(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년03월16일
(11) 등록번호 10-2374572
(24) 등록일자 2022년03월10일

(51) 국제특허분류(Int. Cl.)
H04L 41/00 (2022.01) *H04L 45/122* (2022.01)
H04L 49/00 (2022.01)
(52) CPC특허분류
H04L 41/145 (2013.01)
H04L 41/0806 (2013.01)
(21) 출원번호 10-2016-7034175
(22) 출원일자(국제) 2015년02월12일
심사청구일자 2020년02월12일
(85) 번역문제출일자 2016년12월06일
(65) 공개번호 10-2017-0015320
(43) 공개일자 2017년02월08일
(86) 국제출원번호 PCT/US2015/015604
(87) 국제공개번호 WO 2015/187209
국제공개일자 2015년12월10일
(30) 우선권주장
14/298,717 2014년06월06일 미국(US)
(56) 선행기술조사문헌
US20090313592 A1*
US20110191774 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
넷스피드 시스템즈
미국, 캘리포니아 95134, 산 호세, 빌딩 11, 실리 애비뉴 2670
(72) 발명자
쿠마, 사일레쉬
미국, 캘리포니아 95134, 산 호세, 빌딩 11, 실리 애비뉴 2670
노리쥐, 에릭
미국, 캘리포니아 95134, 산 호세, 빌딩 11, 실리 애비뉴 2670
가포니, 피에르지오르지오
미국, 캘리포니아 95134, 산 호세, 빌딩 11, 실리 애비뉴 2670
(74) 대리인
양영준, 김연송, 백만기

전체 청구항 수 : 총 17 항 심사관 : 윤태섭

(54) 발명의 명칭 **네트워크 온 칩 설계를 위한 트랜잭션 트래픽 스펙**

(57) 요 약

여기서 설명되는 구현 예는 NoC를 생성하고 최적화하기 위한 정보로 통합된 스펙에 따른다. 또한, 통합된 스펙은 트래픽 추적 파일의 생성을 가능하게 할 수 있다. 추적 파일에 기초하여, NoC에서 패킷을 주입하는 성능 시뮬레이션을 수행할 수 있다. 통합된 스펙은 원하는 구현에 따른 대역폭, 트래픽, 지터, 의존 정보 및 속성 정보에 대한 파라미터를 포함할 수 있다.

대 표 도

(52) CPC특허분류
H04L 45/122 (2019.05)
H04L 49/109 (2013.01)

명 세 서

청구범위

청구항 1

프로세스를 실행하기 위한 명령(instruction)을 저장하는 비일시적 컴퓨터 판독 가능 매체에 있어서, 상기 명령은,

복수의 파라미터를 포함하는 네트워크 온 칩(NoC, Network on Chip)의 스펙을 설계하는 단계 - 상기 복수의 파라미터는 NoC 기반 시스템의 설계 및 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내고, 상기 복수의 파라미터는 상기 NoC 기반 시스템의 복수의 라우터 및 상호 연결성 정보를 포함하고, 상기 NoC 기반 시스템의 메시지 의존성과 연관된 의존 정보, 대역폭 정보, 및 속성 정보를 포함함 -;

상기 NoC 기반 시스템의 복수의 라우터 및 상호 연결성이 상기 스펙의 요구를 충족시키도록 구성되게끔 상기 복수의 라우터 및 상호 연결성을 포함하는 상기 NoC 기반 시스템을 생성하는 단계; 및

상기 NoC 기반 시스템을 생성하는데 이용되는 상기 스펙의 복수의 파라미터를 이용하여 상기 NoC 기반 시스템의 성능 시뮬레이션을 수행하는 단계

를 포함하는, 비일시적 컴퓨터 판독 가능 매체.

청구항 2

삭제

청구항 3

제1항에 있어서,

상기 대역폭 정보는,

기설정된 시간 동안 등시성(isochronous) 트래픽에 대한 대역폭을 보장하기 위한 적어도 하나의 파라미터를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 4

제1항에 있어서,

상기 대역폭 정보는,

트래픽 지터(jitter)를 나타내는 적어도 하나의 파라미터를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 5

제1항에 있어서,

상기 의존 정보는,

상기 NoC 기반 시스템의 복수의 요소 간의 의존성 또는 상기 NoC 기반 시스템에서 복수 유형의 메시지 간의 의존성을 나타내는 비일시적 컴퓨터 판독 가능 매체.

청구항 6

제1항에 있어서,

상기 의존 정보는,

상기 NoC 기반 시스템의 복수의 요소 간의 비의존성 또는 상기 NoC 기반 시스템에서 복수 유형의 메시지 간의 비의존성을 나타내는 비일시적 컴퓨터 판독 가능 매체.

청구항 7

제1항에 있어서,

상기 대역폭 정보는,

적어도 하나의 트랜잭션(transaction)의 메시지 대역폭 요구, 상기 적어도 하나의 트랜잭션의 메시지 서비스 품질 요구, 상기 적어도 하나의 트랜잭션의 메시지 대기 시간 요구 및 상기 적어도 하나의 트랜잭션의 대기 시간 요구 중 적어도 하나를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 8

제1항에 있어서,

상기 상호 연결성 정보는,

상기 NoC 기반 시스템의 트래픽 시뮬레이션을 나타내는 상기 NoC 기반 시스템의 적어도 하나의 메시지에 대한 트랜잭션 정보, 상기 NoC 기반 시스템의 메시지 전송을 위한 프로토콜 정보 및 상기 NoC 기반 시스템의 메시지에 대한 시퀀스를 나타내는 주문 정보 중 적어도 하나를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 9

제8항에 있어서,

상기 트랜잭션 정보는, 적어도 하나의 트랜잭션을 포함하고,

상기 적어도 하나의 트랜잭션 각각은 홉(hop)의 시퀀스를 나타내고, 상기 홉의 시퀀스에서 각 홉은 소스 에이전트로부터 목적지 에이전트로의 상기 적어도 하나의 메시지를 나타내며,

상기 NoC 기반 시스템을 생성하는 단계는,

상기 적어도 하나의 트랜잭션 각각에 대한 홉의 시퀀스에 따라, 상기 소스 에이전트로부터 상기 목적지 에이전트로 각각의 홉에 대한 상기 적어도 하나의 메시지를 전송하기 위해, 상기 트랜잭션 정보에 기초하여 상기 NoC 기반 시스템을 구성하는 단계를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 10

제9항에 있어서,

상기 트랜잭션 정보는, 상기 적어도 하나의 트랜잭션의 적어도 하나의 서브 셋에 대한 시스템 레벨 시뮬레이션이 가능하도록 구성되는 비일시적 컴퓨터 판독 가능 매체.

청구항 11

제1항에 있어서,

상기 속성 정보는,

적어도 하나의 트랜잭션의 메시지 크기, 상기 적어도 하나의 트랜잭션의 메시지 우선순위, 상기 적어도 하나의 트랜잭션의 메시지 서비스 품질(QoS, Quality of Service) 및 상기 적어도 하나의 트랜잭션의 메시지 주문 요구 중 적어도 하나를 나타내는 메시지 속성 정보를 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 12

삭제

청구항 13

제1항에 있어서,

상기 명령은, 상기 시뮬레이션을 수행하기 위한 상기 스펙의 복수의 파라미터로부터 트래픽 추적 데이터베이스를 생성하는 단계를 더 포함하고,

상기 트래픽 추적 데이터베이스는, 적어도 하나의 트래픽 추적 파일을 포함하고,

상기 적어도 하나의 트래픽 추적 파일 각각은, 트랜잭션을 개시하기 위한 정보를 나타내는 비일시적 컴퓨터 판

독 가능 매체.

청구항 14

제13항에 있어서,

상기 트랜잭션을 개시하기 위한 정보는,

상기 스펙의 복수의 파라미터 중 어느 하나에 기초하여 결정되는 비일시적 컴퓨터 판독 가능 매체.

청구항 15

제14항에 있어서,

상기 명령은, 상기 복수의 파라미터 및 통계 모델에 기초하여 복수의 트랜잭션에서 상기 트랜잭션을 선택하는 단계를 더 포함하는 비일시적 컴퓨터 판독 가능 매체.

청구항 16

제1항에 있어서,

상기 대역폭 정보는,

트랜잭션 레이트, 상기 트랜잭션의 메시지 레이트 및 상기 NoC 기반 시스템의 적어도 하나의 호스트의 레이트 중 적어도 하나를 나타내는 비일시적 컴퓨터 판독 가능 매체.

청구항 17

네트워크 온 칩(NoC) 기반 시스템으로서,

미리 설계된 스펙으로부터 구성되는, 복수의 라우터 및 상호 연결을 포함하고,

상기 스펙은 복수의 파라미터를 포함하고, 상기 복수의 파라미터는 상기 NoC 기반 시스템의 설계 및 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내고, 상기 복수의 파라미터는 상기 NoC 기반 시스템의 복수의 라우터 및 상호 연결성 정보를 포함하고, 상기 NoC 기반 시스템의 메시지 의존성과 연관된 의존 정보, 대역폭 정보, 및 속성 정보를 포함하고,

상기 NoC 기반 시스템, 상기 복수의 라우터 및 상호 연결성은 상기 스펙의 요구를 충족시키도록 생성되고,

상기 NoC 기반 시스템을 생성하는데 이용되는 상기 스펙의 복수의 파라미터는 상기 NoC 기반 시스템 상에서 수행되는 성능 시뮬레이션에 이용되도록 구성되는, NoC 기반 시스템.

청구항 18

삭제

청구항 19

제17항에 있어서,

상기 대역폭 정보는,

기설정된 시간 동안 등시성(isochronous) 트래픽에 대한 대역폭을 보장하기 위한 적어도 하나의 파라미터를 포함하는 NoC 기반 시스템.

청구항 20

제17항에 있어서,

상기 대역폭 정보는,

트래픽 지터를 나타내는 적어도 하나의 파라미터를 포함하는 NoC 기반 시스템.

발명의 설명

기 술 분 야

[0001] 본 명세서에 기재된 방법 및 실시 예들은 인터커넥트(interconnect) 아키텍처에 관한 것으로, 보다 상세하게는, NoC 설계 및 시뮬레이션에 이용되는 트랜잭션 트래픽 스퀵에 관한 것이다.

배 경 기 술

[0002] 칩의 구성요소의 수는 집적화 레벨, 시스템 복잡도의 증가 및 트랜지스터 구조의 축소로 인하여 급속히 증가하고 있다. 복잡한 시스템 온 칩(SoC, System on Chip)은 다양한 구성요소, 예를 들어, 프로세서 코어, DSP, 하드웨어 가속기, 메모리 및 I/O를 포함할 수 있는 반면, 칩 다중 프로세서(CMP, Chip Multi-Processor)는 다수의 동종의 프로세서 코어, 메모리 및 I/O 서브시스템을 포함할 수 있다. SoC 및 CMP 시스템 모두에서, 온 칩 인터커넥트(on-chip interconnect)는 다양한 구성요소 간의 고성능 통신을 제공하는 역할을 한다. 종래의 버스 및 크로스바 기반 인터커넥트의 확장성 한계로 인하여, NoC는 칩 상의 많은 구성요소를 상호 연결하는 패러다임으로 등장하였다. NoC는 점대점(point-to-point) 물리적 링크를 사용하여 상호 연결된 여러 라우팅 노드로 구성된 글로벌 공용의 통신 인프라이다.

[0003] 메시지는 소스에 의해 주입되고 여러 중간 노드와 물리적 링크를 통해 소스 노드에서 목적지로 라우팅 된다. 이후, 목적지 노드는 메시지를 추출하고 메시지를 목적지로 제공한다. 본 명세서의 나머지 부분에서, '구성요소', '블럭', '호스트', 또는 '코어'라는 용어는 NoC를 이용하여 상호연결되는 다양한 시스템 구성요소를 나타내도록 호환성 있게 사용될 것이다. 또한, '라우터' 및 '노드'라는 용어도 호환성 있게 사용될 것이다. 일반성을 잃지 않고, 다수의 상호연결된 구성요소들을 포함하는 시스템은 '멀티-코어 시스템'으로 지칭될 것이다.

