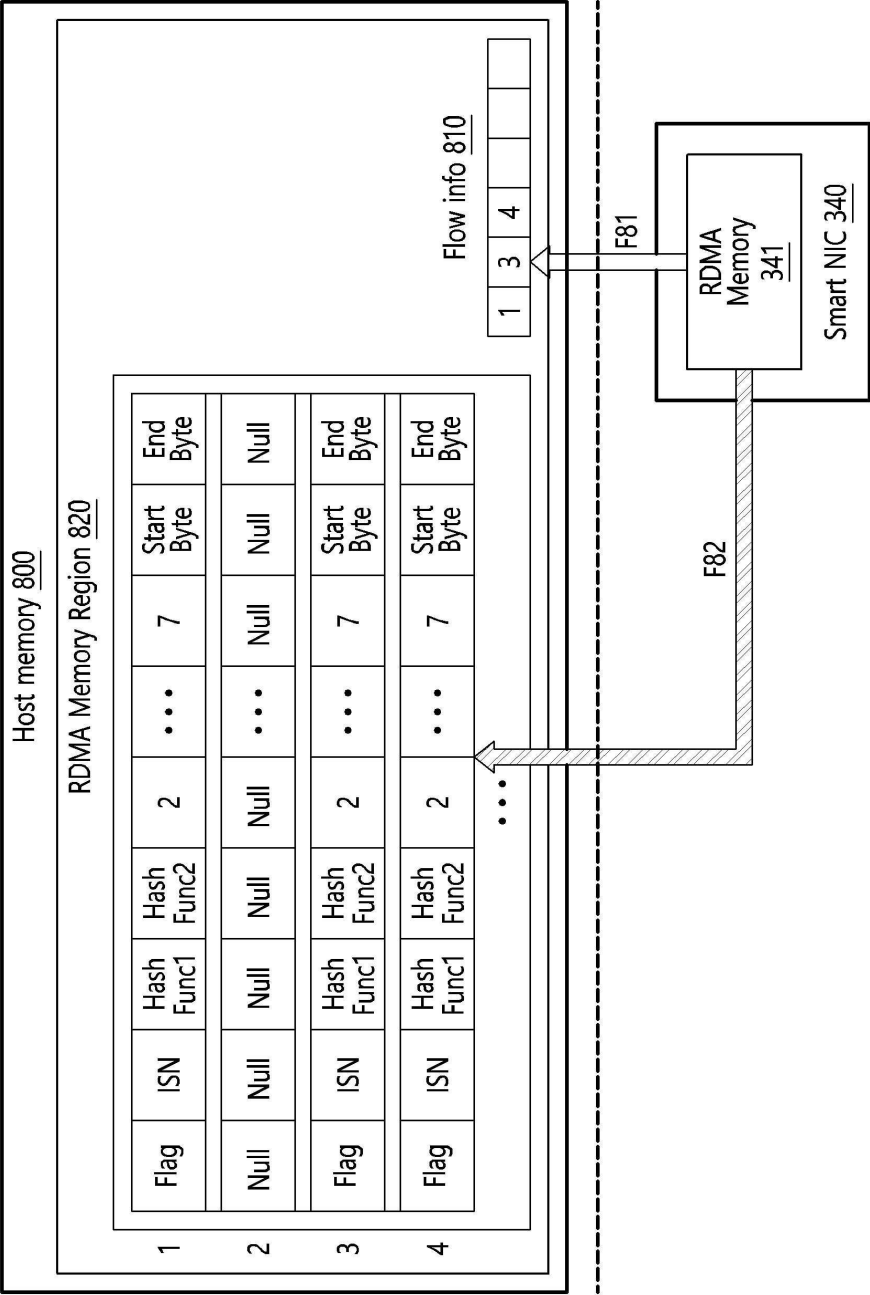
도면7



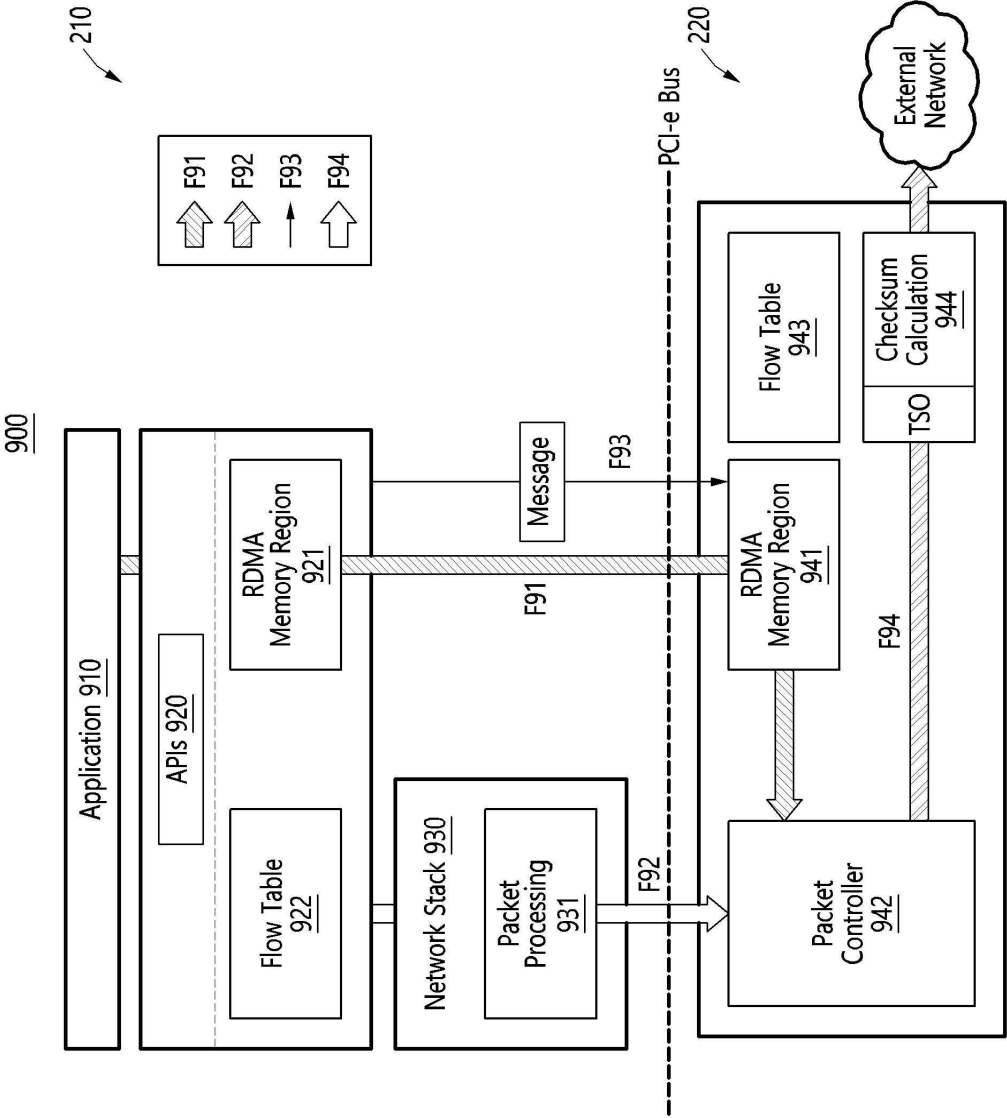