[0004] 시스템 네트워크를 만들기 위해 라우터들이 서로 연결될 수 있는 여러 토폴로지(topoligies)가 있다. 양방향 고리(도 1a 참조), 2-D(이차원) 메쉬(도 1b 참조), 및 2-D 토러스(Torus)(도 1c 참조)는 관련 기술의 토폴로지에 대한 예시들이다. 메쉬 및 토러스는 2.5-D(2.5 차원) 또는 3-D(3차원) 조직으로 또한 확장될 수 있다. 도 1d는 3D 메쉬 NoC을 도시하며, 서로 적층된 3층의 3x3 2D 메쉬를 도시한다. NoC 라우터는 포트를 두 개까지 가질 수 있으며, 하나는 위쪽 층의 라우터에 연결되고, 다른 하나는 아래쪽 층의 라우터에 연결된다. 일 예로, 중간층의 라우터(111)는 두 포트를 모두 가지며, 하나는 상부 층의 라우터에 연결되고, 다른 하나는 하부 층의 라우터에 연결된다. 라우터들(110,112)은 하부 메쉬 층 및 상부 메쉬 층에 각각 있어서, 오직 상향 포트(113)와 하향 포트(113)만을 갖는다.

[0005] 패킷들은 다양한 구성들 사이의 상호 통신을 위한 메시지 전달 유닛이다. 라우팅(routing)은 라우터 및 물리적 링크로 구성된 네트워크의 경로를 식별하는 과정을 포함하며 네트워크상에서 패킷들이 소스에서부터 목적지까지 전달된다. 구성요소들은 하나 또는 다수의 라우터의 하나 또는 다수의 포트에 연결되며, 각각의 포터들은 고유의 ID를 갖는다. 패킷들은 패킷을 목적지 구성요소로 전송하기 위해 중간 라우터들이 사용하는 목적지의 라우터와 포트 ID를 운반한다.

[0006] 라우팅 기술의 예시들은 모든 패킷에 대하여 A에서 B까지 똑같은 경로를 수반하는 결정론적 라우팅(deterministic routing)을 포함한다. 이러한 라우팅은 네트워크의 상태로부터 독립적이고, 기저 네트워크에 존재할 수 있는 다양한 경로들 간 로드(load) 균형을 맞추지 않는다. 그러나 이러한 결정론적 라우팅은 하드웨어 구현될 수 있으며, 패킷 순서를 유지하고, 네트워크의 레벨 교착 상태(deadlock)가 없을 수 있다. 최단 거리 라우팅은 대기 시간 (latency)를 최소화하며, 그러한 라우팅은 소스에서 목적지까지의 홉(hop)의 수를 감소시킨다. 이러한 이유로, 최단 거리는 두 구성요소 사이의 통신에 대한 최소 전력의 경로일 수도 있다. 차원-순서(dimension-order) 라우팅은 2D, 2.5-D, 및 3-D 메쉬 네트워크에서 결정론적 최단 거리(deterministic shortest path) 라우팅의 한 형태이다. 이 라우팅 방식(scheme)에서, 메시지들은 최종 목적지에 도달할 때까지 각 좌표들을 따라 특정한 순서로 전송된다. 예를 들어 3-D 메쉬 네트워크에서, 한 메시지는 그것이 목적지 라우터의 X-좌표와 동일한 X-좌표를 갖는 라우터에 도달할 때까지 우선 X 차원(X dimension)을 따라 먼저 전송될 수 있다. 그 다음, 메시지는 회전하여 Y 차원(Y dimension)을 따라 전송되고, 최종적으로 회전하여 Z 차원을(Z dimension) 따라 메시지가 최종 목적지 라우터에 도달할 때까지 이동한다. 차원 순서 라우팅은 최소 회전과 최단 경로를 갖는 라우팅일 수 있다.

[0007] 도 2a는 그림으로 설명된 2차원 메쉬에서의 XY 라우팅의 예시이다. 더 구체적으로, 도 2a는 XY 라우팅이 노드 '34'에서 노드 '00'까지의 라우팅을 나타낸다. 도 2a 예시에서, 각 구성요소는 오직 하나의 라우터의 하나의 포트에만 연결되어 있다. 패킷은 먼저 목적지 노드와 동일한 x-좌표를 갖는 노드 '04'에 도달할 때까지 x축(x-

axis)을 따라 전송된다. 그 다음, 패킷은 목적지 노드에 도달할 때까지 y축을 따라 전송된다.

[0008] 하나 이상의 라우터 또는 하나 이상의 링크가 존재하지 않는 이종 메쉬 토폴로지에서는, 특정한 소스 및 특정 노드 사이에서는 차원 순서 라우팅이 실현 가능하지 않을 수 있고, 대안이 되는 경로들이 선택될 수 있다. 대안이 되는 경로들은 가장 최단이거나 가장 최소한의 회전을 갖는 것은 아닐 수 있다.

[0009] 소스 라우팅 및 테이블 기반 라우팅은 NoC에서 사용되는 다른 라우팅 옵션들이다. 적응 라우팅은 네트워크의 상태를 기반으로 동적으로 두 지점 간의 경로를 바꿀 수 있다. 이러한 형태의 라우팅은 분석하고 시행하기 복잡할 수 있다.

[0010] NoC 인터커넥트는 다수의 물리적 네트워크를 포함할 수 있다. 각각의 물리적 네트워크상에서 다수의 가상 네트워크가 존재할 수 있으며, 서로 다른 가상의 네트워크에서는 서로 다른 종류의 메시지가 전송될 수 있다. 이 경우, 각각의 물리 링크 또는 채널에서 다수의 가상 채널이 존재하며, 가상 채널 각각은 양끝 포인트 모두에서 전용 버퍼를 가질 수 있다. 주어진 어느 클록 주기(cycle)에서, 단지 하나의 가상 채널만이 물리적 채널에서 데이터를 전송할 수 있다.

[0011] NoC 인터커넥트는 웜홀(wormhole) 라우팅을 사용할 수 있으며, 이 웜홀 라우팅에서는 큰 메시지 또는 패킷은 플릿(flits)(흐름 제어 디지트(digit)라고도 한다)으로 알려진 작은 조각으로 쪼개진다. 제1 플릿은 헤더 플릿으로(header flit) 페이로드 데이터와 함께 중요 메시지 레벨 정보(key message level info)와 패킷의 라우트(route)에 대한 정보를 보유하며, 이 메시지와 연관된 모든 후속 플릿에 대한 라우팅 양식을 마련한다. 선택적으로, 하나 또는 더 많은 본체 플릿은 헤드 플릿을 뒤따르며, 본체 플릿은 나머지 데이터 페이로드를 보유한다. 최후 플릿은 꼬리 플릿으로서 마지막 데이터 페이로드를 보유할 뿐만 아니라, 메시지에 대한 연결을 폐쇄하기 위해 몇몇 부기(bookkepping)를 수행한다. 웜홀 흐름 제어에서, 가상 채널이 종종 구현된다.

[0012] 물리적 채널은 가상 채널(VS)이라고 불리는 복수 개의 독립적인 논리 채널로 시간 구획된다. VC는 패킷을 라우팅하기 위해 다수의 독립 경로를 제공하지만, 물리적인 채널 상에서 시간-다중화된다. 가상 채널은 채널에 대해 패킷의 플릿들의 처리를 조정하는데 필요한 상태를 유지한다. 최소한으로, 라우트의 다음 홉을 위한 현재 노드의 출력 채널과 가상 채널의 상태(유휴 상태, 리소스 대기 상태 또는 활성화 상태)를 식별한다. 또한, 가상 채널은 현재 노드에 버퍼되는(buffered on) 패킷의 플릿들에 대한 포인터와 다음 노드에서 사용 가능한 플릿 버퍼 수를 포함할 수 있다.

[0013] "웜홀"이라는 용어는 메시지가 채널을 통해 전송되는 방식에 사용된다. 다음 라우터의 출력 포트가 너무 짧은 경우, 전체 메시지가 도달하기 전에 수신된 데이터가 헤드 플릿에서 변환될 수 있다. 이는, 라우터가 헤드 플릿이 도착하면 신속하게 라우터를 설정하고 이어서 대화의 나머지로부터 옵트 아웃(opt out)할 수 있다. 메시지가 플릿 단위로 전송되기 때문에, 메시지는 다른 라우터들에서 그 경로를 따라서 여러 개의 플릿 버퍼를 점유할 수 있고, 이는 웜(worm) 모양의 이미지를 생성한다.

[0014] 다양한 말단 포인트들 사이의 통신량에 기초하여 그리고 다양한 메시지들을 위해 사용되는 물리적 네트워크 및 라우트에 기초하여, NoC 인터커넥트의 다양한 물리적 채널들은 다양한 레벨의 로드(load)와 정체가 발생할 수 있다. NoC 인터커넥트의 다양한 물리적 채널의 용량(capacity)은 채널 폭(물리적 배선의 개수) 및 동작하는 클록 주파수(clock frequency)에 의해 결정된다. NoC의 다양한 채널들은 다른 클록 주파수들에서 작동할 수 있고, 다양한 채널들은 당해 채널의 대역폭 조건에 따라 다른 폭을 가질 수 있다. 채널의 대역폭 조건은 채널을 이동하는(traverse) 흐름 및 그 대역폭 값에 의해 결정된다. 다양한 NoC 채널을 이동하는 흐름은 다양한 흐름에 의해 점유되는 라우트에 의해 영향을 받는다. 메쉬이나 토러스 NoC에서는, 소스 및 목적지 노드 사이의 홉의 길이 또는 개수가 동일한 라우팅 경로가 여러 개 존재할 수 있다. 예를 들어, 도 2b 에서, 노드 34 및 00 사이의 표준 XY 라우트 외에도 YX 라우트(203) 또는 소스에서 목적지까지 두 번 이상 방향 전환을 하는 다중 턴(multi-turn) 라우트(202)와 같은 추가 라우트를 사용할 수 있다.

[0015] 다양한 트래픽 플로우(flow)에 대한 정적으로 할당된 라우트를 갖는 NoC에서, 다양한 채널에서의 로드는 다양한 흐름에 대해 지능적으로 라우트를 선택함으로써 제어될 수 있다. 매우 큰 트래픽 플로우와 상당히 다양한 경로(path diversity)가 존재할 때, 모든 NoC의 채널들의 로드가 거의 균일하게 균형이 되도록 라우트가 선택될 수 있고 이로써 단일 포인트의 교착상태를 피할 수 있다. 일단 라우트가 선택되면, NoC 채널 폭은 채널 상의 흐름의 대역폭 수요에 기반하여 결정될 수 있다. 불행하게도 채널 폭은 타이밍 또는 배선 혼잡(wiring congestion)과 같은 물리적인 하드웨어 디자인 제한으로 인해 임의로 커질 수 없다. 최대 채널 폭이 제한될 수 있고 이에 따라 임의의 한 NoC 채널의 최대 대역폭에 제한을 가하게 된다.

[0016] 또한, 메시지가 짧다면, 물리적으로 더 넓은 채널은 더 높은 대역폭을 달성하는데 도움이 되지 않을 수 있다. 예를 들어, 패킷이 64-비트의 폭의 단일 플릿 패킷이라면, 아무리 채널이 넓더라도, 그 채널은 모든 패킷이 비슷하다면 주기당 64-비트만을 전달할 것이다. 따라서, NoC에서 채널 폭은 또한 메시지 크기에 의해서도 제한된다. 최대 NoC 채널 폭에 대한 이와 같은 제한으로, 채널이 라우트의 균형에도 충분한 대역폭을 가지지 않을 수 있다.

[0017] 전술한 대역폭의 문제를 해결하기 위해, 여러 개의 병렬의 물리적 NoC가 사용될 수 있다. 각 NoC는 층으로 불릴 수 있고 이로써 여러 층의 NoC 구조를 형성한다. 호스트가 NoC 층에 메시지를 주입한 후, 메시지는 동일한 NoC 층 상의 목적지로 라우팅 되고, 메시지는 NoC 층에서 호스트로 전달된다. 따라서, 각 층은 서로에 대해서 어느 정도 독립적으로 동작하고, 층들 사이의 상호작용은 단지 메시지의 삽입 및 추출 시에만 일어난다. 도 3a는 두 층의 NoC를 도시한다. 여기서 NoC 두 층은 좌측 및 우측에 서로 인접하게 도시되었으며, 좌측 및 우측 도면 모두에 복제된 NoC에 호스트들이 연결된다. 호스트는 서로 다른 층의 두 개의 라우터에 연결된다. 예를 들어, 호스트는 R1으로 표시된 제1 층의 라우터와 연결되고, R2로 표시된 제2 층의 라우터와 연결된다. 이 예에서, 다층 NoC는 3D NoC와 다르다. 즉, 다층이 하나의 실리콘 다이(die)에 존재하고 높은 동일한 실리콘 다이 상의 호스트들 사이의 대역폭 수요를 충족하는데 사용된다. 메시지는 한 층에서 다른 층으로 가지 않는다. 명확성을 위해서, 본 출원은, NoC 들이 서로에 대해서 수직으로 도시되는 3D NoC와 구별하기 위해, 다층 NoC에 대해 수평으로 좌측 및 우측에 도시하는 것을 사용할 것이다.

[0018] 도 3b에서는, 도시된 바와 같이 호스트는 각 층의 라우터 R1 및 R2에 각각 연결된다. 각 라우터는 방향 포트(301)를 사용하여 자신이 속한 층에서 다른 라우터에 연결되고, 주입 및 추출 포트(302)를 사용하여 호스트에 연결된다. 브릿지-로직(303)은 나가는 메시지를 위한 NoC 층을 결정하기 위해 호스트와 두 NoC 층 사이에 위치하고, 호스트로부터 NoC 층으로 메시지를 보내고, 또한 두 NoC 층들로부터 오는 메시지들 사이의 중재(arbitration) 및 다중화(multiplexing)를 수행하고 호스트로 전달한다.

[0019] 다층 NoC에서, 필요한 층의 개수는 여러 요인, 예를 들어, 시스템의 모든 트래픽 플로우의 전체 대역폭 조건, 다양한 흐름에 사용되는 라우트, 메시지 크기 분포, 최대 채널 폭 등에 의존할 수 있다. NoC 인터커넥트에서 NoC 층의 개수가 디자인으로 정해지면, 다른 메시지들 및 트래픽 플로우들이 다른 NoC 층들에서 라우팅 될 수 있다. 또한, 다른 층들이 라우터, 채널 및 연결의 개수에서 다른 토폴로지를 갖도록 NoC 인터커넥트를 디자인할 수 있다. 다른 층의 채널들은 채널을 이동하는 흐름 및 대역폭 조건에 기하여 다른 폭을 가질 수 있다. 이렇게 다양한 디자인을 선택하면 특정 시스템에 대한 라우터, 채널 및 인터커넥션(interconnection)의 조합을 결정하는 것이 어려워지며, 시간이 많이 소모되는 수동 프로세스로 인하여 최적이 아닌 비효율적인 디자인이 될 수 있다.

[0020] 시스템 온 칩(SoC, System on Chips)은 점점 더 많은 수의 표준 프로세서 코어, 메모리 & I/O 하위시스템 및 특수화된 가속 IP를 통합하여, 점차 복잡해지고 기능이 다양해지며 성능이 향상되고 있다. 이러한 복잡성을 해결하기 위하여, SoC 구성요소를 연결하는 네트워크 온 칩(NoC) 방식이 널리 보급되고 있다. NoC는 많은 구성요소 및 인터페이스에 대한 연결을 제공할 수 있고, 동시에 높은 수준의 스펙에서 자동 생성되어 신속한 설계를 가능하게 한다. 이 스펙은 연결, 대역폭 및 대기시간 측면에서 SoC의 인터커넥트 요구를 설명할 수 있다. 그 외에도 다양한 구성요소의 위치, 프로토콜 정보, 클로킹 및 전원 도메인 등과 같은 정보가 제공될 수 있다. NoC 컴파일러는 이러한 스펙을 이용하여 SoC에 대한 NoC를 자동으로 설계할 수 있다. 많은 NoC 컴파일러가 트래픽 스펙에 맞는 NoC를 자동으로 합성하도록 관련 기술에서 소개되었다. 이러한 플로우 설계에서는, 합성된 NoC를 시뮬레이션하여 다양한 작동 조건에서 성능을 평가하고 스펙을 충족하는지 여부를 결정한다. 이는 NoC 스타일 인터커넥트가 분산 시스템이고, 로드에서 동적 성능 특성을 정적으로 예측하기 어려우며, 다양한 매게 변수에 대해 매우 민감하기 때문이다.

[0021] 도 4a는 2개의 호스트 및 2개의 플로우를 갖는 예시적인 시스템(400)을 도시한 것으로, 본 발명의 일 실시 예에 따른 플로우 레벨 스펙으로 표현된다. 일반적으로, 이러한 플로우 레벨 스펙은 그래프의 각 노드가 네트워크의 호스트이고, 에지가 한 노드에서 다른 노드로 전송되는 트래픽을 나타내는, 에지 가중치 다이그래프(edge-weighted digraph) 형태이다. 또한, 가중치는 트래픽의 대역폭을 나타낼 수 있다. 때때로, 이러한 스펙은 각 플로우에 대한 대기 시간 요구가 있으며, 이는 전송 시간의 제한을 나타낸다. 시스템(400)은 CPU(402)와 같은 제1 호스트와 메모리(404)와 같은 제2 호스트의 연결을 제1 호스트와 제2 호스트 사이의 두 개의 트래픽 플로우(406, 408)로 나타내며, 여기서, 제1 플로우는 CPU(402)에서 메모리(404)로의 '로드 요청(load request)'이고, 제2 플로우는 메모리(404)에서 CPU(402)로 회신되는 '로드 데이터(load data)'이다.

[0022] 이러한 트래픽 플로우 정보는 NoC의 스펙에 저장되며, NoC를 설계하고 시뮬레이션하는데 사용될 수 있다. 플로우 레벨 정보를 저장하는 스펙은 이하 플로우 레벨 스펙(flow-level specification)으로 지칭될 수 있다. 플로우 레벨 스펙은 언급되지 않은 다른 제한 사항 외에 추가적으로 다음의 두 가지 제한 사항이 있을 수 있다. 첫 번째 제한 사항은, 플로우 레벨 스펙에 포함된 정보가 NoC를 통해 SoC의 호스트들 사이의 교착 상태(deadlock)가 발생하지 않는 라우팅을 만드는데 충분하지 않을 수 있다는 것이다. 플로우 레벨 스펙에는 다른 호스트의 포트 간 외부 의존성에 대한 정보가 포함되어 있지만, 호스트 및/또는 메시지/패킷의 내부 의존성에 대한 정보는 포함되어 있지 않다. 두 번째 제한 사항은, 플로우 레벨 스펙에서 플로우로 표시되는 점대점(point to point) 트래픽을 이용하여 수행되는 네트워크 시뮬레이션이 충분하지 않거나 상호 의존성(inter-dependency) 정보와 같은 누락된 정보로 인해 정확하지 않을 수 있는 것이다. 플로우 기반 시뮬레이션을 통해 각 호스트는 다른 호스트의 동작과 독립적으로 패킷을 전송할 수 있다. 요청/응답 프로토콜로 인한 트래픽 상관관계(correlation)는 네트워크 동작에 중요한 영향을 미칠 수 있다.

[0023] 따라서, 더 나은 설계 및 시뮬레이션을 위하여, 트래픽 플로우 정보가 포함된 스펙을 제공할 수 있는 NoC가 요구된다. 또한, 트래픽의 대역폭 및 대기 시간을 특정하고 메시지들 간의 상호 의존성을 특정하며, 상호 의존성을 포착(capture)하기 위하여, 통합 NoC(integrated NoC) 스펙, 설계 및 성능 평가를 위한 트래픽 스펙을 이용하는 트랜잭션 기반 스펙(transaction-based specification)이 요구된다. 여기서, 트래픽은 트랜잭션들로 특정되고, 각각의 트랜잭션은 다중 점대점 홉(multiple point-to-point hops)을 포함할 수 있다.

발명의 내용

해결하려는 과제

[0024] 본 발명은 트랜잭션 트래픽 스펙으로부터 네트워크 온 칩(NoC) 기반 시스템을 생성(generating)/창작(creating)/설계(designing)하기 위한 명령어를 저장하는 비일시적 컴퓨터 판독 가능 매체에 관한 것이다. 일 예로, 이러한 트랜잭션 트래픽 스펙은 NoC 기반 시스템의 설계 및 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내는 복수의 파라미터를 포함할 수 있다. 일 실시 예로, NoC 기반 시스템의 NoC는 스펙 요구를 충족시키기 위해 트랜잭션 트래픽 스펙으로부터 생성될 수 있다.

과제의 해결 수단

[0025] 본 발명의 일 실시 예에 따른 트랜잭션 트래픽 스펙은 NoC를 생성하고 최적화하기 위한 정보를 제공할 수 있고, 종래의 플로우 레벨 스펙의 플로우 레벨 엔트리와 비교할 때 트랜잭션 기반 엔트리를 포함할 수 있다. 여기서, 트랜잭션 트래픽 스펙은 NoC를 설계 및/또는 시뮬레이션하는데 사용될 수 있는 연결성(connectivity), 의존성(dependency), 대역폭, 대기 시간, 트래픽 지터(traffic jitter), 속성 및 다른 필요한 정보에 관한 정보를 나타내는/정의하는 복수의 파라미터 및 정보를 포함할 수 있다.

[0026] 본 발명의 일 실시 예에 따른 구현은 트랜잭션 트래픽 스펙을 이용하는 NoC 설계 및 시뮬레이션을 위한 시스템을 제공한다. 또한, 본 발명의 일 실시 예에 따른 구현은 설계 성능 시뮬레이션이 수행될 수 있는 트래픽 추적 파일의 생성을 용이하게 할 수 있는 트랜잭션 트래픽 스펙을 제공할 수 있다.

[0027] 본 발명의 일 실시 예에서 전술한 목적 및 다른 목적, 특징 및 이점은 첨부된 도면에 도시된 것과 같이 예시적인 구현에 대한 다음의 상세한 설명에 의하여 보다 명백해질 것이다. 여기서, 동일한 참조번호는 일반적으로 동일한 부분을 나타낼 수 있다.

도면의 간단한 설명

[0028] 도 1a 내지 도 1d는 각각 양방향 링, 2차원 메쉬, 2차원 토러스 및 3차원 메쉬 NoC 토폴로지를 나타낸다.

도 2a는 종래의 2차원 메쉬에서 XY 라우팅의 일 예를 나타낸다.

도 2b는 소스 노드 및 목적지 노드 사이의 서로 다른 세 개의 라우트를 나타낸다.

도 3a는 종래의 두 층 NoC 인터커넥트의 일 예를 나타낸다.

도 3b는 종래의 호스트 및 다중 NoC 층 사이의 브릿지 로직을 나타낸다.

도 4a는 본 발명의 플로우 레벨 스펙의 일 실시 예에 따른 두 개의 호스트 및 두 개의 플로우를 포함하는 시스템을 나타낸다.

도 4b는 본 발명의 일 실시 예에 따른 하나의 트랜잭션으로써 두 개의 호스트 사이에 두 개의 플로우를 포함하는 시스템을 나타낸다.

도 5는 본 발명의 일 실시 예에 따른 호스트 간의 트랜잭션을 나타낸다.

도 6a는 트랜잭션을 위해 구성된 체인의 일 예를 나타낸다.

도 6b는 본 발명의 일 실시 예에 따른 체인 생성 알고리즘을 이용하여 생성된 트랜잭션 체인을 나타낸다.