After copy



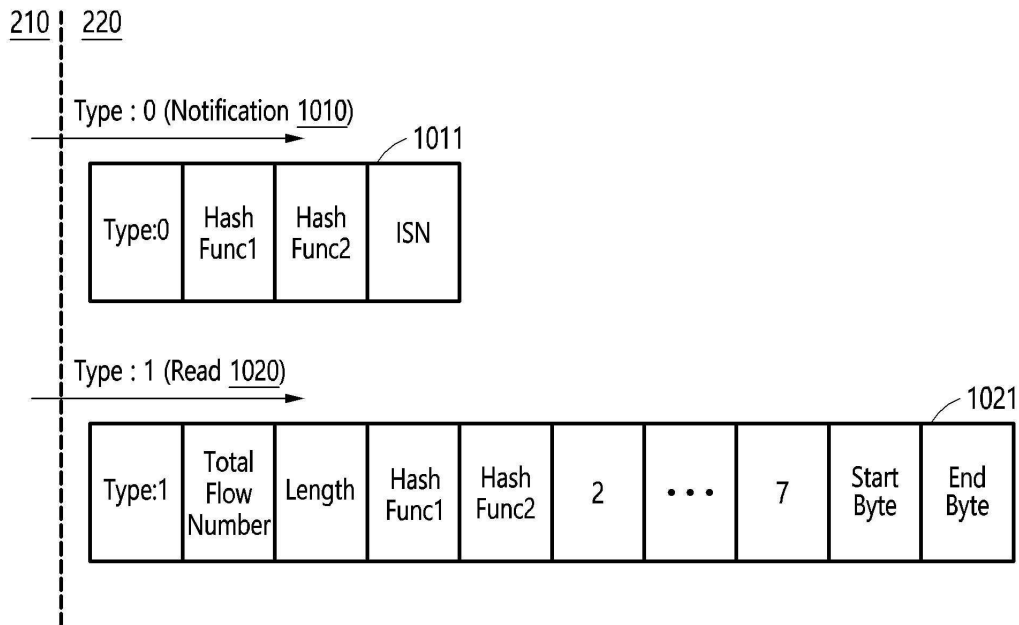
도면8



도면9



도면10



도면11