도 7은 본 발명의 일 실시 예에 따른 예시적인 구현이 수행될 수 있는 컴퓨터 시스템을 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0029] 이하의 상세한 설명은 본 출원의 도면들 및 예시적인 구현들에 대해 상세한 명을 제공한다. 도면들 사이의 중복 구성들에 대해 설명 및 참조번호는 명확성을 위해 생략된다. 본 발명에 대한 설명에서 사용된 용어들은 예로서 제공된 것으로서 제한적으로 사용된 것은 아니다. 예를 들어, "자동적"이라는 용어의 사용은, 완전 자동적인 구현이거나 본 출원의 구현들을 실행하는데 통상의 지식을 가진 자의 원하는 구현에 따라서, 구현의 어떤 양상들에 대한 사용자 또는 관리자 제어를 수반하는 반-자동적인 구현을 포함한다.

[0030] 본 명세서에서 설명된 예시적 구현들은 복수의 파라미터를 포함하는 트랜잭션 트래픽 스펙으로부터 네트워크 온 칩(NoC) 기반 시스템을 생성/창작/설계하기 위한 명령어를 저장하는 비일시적 컴퓨터 판독 가능 매체에 관한 것이다. 예시적인 구현에서, 트랜잭션 트래픽 스펙은 NoC 기반 시스템의 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내는 복수의 파라미터를 포함한다. 여기서, NoC 기반 시스템의 NoC는 스펙 요구를 충족시키기 위해 스펙으로부터 생성될 수 있다.

[0031] 본 명세서에서 설명된 예시적 구현들은 NoC를 생성하고 최적화하기 위한 정보를 갖는 트랜잭션 트래픽 스펙에 관한 것이다. 여기서, 트랜잭션 트래픽 스펙은 통합된 NoC 스펙, 설계 및 성능 평가를 위한 트래픽 스펙을 이용할 수 있다. 트랜잭션 트래픽 스펙은 플로우 레벨 스펙의 플로우 레벨 엔트리와 비교할 때 하나 이상의 트랜잭션 기반 항목을 포함할 수 있다. 트랜잭션 트래픽 스펙은 NoC를 설계 및/또는 시뮬레이션하는데 사용될 수 있는 연결성, 의존성, 대역폭, 대기 시간, 트래픽 지터, 속성에 관한 정보 및 다른 필수 정보를 포함하는 정보를 포함할 수 있으며, 다만, 이에 한정되는 것은 아니다. 트랜잭션 트래픽 스펙은 NoC에서 패킷을 주입하는 성능 시뮬레이션을 수행할 수 있는 트래픽 추적 파일의 생성을 가능하게 한다.

[0032] 제안된 시스템의 예시적 구현은 트랜잭션 트래픽 스펙을 제공하는 NoC에 관한 것이며, 스펙은 NoC 및 NoC의 성능/동작의 더 나은 설계 및 시뮬레이션에 이용될 수 있다.

[0033] 전술한 바와 같이, 잘 알려진 예시적 구현에서, NoC 트래픽 요구들은 플로우 레벨 스펙을 이용하여 특정될 수 있다. 일반적으로, 이러한 유형의 스펙은 에지 가중치 다이그래프(edge-weighted digraph)의 형태로 나타낼 수 있고, 여기서, 그래프의 각 노드는 네트워크의 호스트일 수 있다. 에지 가중치 다이그래프의 각 에지는 한 노드에서 다른 노드로 전송되는 트래픽을 나타내며, 가중치는 해당 트래픽의 대역폭을 나타낸다. 때때로, 이러한 유형의 스펙은 각 플로우에 대한 대기 시간 요구가 나타나며, 이는 전송 시간의 제한을 나타낸다.

[0034] 본 명세서에서 설명되는 예시적 구현들은 트랜잭션 기반 트래픽 스펙에 관한 것으로, 스펙은 메시지 및/또는 호스트 간의 상호 의존성(inter-dependency) 및 트래픽의 대역폭, 대기 시간을 특정할 수 있다. 상호 의존성을 포착(capture)하기 위해, 트래픽을 트랜잭션으로 특정할 수 있으며, 각 트랜잭션은 다중 점대점 홉(multiple point-to-point hops) 및 상호 의존성의 세부 사항들을 포함할 수 있다. 또한, 예시적 구현은 프로세스를 실행하기 위한 명령들을 저장하는 비일시적 컴퓨터 판독 가능 매체에 관한 것으로, 명령들은 스펙으로부터 네트워크 온 칩(NoC) 기반 시스템을 생성하는 단계를 포함할 수 있다. 제안된 스펙은 NoC 기반 시스템의 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내는 복수의 파라미터를 포함할 수 있으며, NoC 기반 시스템의 NoC는 스펙 요구를 충족시키기 위해 스펙으로부터 생성될 수 있다.

[0035] 예시적 구현에서, 복수의 파라미터는 NoC의 연결성 정보, 의존성 정보, 대역폭 정보 및 속성 정보 중 하나 또는 NoC의 연결성 정보, 의존성 정보, 대역폭 정보 및 속성 정보의 조합을 포함할 수 있다. 예시적 구현에서, 대역폭 정보는 지정된 시간 동안 등시성 트래픽(isochronous traffic)에 대한 대역폭을 보장하기 위한 하나 이상의 파라미터를 포함할 수 있다. 또한, 대역폭 정보는 트래픽 지터를 나타내는 하나 이상의 파라미터를 포함할 수 있다. 의존 정보는 NoC의 복수의 요소들 간의 의존성 또는 NoC에서 복수의 유형의 메시지들 간의 의존성을 나타낼 수 있다. 또한, 의존 정보는 NoC의 복수의 요소들 간의 비의존성 또는 NoC의 복수의 유형의 메시지들 간의

비의존성을 나타낼 수 있다.

[0036] 예시적 구현에서, 대역폭 정보는 하나 이상의 트랜잭션의 메시지의 대역폭 요구, 하나 이상의 트랜잭션의 메시지의 서비스 품질 요구, 하나 이상의 트랜잭션의 메시지의 대기 시간 요구, 하나 이상의 트랜잭션의 대기 시간 요구 중 하나 또는 그 조합을 포함할 수 있다. 다른 구현에서, 연결 정보는 NoC 기반 시스템의 트래픽 시뮬레이션을 나타내는 NoC 기반 시스템의 하나 이상의 메시지에 대한 트랜잭션 정보, NoC의 메시지 전송을 위한 프로토콜 정보, NoC의 메시지 시퀀스를 나타내는 순서 정보 중 하나 또는 그 조합을 포함할 수 있다. 한편, 예시적 구현에서 트랜잭션 정보는 하나 이상의 트랜잭션을 포함할 수 있고, 하나 이상의 트랜잭션 각각은 일련의 홉을 나타낼 수 있으며, 일련의 홉에서 각각의 홉은 소스 에이전트로부터 목적 에이전트로의 하나 이상의 메시지를 나타낼 수 있다. NoC의 생성은 하나 이상의 트랜잭션 각각을 위한 일련의 홉에 따라 소스 에이전트로부터 목적 에이전트로의 각 홉을 위한 하나 이상의 메시지를 전송하기 위해 트랜잭션 정보에 기초하여 NoC 기반 시스템을 구성하는 단계를 포함할 수 있다. 다른 예시적 구현에서, 트랜잭션 정보는 하나 이상의 트랜잭션의 적어도 하나의 서브 셋에 대한 시스템 레벨 시뮬레이션이 가능하도록 구성될 수 있다. 구현에서, 속성 정보는 하나 이상의 트랜잭션의 메시지 크기, 하나 이상의 트랜잭션의 메시지 우선순위, 하나 이상의 트랜잭션의 메시지의 서비스 품질(QoS) 및 하나 이상의 트랜잭션의 메시지의 주문 요구 중 하나 또는 그 조합을 나타내는 메시지 속성 정보를 포함할 수 있다. 본 발명의 시스템은 스펙의 복수의 파라미터에 기반한 NoC의 시뮬레이션을 포함할 수 있다. 또한, 본 발명의 시스템은 시뮬레이션을 수행하기 위한 스펙의 복수의 파라미터로부터 트래픽 추적 데이터베이스를 생성하는 단계를 더 포함할 수 있으며, 데이터베이스는 하나 이상의 트래픽 추적 파일을 포함할 수 있고, 각 트래픽 추적 파일은 트랜잭션을 개시하기 위한 정보를 나타낼 수 있다. 트랜잭션을 개시하는 정보는 스펙의 복수의 파라미터에 기초하여 결정될 수 있다. 또한, 본 발명의 명령들 및/또는 시스템은 복수의 파라미터 및 통계 모델에 기초하여 복수의 트랜잭션으로부터 트랜잭션을 선택하는 단계를 더 포함할 수 있다.

[0037] 또 다른 실시 예에서, 대역폭 정보는 트랜잭션 레이트(rate), 트랜잭션의 메시지 레이트 및 NoC 기반 시스템의 하나 이상의 호스트의 레이트 중 하나 또는 그 조합을 나타낼 수 있다.

[0038] 예시적 구현에서, 본 발명은 스펙으로부터 구성되는 네트워크 온 칩(NoC) 기반 시스템을 더 포함할 수 있다. 스펙은 복수의 파라미터를 포함할 수 있고, 여기서, 복수의 파라미터는 NoC 기반 시스템의 성능 시뮬레이션이 가능하도록 구성된 정보를 나타내며, NoC 기반 시스템의 NoC 스펙 요구를 충족시키기 위해 스펙으로부터 생성될 수 있다. 복수의 파라미터는 NoC의 연결 정보, 의존 정보, 대역폭 정보 및 속성 정보 중 하나 또는 그 조합을 포함할 수 있으며, 다만, 이에 한정되는 것은 아니다. 대역폭 정보는 특정 시간 동안 등시성 트래픽에 대한 대역폭을 보장하기 위한 하나 이상의 파라미터를 포함할 수 있다. 또한, 대역폭 정보는 트래픽 지터를 나타내는 하나 이상의 파라미터를 포함할 수 있다.

[0039] 도 4b는 본 발명의 일 실시 예에 다른 하나의 트랜잭션(456)으로써, 두 개의 호스트(452, 454) 간의 두 개의 플로우가 나타나는 시스템(450)을 나타낸다. 따라서, 도 4a의 두 개의 플로우는 여기서 하나의 트랜잭션으로 나타날 수 있다. 예시적 구현에서, 스펙은 두 개의 홉 사이의 하나의 트랜잭션으로써 두 개의 메시지를 표시하여 두 개의 메시지 간의 관계를 유지할 수 있다. 따라서, 예시적 구현에서, 스펙은 두 개 이상의 호스트 간의 이러한 트랜잭션의 조합을 포함할 수 있다.

[0040] 플로우 레벨 스펙 대신에 트랜잭션 기반 엔트리를 트래픽 스펙에 이용하여 트래픽을 특정하는 것은 이전에 언급되거나 기존에 알려진 플로우 레벨 트래픽 스펙의 한계를 극복하는데 도움이 될 수 있다. 또한, 트랜잭션 기반 엔트리를 이용하여 설계된 스펙은 트랜잭션 트래픽 스펙과 상호교환될 수 있다. 트랜잭션 트래픽 스펙을 이용하여 소스 라우팅 시스템에서 교착 상태를 피할 수 있고, 여기서, 교착 상태 방지 알고리즘은 트랜잭션의 홉들 간의 암시적 인코딩(implicitly encoded)된 내부 의존성을 이용하여 교착 상태가 없는 NoC 설계를 구축할 수 있다. 또한, 트랜잭션은 NoC의 시뮬레이션을 보다 정확하게 할 수 있다. 요청이 도달하여 트리거된 응답 메시지는 간단한 스펙으로 인하여 좀 더 복잡한 트래픽 동작이 될 수 있다. 또한, 트랜잭션 시뮬레이션은 단순한 점대점 트래픽 모델에서 발견되지 않는 잠재적인 시스템 병목 현상 및 교착 상태 문제를 부각시킬 수 있다. 실제 SoC 스펙 및 작업 부하에 대한 광범위한 평가를 통해 예시적 구현과 관련된 시뮬레이션을 수행하면 신속한 설계 및 평가를 수행할 수 있고, 간단한 점대점 NoC 시뮬레이션과 비교할 때 NoC 성능을 훨씬 효과적으로 평가할 수 있다. 스펙의 예시적 구현은 NoC의 트래픽 요구의 높은 레벨의 픽쳐(picture)를 제공할 수 있다. 스펙 언어는 아키텍트(architect)가 다중 홉 트랜잭션 그룹의 구조를 설명할 수 있게 하고, 트랜잭션 전체의 속성 및 트랜잭션의 각 홉의 속성을 지원할 수 있다.

[0041] 또한, 예시적 구현에서, 트랜잭션 트래픽 스펙은 NoC를 설계, 생성 및 최적화하는데 이용할 수 있는 트랜잭션

정보를 포함할 수 있다. 또한, 트랜잭션 트래픽 스펙은 트래픽 추적 파일을 생성하는데 이용될 수 있다. 도 4a에 도시된 바와 같이, 알려진 스펙들은 에이전트들 간의 페어와이즈(점대점) 연결을 제공하여 NoC 트래픽을 특정할 수 있다. 그러나 호스트들 간의 필수 연결이 항상 점대점일 필요는 없다. 즉, 패킷이 항상 한 호스트에서 다른 호스트로 직접 전송되는 것은 아니며, 중간에 다중 호스트 또는 관련된 호스트가 더 있을 수 있고, 이에 따라 다중 의존성(multiple dependencies)이 생성될 수 있다. 또한, 다중 패킷/메시지 자체가 상호 의존성을 가질 수도 있다. 전술한 바와 같이, 주어진 NoC 인터커넥트 시스템은 소스 호스트, 하나 이상의 중간 호스트 및 목적 호스트를 포함할 수 있다. 예를 들어, 소스 호스트 자체가 목적 호스트일 수도 있으나, 트랜잭션을 완료하기 위해 여러 중간 호스트가 필요할 수도 있다. 트랜잭션 트래픽 스펙은 상호 의존성, 중간 홉 및 상호 의존성, 중간 홉의 상태와 같은 모든 트랜잭션 정보를 포함할 수 있으며, 이들은 종래의 플로우 레벨 스펙에서는 빠져있다.

[0042] 예시적 구현에서, 트랜잭션 트래픽 스펙은 연결성, 호스트/홉 간의 상호 의존성, 메시지/패킷/플릿 간의 상호 의존성, 대역폭, 대기 시간 및 NoC 트래픽에 대한 메시지 속성에 관한 정보를 포함할 수 있다. 예를 들어, 의존 정보는 차단/실패의 경우 소스 노드에서 목적 노드로 트랜잭션을 중단 및/또는 방해할 수 있는 중간 홉에 대한 정보를 포함할 수 있다. 다른 예시적 구현에서, 의존성은 메시지의 시퀀스 분석과 같은 적절한 방법을 이용하여 결정될 수 있다. 트래픽 스펙은 소스가 중간 홉에 의존하는 방법 또는 메시지/패킷/플릿 및/또는 중간 홉들 간의 비의존성에 대한 방법에 대한 명백한 의존성 정보를 포함할 수 있다.

[0043] 또 다른 예시적 구현에서, 트랜잭션 트래픽 스펙은 메시지 대기 시간, 트랜잭션 대기 시간, 각 홉 및/또는 전체 트랜잭션에 대한 대기 시간과 같은 대기 시간 및/또는 대역폭에 대한 정보를 포함한다. 또 다른 예시적 구현에서, 대기 시간은 상술한 하나 이상의 상호 의존성에 기초할 수 있다. 대역폭 정보에는 각 호스트와 홉 사이의 대역폭 요구 및 전체 트랜잭션에 대한 전체 대역폭 요구가 포함될 수 있다. 대역폭 정보는 서로 다른 홉 간의 링크 크기 및 NoC 호스트의 정보가 포함될 수 있다. 트래픽 스펙은 메시지 크기, 메시지 우선순위, 메시지 유형, 메시지 품질 및 메시지에 대한 다른 스펙 정보와 같은 메시지 속성에 대한 정보를 더 포함할 수 있다. 또한, 크기, 우선순위, 유형 및 품질과 같은 유사한 속성은 각 패킷 및/또는 패킷의 플릿에 대해 정의될 수 있다.

[0044] 또 다른 예시적 구현에서, 소스 라우팅 시스템에서 트랜잭션 트래픽 스펙을 이용하여, 트랜잭션 트래픽 스펙이 메시지 및/또는 NoC 호스트 간의 내부 의존성에 대한 세부사항을 포함하는 트랜잭션 레벨 스펙을 포함하므로, 잠재적 트래픽 교착 상태를 피할 수 있다. 예시적 구현에서, 교착 상태 방지 알고리즘은 교착 상태가 없는 NoC 설계를 구축하기 위해 트랜잭션의 홉들 간의 암시적 인코딩(implicitly encoded)된 트랜잭션 트래픽 스펙에 정의된 내부 의존성을 이용할 수 있다. 또한, 예시적 구현에서, 트랜잭션 트래픽 스펙은 NoC 트래픽의 정확한 시뮬레이션을 위해 이용될 수 있다. 트랜잭션 트래픽 스펙을 이용하는 트랜잭션 시뮬레이션은 잠재적인 시스템 병목 현상 및 교착 상태 문제가 강조될 수 있으며, 플로우 레벨 스펙을 이용하는 시뮬레이션에서는 잠재적인 시스템 병목 현상 및 교착 상태 문제가 강조되지 않을 수 있다.

[0045] 트랜잭션 트래픽 스펙의 예시적 구현은 NoC의 트래픽 요구에 대한 높은 레벨 뷰(view)/표현을 제공할 수 있다. 스펙 언어는 아키텍트가 다중 홉 트랜잭션 그룹의 구조를 설명할 수 있으며, 트랜잭션 전체의 속성 및 트랜잭션의 각 홉의 속성을 지원할 수 있다.

[0046] 예시적 구현에서, NoC를 설계하기 위하여 별도의 트랜잭션 트래픽 스펙이 이용될 수 있고, NoC 시뮬레이션을 위한 별도의 트랜잭션 트래픽 스펙이 이용될 수 있다.

[0047] 예시적 구현에서, 실제 SoC 스펙 및 작업 부하에 대한 광범위한 평가를 통해 시뮬레이션을 수행하면, 신속한 설계 및 평가 반복을 수행하면서 간단한 점대점 NoC 시뮬레이션과 비교할 때 NoC 성능을 훨씬 효과적으로 평가할 수 있다. 도 5는 본 발명의 일 실시 예에 따른 호스트들 간의 트랜잭션을 도시한다. 트랜잭션은 NoC 상의 CPU(502, 504) 사이에 구성된 하드웨어 캐시 코히런스 모듈(hardware cache coherence module)(506)과 두 개의 중앙 프로세싱 유닛 CPU-1(502), CPU-2(504)를 포함하는 시스템(500)을 참조하여 설명한다. 예시적 구현에서, 캐시 코히런스 모듈(506)이 요청된 데이터를 수신하지 않은 경우, 캐시 코히런스 모듈(506)은 CPU-1(502)로부터 캐싱된 로드 요청(508)을 수신하고 CPU-2(504)에 스눕(snoop) 요청(510)을 할 수 있다. 요청된 데이터가 CPU-2(504)의 캐시에 있을 때, CPU-2(504)는 캐시 코히런스 모듈(506)에 의해 수신되어 요청한 CPU-1(502)에 전송되는 스눕 데이터(512)로 응답할 수 있다. 전체 메시지 시퀀스는 도 5의 하나의 라인으로 표시된 하나의 트랜잭션(516)으로 표현될 수 있다.

[0048] 예시적 구현에서, CPU(502) 및 CPU(504)는 각각 CPU_1 및 CPU_2로 표현될 수 있고, 캐시 코히런스 모듈(506)은 스펙에서 트랜잭션을 설명하기 위해 CC로 표현될 수 있다. 유사하게, 스펙에서 트랜잭션을 설명하기 위해, 로드

요청을 전달하는 포트는 LD, 스눕 메시지 전달하는 포트는 SN, 스눕 응답을 수신하는 포트는 SN_RESP, 로드 데이터를 수신하는 포트는 LD_DATA로 표현될 수 있다.

[0049] 따라서, 트랜잭션의 메시지 시퀀스는 다음과 같이 설명할 수 있다:

[0050] pri 1 rates 0.05 0.1 latency 30 profile 0,1

[0051] CPU_1/ID CPU_2/ID → beats 1 CC/LD/SN →

[0052] beats 1 CPU_1/SN/SN_RESP CPU-2/SN/SN_RESP →

[0053] beats 4 CC/SN_RESP/LD_DATA →

[0054] beats 4 CPU_1/LD_DATA CPU_2/LD_DATA

[0055] 상기 트랜잭션의 실시 예는 네 가지 속성이 있다: 1) 우선순위, 2) 메시지 레이트, 3) 대기 시간 타겟(latency target), 4) 해당 트래픽 프로파일(profile). 트랜잭션의 실시 예에서, 'pri 1'은 트랜잭션의 우선순위를 1로 나타낼 수 있고, 여기서, 우선순위는 라우터에서 우선순위 기반 중재(priority-based arbitration)에 이용될 수 있다. "rate 0.05 0.1"은 트랜잭션이 시작되는 빈도를 나타내는 두 개의 메시지 레이트를 나타내기 위해 구성될 수 있다. 주파수는 사이클 당 메시지 수로 측정할 수 있다. 측정된 값은 인터페이스 폭, 메시지당 플릿 수 및 NoC의 클럭 레이트를 곱하여 대역폭 수치로 변환할 수 있다. 일 예로, 0.05의 레이트(rate of 0.05)는 평균 레이트 0.05, 즉, 20 사이클마다 하나의 메시지를 나타낼 수 있고, 레이트 0.1(rate 0.1)은 피크 레이트, 즉, 10 사이클마다 하나의 메시지를 나타낼 수 있다. 레이트는 포트의 용량을 분석하는 데 중요하다. 예를 들어, 1/3보다 큰 트랜잭션 레이트를 가진 3 비트의 메시지는 포트의 용량을 초과한다.

[0056] 스펙에 설명된 다음 트랜잭션 속성은 대기 시간 타겟(latency target)이다. 일 실시 예에 따른 스펙의 대기 시간 타겟은 "대기 시간 30"일 수 있고, 이는 필요한 시스템 성능을 얻기 위한 트랜잭션의 완료 대기 시간을 나타낸다. 일 실시 예에 따른 트랜잭션의 대기 시간 타겟은 30 사이클일 수 있다. 즉, 트랜잭션을 완료하는 데 30개보다 많은 사이클이 걸리면 시스템 성능에 부정적인 영향을 미칠 수 있다. 스펙에 설명된 마지막 트랜잭션 속성은 프로파일이다. 여기서, 각 트랜잭션은 임의의 수의 트래픽 프로파일의 구성 또는 트래픽 모드의 구성일 수 있다. 하나의 프로파일에서 트랜잭션만 활성화되면 다른 트래픽 모드 간에 쉽게 전환할 수 있다. 또한, 예시적 구현에서, 트랜잭션 트래픽 스펙은 상이한 홉의 입력 및 출력 포트에 대한 2가지 유형의 설명을 포함한다. 2가지 유형은 소스 및 목적지의 입력 및 출력 포트에 대한 하나의 유형 및 중간 홉의 입력 및 출력 포트에 대한 다른 유형을 포함한다. 예를 들어, 첫 번째 홉의 소스와 마지막 홉의 목적지는 호스트/포트로 특정될 수 있다. 이는 트랜잭션에 참여할 호스트와 사용할 포트를 나타낸다. 일 실시 예에 따른 트랜잭션에서 첫 번째 홉은 두 개의 가능한 포트, 즉, CPU_1의 LD 포트 또는 CPU_2의 LD 포트에서 시작하므로 트랜잭션 스펙에서는 CPU_1/LD 및 CPU_2/LD로 나타낼 수 있다.

[0057] 예시적 구현에서, 중간 홉은 CC/LD/SN과 같은 호스트/포트로 특정할 수 있다. 예를 들어, 첫 번째 포트는 메시지를 수신할 수 있는 LD와 같은 수신 포트이고, 두 번째 포트는 다음 메시지가 중간 홉으로부터 전송되는 발신 포트 SN이 될 수 있다. 즉, 메시지가 하나의 포트 LD에 도달하고 다른 포트 SN에 의해 전송된 것으로 설명될 수 있다. 일부 프로토콜은 홉 내에서 포트를 혼합할 수 있으므로 홉의 양쪽 끝에서 포트를 지정할 수 있다.

[0058] 제안된 트랜잭션 트래픽 스펙의 예시적 구현에서, 트랜잭션의 각 홉은 완전한 메시지를 구성하는 데이터 청크(chunks)/플릿의 수를 나타내는 "비트 1"과 같은 비트 속성을 나타낼 수 있다. 또한, 각 홉은 해당 메시지에 대한 패킷의 플릿 수를 포함할 수 있다. 예를 들어, 데이터를 전달하는 스눕 응답 및 로드 데이터 메시지는 비트 4에 의해 길이가 4 플릿일 수 있는 반면, 로드 메시지 및 스눕 메시지는 각각 하나의 플릿일 수 있다. 페어와이즈(pairwise) 대역폭 그래프는 레이트 및 비트에 포트의 폭(width)을 곱하여 트랜잭션 트래픽 스펙의 예시적 구현으로부터 계산될 수 있다.

[0059] 예시적 구현에서, 트랜잭션은 최적화될 수 있으며, 예를 들어, 요청, 응답, 관련 메시지 및 하나의 메시지에 대한 설명(description)을 갖는 프로토콜을 이용하는 트랜잭션이 다른 메시지를 생성할 수 있다. 이 경우, 두 메시지 모두에 대한 설명은 스펙에서 특정되지 않을 수 있다. 프로토콜의 정의에 기초하여 시퀀스의 다른 메시지가 도출될 수 있다. 다른 예시적 구현에서, 사용자는 메시지 정보 및 프로토콜 스펙에 기초하여 전체 설명을 생성할 수 있는 트랜잭션의 부분 설명을 제공할 수 있다.

[0060] 예시적 구현에서, 메시지를 메모리에 전송하는 CPU와 같은 공통 패턴을 갖는 다중 트랜잭션은 트랜잭션 체인으로써 트랜잭션 트래픽 스펙에 설명될 수 있다. 트랜잭션 트래픽 스펙의 트랜잭션 체인에 대한 설명은 다중 홉

(multi-hop) 트랜잭션의 효율적인 시뮬레이션이 가능하도록 이용될 수 있다.

[0061] 트래픽 체인

[0062] 트랜잭션은 각 홉이 단일 소스/목적지를 갖는 체인으로 나타낼 수 있다. 시뮬레이션을 시작할 때 모든 트랜잭션을 체인으로 변환함으로써 전송할 다음 메시지가 미리 결정되어 호스트의 동작 속도를 높일 수 있다. 트랜잭션을 여러 체인으로 변환하는 방법은 각 홉에서 구성 요소를 선택할 때마다 체인을 만드는 것이다. 트랜잭션은 도 6a에 도시된 것과 같은 하나의 체인을 생성할 수 있다. 코히런스 제어부는 데이터를 원하는 CPU에 스눕하지 않으며, 데이터를 요구하지 않은 CPU에 데이터를 보내지 않는다. 도 6a에 도시된 체인은 도 5에 도시된 트랜잭션을 위해 구성될 수 있다. 도 6a에 도시된 것과 같이, CPU-1(602-a) 및 CPU-1(602-b)은 하나의 CPU일 수 있으며, 설명의 편의를 위하여 통합된 CPU-1(602-a) 및 CPU-1(602-b)은 CPU-1(602)로 기재한다. 유사하게, 캐시 코히런스 모듈(604-a) 및 캐시 코히런스 모듈(604-b)은 하나의 캐시 코히런스 모듈일 수 있으며, 설명의 편의를 위하여 통합된 캐시 코히런스 모듈(604-a) 및 캐시 코히런스 모듈(604-b)은 캐시 코히런스 모듈(604)로 기재한다. 일 구현 예에서, CPU-1(602-a)은 캐시 코히런스 모듈(604-a)에 데이터 요청을 전송할 수 있고, 캐시 코히런스 모듈(604-a)은 다시 CPU-1 (602-b)에 요청을 전송할 수 있으며, 이것은 실제로 데이터를 요청한 CPU-1(602-a)에 스눕 요청을 전송하거나 CPU-1(602-a)에 데이터를 전송하는 의미이다. 전술한 바와 같이, 실시 예에서, 스눕 요청은 CPU-1(602-a) 대신 CPU-2(606)로 보내져야 한다. 전술한 트랜잭션에 대해, 코히런스 제어부(604)가 실제로 데이터를 요구한 CPU-1(602-a)에 다시 스눕할 이유가 없으며, 마찬가지로, 코히런스 제어부(604)는 데이터를 요청하지 않은 CPU-2(606)에 데이터를 전송하지 않아야 한다. 따라서, 생성된 트랜잭션 체인은 트랜잭션을 명확하게 나타내지 못하므로, 적절한 트랜잭션 체인을 생성하기 위하여 트랜잭션 트래픽 체인을 적절하게 나타낼 수 있는 체인 생성 알고리즘이 필요하다. 예시적 구현은 실제 시스템이 생성할 체인 생성에 대한 체인 생성 알고리즘을 이용한다. 체인 생성 알고리즘은 현재 메시지의 유형과 현재까지 만들어진 부분 체인을 기반으로 현재 홉의 잘못된 대상을 제거하기 위해 요청 vs. 응답의 구별을 이용할 수 있다. 이상적인 요청은 현재까지 생성된 체인의 일부가 아닌 새로운 목적지로 이동해야 하며, 예를 들어, 응답은 현재까지 생성된 체인의 일부인 목적지로 가야한다.

[0063] 이 알고리즘을 실시 예에 따른 트랜잭션에 적용하면 CPU-1 → CC → CPU-2 → CC → CPU-1의 체인 하나만 구성할 수 있다.

[0064] 도 6b는 본 발명의 일 실시 예에 따라 제안된 체인 생성 알고리즘을 이용하여 생성된 예시적인 트랜잭션 체인(650)을 나타낸다. CPU-1(652-a) 및 CPU-1(652-b)은 이하 CPU-1(652)로 총칭되는 단일 CPU-1이다. 유사하게, 캐시 코히런스 모듈 (654-a)및 캐시 코히런스 모듈(654-b)은 이하 캐시 코히런스 모듈(654)로 총칭되는 단일 캐시 코히런스 모듈이다. 예시적 구현에서, CPU-1(652-a)은 654-a로 표현되는 캐시 코히런스 모듈(654)에 데이터 요청을 전송할 수 있고, CPU-2(656)는 스눕 요청을 캐시 코히런스 모듈(654-b)로 나타낸 캐시 코히런스 모듈(654)로 다시 전송할 수 있다. 캐시 코히런스 모듈(654-b)로 나타낸 캐시 코히런스 모듈(654)은 652-b로 나타낸 CPU-1(652)에 데이터를 전송한다. 따라서, CPU-1 → Cache Coherency Module → CPU-2 → Cache Coherency Module → CPU-1 체인이 생성된다.

[0065] 전술한 바와 같이, 캐시 코히런스 모듈(654)로부터의 스눕 메시지는 원래 CPU-1(652)에 의해 개시되었던 요청이므로, CPU-1에 전송될 수 없고, 로드 데이터 메시지는 CPU-2(656)에 의해 전송된 응답이므로, CPU-2(656)으로 전송될 수 없다.

[0066] 예시적 구현에서, 체인 생성 알고리즘은 보존될 수 있다. 예를 들어, 어떤 시점에서 트래픽 스펙이 다음 홉을 허용하지 않으면 필터링이 수행되지 않고, 스펙이 가능한 한 최상으로 유지되려고 한다. 대신에, 요청 및 응답의 형태에 맞지 않는 트랜잭션에 대한 체인을 생성하지 않을 수도 있다. 이 경우, 입력이 체인에 입력될 때 알고리즘이 사용자의 의도를 보호할 수 있다.

[0067] 예시적 구현에서, 트랜잭션 트래픽 스펙은 NoC 시뮬레이션을 위해 별도로 생성될 수 있다. 여기서, 트랜잭션 트래픽 스펙의 설명은 NoC 시뮬레이션을 위하여 필요한만큼 매우 구체적일 수 있다.

[0068] 추적 생성

[0069] NoC 시뮬레이션의 예시적 구현에서, 외부 트랜잭션 추적을 이용하여 추가적인 유연성(flexibility)을 제공할 수 있다. 이러한 트랜잭션 추적은 통계 모델에서 계산되거나 실제 시스템에서 생성될 수 있다. 구성 요소에 대한 트랜잭션 추적은 트랜잭션 시작 메시지 목록 및 트랜잭션 간 간격을 포함할 수 있다. 예를 들어, 예시적 구현에서 트랜잭션 추적은 시뮬레이션을 위한 체인만 나타낼 수 있으며, 트랜잭션의 전체 세부 사항은 나타낼 수

없다.

[0070] 예시적 구현에서, 트랜잭션 트래픽 스펙을 이용하여 트래픽 추적을 생성할 수 있다. 트랜잭션 트래픽 스펙의 각 트랜잭션에 대해 트랜잭션 속도가 특정되어, 트랜잭션 트래픽 스펙에 맞는 트랜잭션에 대한 이용 가능한 추적을 생성할 수 있다. 예시적 구현에서, n 사이클에 걸쳐있는 추적을 생성하려면, 레이트 r 인 트랜잭션은 추적에서 nr 메시지를 전송해야 한다. 트랜잭션 배열은 특정 소스의 각 트랜잭션 ti가 nri 번 표시되도록 구성할 수 있다. 이후, 배열을 무작위로 섞어서 메시지를 보낼 순서를 정할 수 있다. 다음으로, 추적이 n 사이클에 걸쳐 있도록 메시지 사이의 지연을 계산하는 것이 남아있다. 예를 들어, 한 주기당 하나의 비트만 보낼 수 있기 때문에, 총 박자 수 t가 n보다 크면 지정한 총 트래픽이 전송주기를 초과할 수 있다. 또 다른 구현에서, 표준 레이트 제한 알고리즘은 메시지들 간의 갭을 갖는 메시지 송신을 스케줄링하는데 이용될 수 있고, 짧은 간격에 걸친 평균 대역폭은 사이클 당 t/n 비트에 대한 송신 비트 레이트를 대략적으로 제한할 수 있다. 또 다른 구현에서, NoC는 모델에 대해 버스트(burstiness) 및 지터(jitter)와 같은 속성을 이용하여 보다 현실적인 추적을 만들 수 있다.

[0071] 트랜잭션 시뮬레이션

[0072] 예시적 구현에서, NoC는 트랜잭션 시뮬레이션을 위해 본 발명의 트랜잭션 트래픽 스펙을 이용할 수 있다. 본 발명의 트랜잭션 트래픽 스펙을 이용하는 트랜잭션 시뮬레이션은 보다 정확하고 상세한 시뮬레이션을 제공하는 플로우 기반 시뮬레이션과 상이하다. 플로우 레벨 스펙을 이용하는 플로우 기반 시뮬레이션과 트랜잭션 트래픽 스펙을 이용하는 트랜잭션 시뮬레이션은 메시지가 트랜잭션 중간 지점(midpoint)에 도달할 때 어떤 일이 발생하는지에 따라 다를 수 있다. 두 시뮬레이션 모두에서 보낸 메시지의 추적은 네트워크를 활성화하는데 이용될 수 있다. 트랜잭션 기반 시뮬레이션의 경우, 추적은 메시지 수신시 다른 메시지가 트리거되므로, 각 트랜잭션의 첫 번째 메시지만 스케줄(schedule) 하면 된다. 패킷이 목적지에 도달하면, 패킷에는 체인의 다음 패킷에 대한 포인터가 있을 수 있다. 다음 패킷은 패킷을 보낼 네트워크 호스트/홉의 전송 큐(queue)에 삽입될 수 있다. 여기서, 응답의 전송은 해당 호스트/홉에 의한 처리를 설명하기 위해 지연될 수 있다. 만약, 큐에 공간이 없으면, 호스트는 메시지를 수신할 수 없으므로 배압(backpressure)이 업스트림(upstream) 될 수 있다.

[0073] 트랜잭션 기반 시뮬레이션의 예시적 구현은, 플로우 레벨 스펙을 이용하는 점대점 시뮬레이션을 통한 트랜잭션 트래픽 스펙의 이용을 통한 이점이 있을 수 있다. 예를 들어, 트랜잭션 시뮬레이션은 레지스터 전송 레벨(RTL) 시뮬레이션 중에 발생할 수 있는 모든 잠재적인 교착 상태를 재현할 수 있다. 잠재적 교착 상태를 미리 감지하면, NoC에서 교착 상태가 없는 트래픽을 보장하고 NoC가 라우팅 기능 및 VC 할당 수준까지 적절하게 구성되도록 할 수 있다. VC와 구성 요소 간의 종속성을 정확하게 모델링 할수록, 시뮬레이션을 통해 실제 시스템에서 발생할 수 있는 잠재적인 교착 상태를 감지하는 것이 보다 더 중요해질 수 있다. NoC 설계 및 시뮬레이션을 동작하는 동안, VC 교착 알고리즘이 시스템의 정확한 의존성 모델을 이용하지 않을 수 있음을 나타내는 많은 잠재적 교착 상태(potential deadlocks)가 발생할 수 있다.

[0074] 또한, 트랜잭션 트래픽 스펙을 이용함으로써, 시뮬레이션은 NoC의 보다 현실적인 결과를 제공할 수 있다. 트랜잭션 트래픽 스펙을 이용하는 시뮬레이터는 요청 후 응답을 생성하므로, 시뮬레이션 중 감지된 트래픽 패턴은 실제 시스템을 보다 정확하게 나타낼 수 있다. 또한, 네트워크의 동작에 대한 보다 정확한 결과가 제공되어, 대역폭의 정확한 프로비저닝(provisioning)을 설정할 수 있다. 이러한 이점들은 시뮬레이터의 유용성을 크게 향상시키지만, 시뮬레이터를 더 복잡하게 할 수 있다.

[0075] NoC 설계

[0076] 또한, 보다 정확한 성능 시뮬레이션 외에도, 예시적 구현에서, 트랜잭션 트래픽 스펙을 활용하여 NoC를 설계할 수 있다. 트랜잭션 트래픽 스펙을 이용하여 NoC 설계는 견고하고 신뢰할 수 있는 결과를 얻을 수 있으며, 필요한 경우 신속하게 설계된 반복 작업을 수행할 수 있다. 트랜잭션 트래픽 스펙에는 상호 의존성 정보, 대역폭 정보, 대기 시간 정보가 포함되어 있으므로, 트랜잭션 트래픽 스펙은 경로 할당 및 기타 NoC 연결 설정에 이용될 수 있다. 또한, 트랜잭션 트래픽 스펙은 교착 상태가 없는 NoC를 설계하는데 도움이 되는 VC 할당이 가능하도록 이용될 수 있다. 예시적 구현에서, 트랜잭션 트래픽 스펙은 NoC의 정확한 코어 배치를 위해 이용될 수 있다.

[0077] VC 매핑

[0078] VC 할당 중 교착 상태를 피하기 위해, 트래픽 플로우에 NoC에 경로와 VC를 할당할 수 있다. 트랜잭션 정보는 최적의 VC 할당에 이용될 수 있다. 트랜잭션 정보는 트랜잭션의 다양한 플로우 간에 추가 의존성 정보를 수신하고 VC 할당이 교착 상태가 없는지 확인하는데 이용될 수 있다.

[0079] 플로우 기반 트래픽 그래프를 사용하는 시뮬레이션은 NoC에서 교착 상태를 감지하고 방지하는데 충분하지 않을 수 있지만, 트랜잭션 트래픽 스펙은 더 나은 시뮬레이션 및 교착 상태 감지 기능을 제공할 수 있다. 플로우 기반 모델은 각 메시지를 서로 독립적으로 제어하지만, 실제 시스템에서는 리퀘스트(re-quest)/응답 프로토콜이 이용될 수 있다. 응답은 요청에 따라 달라지며 응답 및 요청 메시지는 트랜잭션 트래픽 스펙에 설명된 것과 같이, 단일 트랜잭션에 속한다. 예를 들어, 캐시 누락시, CPU 페이싱(facing) 인터페이스에서 로드 요청을 수신하고 메모리 페이싱 인터페이스에서 로드 요청을 전송하는 캐시 컨트롤러가 있을 수 있다. 제한된 버퍼링으로 인해 응답을 보낼 수 없는 경우, 구성 요소는 요청 수락을 중지해야 한다. 이러한 종속성은 상호 연결 교착 상태를 방지하기 위해 매핑에 포함될 수 있다. 이러한 상황을 제어하는 관련 기술은 요청 및 응답을 개별 VC에 넣는 것을 포함하지만, 이러한 설계는 복잡한 코히런스 프로토콜에 과도하게 VC를 사용하게 될 수도 있다.

[0080] 예시적 구현에서, 트랜잭션 트래픽 스펙은 동일한 트랜잭션의 플로우 간 의존성을 나타내는 NoC 구성 요소의 VC 할당에 이용될 수 있다. 예시적 구현에서, VC 할당을 위한 트랜잭션 트래픽 스펙의 이용은 멀티 홉 트래픽 트랜잭션으로부터 의존성을 추론함으로써 동일한 트랜잭션의 플로우들 사이의 의존성을 나타낼 수 있다. 예를 들어, CPU와 메모리 간의 트랜잭션에서 인커밍(incoming) 인터페이스에서 아웃고잉(outgoing) 인터페이스에 이르기까지 메모리의 의존성을 유추할 수 있다. VC 할당 중 이러한 종속성을 고려하면, 훨씬 적은 수의 VC를 사용하면서 교착 상태가 없는 네트워크를 설계할 수 있으므로 보다 효율적인 NoC 설계가 가능하다. 또한, 설계 결과는 복잡한 프로토콜의 경우에도 교착 상태가 발생하지 않고 정확한 구조로 설계될 수 있다.

[0081] 코어 배치

[0082] NoC의 통신 비용은 구성 요소 간 거리에 따라 증가하므로 구성 요소의 위치를 변경하면 NoC 성능에 큰 영향을 줄 수 있다. 위치를 지정하여 대기 시간과 복잡도를 줄일 수 있으나, 일부 구성 요소를 가까이 이동시키면 다른 구성 요소 간의 거리가 늘어날 수 있기 때문에 정확한 위치 배치가 필요하다. 시뮬레이션 된 어닐링부터 분기 한정(branch-and-bound) 기술에 이르기까지 NoC의 NP 배치 문제에 대한 다양한 접근 방식이 있다. 이하 설명된 예시적 구현은 트래픽이 대기 시간 타겟을 충족시키도록 할 수 있다.

[0083] 플로우 기반 트래픽 모델은 구성 요소 쌍에 대기 시간 제약 조건이 있을 수 있다. 예를 들어, CPU와 캐시 컨트롤러 간의 대기 시간은 5 미만으로 제한될 수 있다. 어플리케이션에서 캐시 컨트롤러에 대한 대기 시간은 메모리 요청과 응답 사이의 지연 중 일부일 수 있다. 따라서, 어플리케이션에서 최적화될 수 있는 전체 트랜잭션 대기 시간을 알 수 있다.

[0084] 예시적 구현에서, 홉당 속성 대신 트랜잭션 속성으로 대기 시간을 지정하면 배치 최적화 알고리즘의 제약을 완화시킬 수 있다. 트랜잭션 트래픽 스펙에 나타나는 트랜잭션 속성을 이용하면, 스펙이 시스템의 모든 제약 조건을 충족하는 솔루션을 쉽게 찾을 수 있도록 해당 트랜잭션의 모든 홉 사이에 대기 시간을 할당할 수 있다.

[0085] 예시적인 구현 예에서, NoC는 전체 시스템 레벨 연결성, 의존성 정보, 메시지/패킷/플릿 및/또는 채널의 대역폭 정보, 메시지 트랜잭션의 대기 시간 정보, 메시지의 크기와 같은 메시지 속성 정보, 우선순위, 스펙에서 트래픽의 다른 설명들의 순서 등을 갖는 트랜잭션 트래픽 스펙으로부터 생성될 수 있다.

[0086] 일 실시 예에 따르면, 본 발명의 제안된 스펙은 임의의 유형의 경로 및/또는 홉 시퀀스를 처리하도록 구성될 수 있으며, 제1 예에서, 소스 호스트 및 목적지 호스트는 하나 이상의 중간 홉과 동일할 수 있다. 또한, 목적지가 소스에 직접 연결되고, 소스가 첫 번째 중개자(intermediary)에 연결되고, 첫 번째 중개자가 주기에서 두 번째 중개자로 이동하여 목적지로 연결되는 연결성을 가질 수 있다. 종래의 스펙은 쌍방향 연결만 지원하는 반면, 본 발명의 스펙은 연결성, 의존성, 대역폭, 대기 시간 및 메시지 특성 중 하나 또는 조합을 포착(capture)할 수 있도록 구성할 수 있다.

[0087] 구현시, 메시지의 순서를 분석하거나 스펙 또는 비의존성을 분석하여, 의존성을 결정할 수 있다. 또한, 대역폭 및 대기 시간을 특정할 수 있도록 스펙을 구성할 수 있다. 여기서, 대기 시간은 메시지 대기 시간, 트랜잭션 대기 시간 등이 될 수 있다. 따라서, 대기 시간은 각 홉 또는 전체 트랜잭션(복수 홉 포함)일 수 있다. 메시지 속성은 메시지의 크기, 우선순위, 메시지의 서비스 품질 및 메시지에 대한 기타 정의된 정보를 포함할 수 있다. 대역폭 분포는 시간에 따라 변화하는 대역폭 분포를 포함할 수 있고, 대역폭 분포는 테이블에 저장되거나 시간의 함수로 표현될 수 있다. 일 예로, 일정 시간 내에 특정 대역폭을 제공할 수 있는 등시성 (isochronous) 트래픽으로부터 대역폭 분포를 포착(capture)할 수 있다.

[0088] 도 7은 예시적 구현이 수행될 수 있는 예시적인 NoC 설계 및 구현 시스템 (700)을 도시한다. 컴퓨터 시스템 (700)은 입출력부(735), 저장 매체(760), 및 당업자에게 공지된 하나 이상의 유닛을 실행하도록 동작 가능한 프

로세서(710)를 포함하는 서버(705)를 포함한다. 본 명세서에서 사용된 "컴퓨터 판독 가능 매체"라는 용어는 실행을 위해 프로세서(710)에 명령을 제공하는 임의의 매체를 지칭하고, 컴퓨터 판독 가능 저장 매체의 형태, 판독 전용 메모리, 랜덤 액세스 메모리, 솔리드 스테이트 장치(solid state devices) 및 드라이브, 또는 전자 정보를 저장하기에 적합한 임의의 다른 유형의 매체를 포함하며, 다만, 이에 한정되는 것은 아니다. 입출력부(735)는 키보드, 마우스, 터치 장치 또는 음성 인식 장치와 같은 입력 장치를 이용할 수 있는 사용자 인터페이스(740) 및 조작자 인터페이스(745)로부터의 입력을 처리한다.

[0089] 또한, 서버(705)는 휴대용 하드 드라이브, 광 매체(CD 또는 DVD), 디스크 매체 또는 컴퓨터가 실행 가능한 코드를 판독할 수 있는 임의의 다른 매체와 같은 착탈식 저장 장치를 포함할 수 있는 외부 저장 매체(750)에 연결될 수 있다. 또한, 서버(705)는 디스플레이와 같은 출력 장치(755)에 접속되어 데이터 및 다른 정보를 사용자에게 출력할 수 있을 뿐만 아니라 사용자로부터 추가 정보를 요청할 수 있다. 서버(705)로부터 사용자 인터페이스(740), 조작자 인터페이스(745), 외부 저장 매체(750) 및 출력 장치(755)로의 연결은 802.11 표준, 블루투스(Bluetooth®) 또는 셀룰러 프로토콜(cellular protocols)과 같은 무선 프로토콜 또는 케이블 또는 광섬유와 같은 물리적 전송 매체를 통해 이루어질 수 있다. 또한, 출력 장치(755)는 사용자와 상호 작용하기 위한 입력 장치로서도 동작할 수 있다.

[0090] 프로세서(710)는 트랜잭션 트래픽 스펙 모듈(711), 상호 의존성 결정 모듈(712), 대역폭 결정 모듈(713), 응답 우선순위 결정 모듈(714), 트래픽 생성 모듈(715) 및 시뮬레이션 모듈(716)을 포함하는 하나 이상의 모듈을 실행할 수 있으며, 트랜잭션 트래픽 스펙 모듈(711)은 하나 이상의 구성 요소 사이의 상호 의존성, 대역폭 할당, 메시지 속성 및 다른 트랜잭션 트래픽 레벨 파라미터/요소/속성 중 하나 또는 조합에 관한 정보를 포함하는 NoC 트랜잭션으로 구성될 수 있다.

[0091] 일 실시 예에서, 상호 의존성 결정 모듈(712)은 트랜잭션 트래픽 스펙 모듈(711)로부터 의존성 정보를 추출하여 하나 이상의 메시지/패킷/플릿 간의 상호 의존성, 요청 호스트와 목적지 호스트 사이의 상호 의존성, 중간 홉들의 의존성, 트랜잭션 트래픽 스펙 모듈(711)의 다른 의존성을 결정할 수 있다. 예시적 구현에서, 트랜잭션 트래픽 스펙 모듈(711)은 의존성 정보, 대역폭 정보 및 속성 정보와 같은 하나 이상의 파라미터를 이용하여 트랜잭션 트래픽 스펙을 생성하도록 구성될 수 있다.

[0092] 다른 예시적 구현에서, NoC 설계 및 시뮬레이션 시스템(700)은 NoC의 호스트와 중간 홉 사이의 가용 대역폭을 결정하고, 이에 따라 트랜잭션 트래픽 스펙에 기초하여 트래픽을 설계하도록 구성된 대역폭 분배 모듈(713)을 포함할 수 있다. 전술한 바와 같이, 트랜잭션 트래픽 스펙은 메시지 크기 우선순위 및 트랜잭션의 모든 메시지의 순서에 관한 정보를 포함할 수 있다.

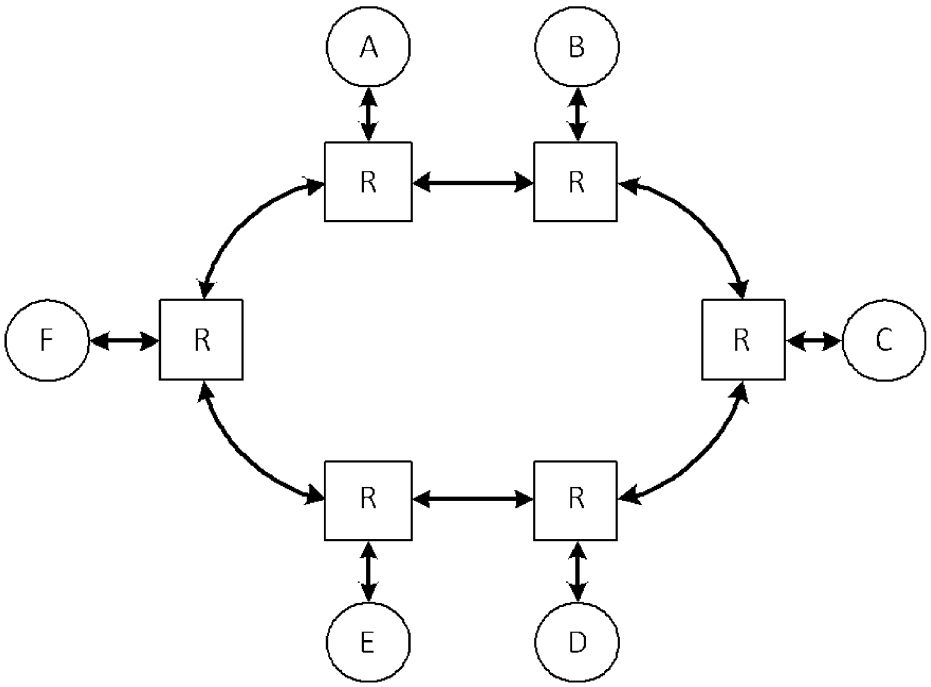
[0093] 다른 실시 예에서, 우선순위 결정 모듈(714)은 정보를 추출하여 호스트 및/또는 중간 홉으로부터 들어오거나 나가는 메시지를 분석하고 우선순위를 결정하도록 구성될 수 있다. 트랜잭션 구현 트래픽 규격을 이용하는 예시적 구현에서, 트래픽 생성 모듈(715)은 다수의 의존성, 우선순위, 대역폭, 대기 시간 및 다른 관련 정보를 고려하여 이에 따라 트래픽을 처리/제어하도록 구성될 수 있다. 다른 예시적 구현에서, 시뮬레이션 모듈(716)은 트랜잭션 트래픽 스펙을 이용하여 전체 NoC를 시뮬레이션하여 NoC를 분석하고 NoC에서 잠재적인 교착 상태를 결정할 수 있다.

[0094] 또한, 본 발명의 상세한 설명의 일부는 컴퓨터 내의 연산의 알고리즘 및 기호 표현으로 나타날 수 있다. 이러한 알고리즘 및 기호 표현은 데이터 처리 분야의 기술자가 기술의 본질을 당업자에게 가장 효과적으로 전달하기 위해 사용하는 수단이다. 알고리즘은 원하는 최종 상태 또는 결과를 유도하는 일련의 정의된 단계일 수 있다. 예시적 구현에서 수행된 단계는 유형의 결과를 얻기 위하여 실제 조작을 필요로 할 수 있다.

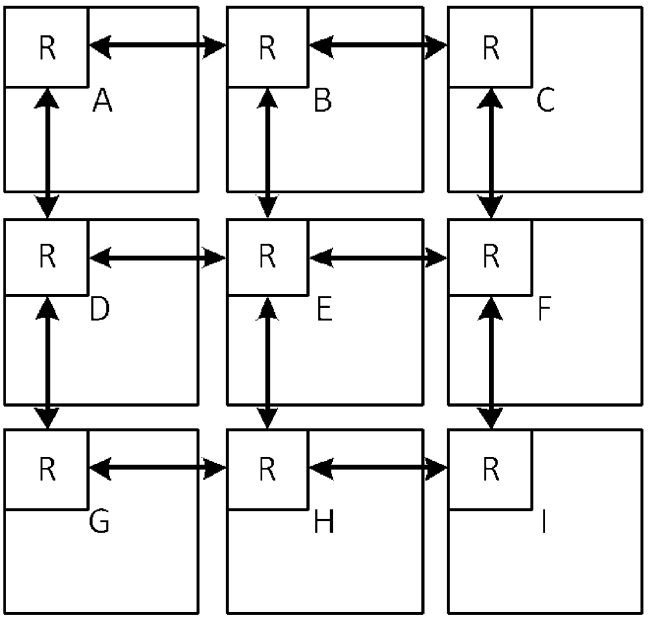
[0095] 또한, 본 명세서의 다른 구현 예는 본 명세서의 기재 및 본 명세서에 개시된 예시적 구현의 실시로부터 당업자에게 명백할 것이다. 기술된 구현 예의 다양한 양태들(aspects) 및/또는 컴포넌트들(components)은 단독으로 또는 임의의 조합으로 사용될 수 있다. 본 명세서 및 실시 예는 예시로서 고려되며, 이하의 특허 청구 범위에 의해 그 응용의 범위 및 사상이 특정될 수 있다.

도면

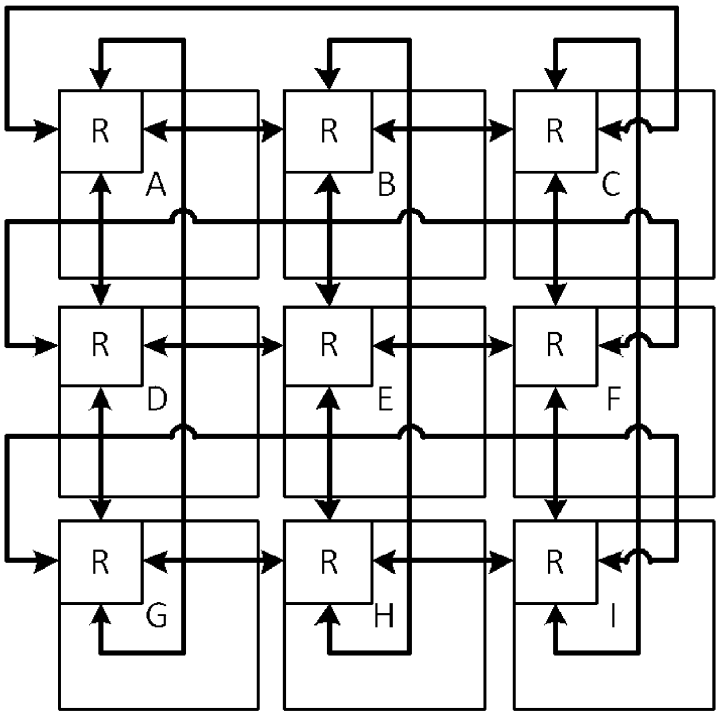
도면1a



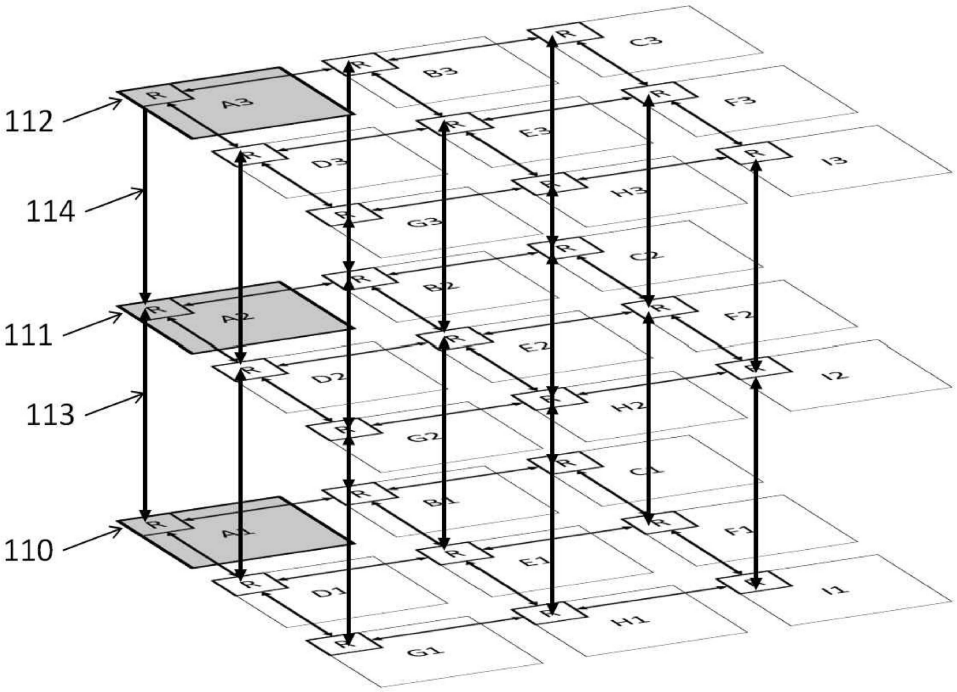
도면1b



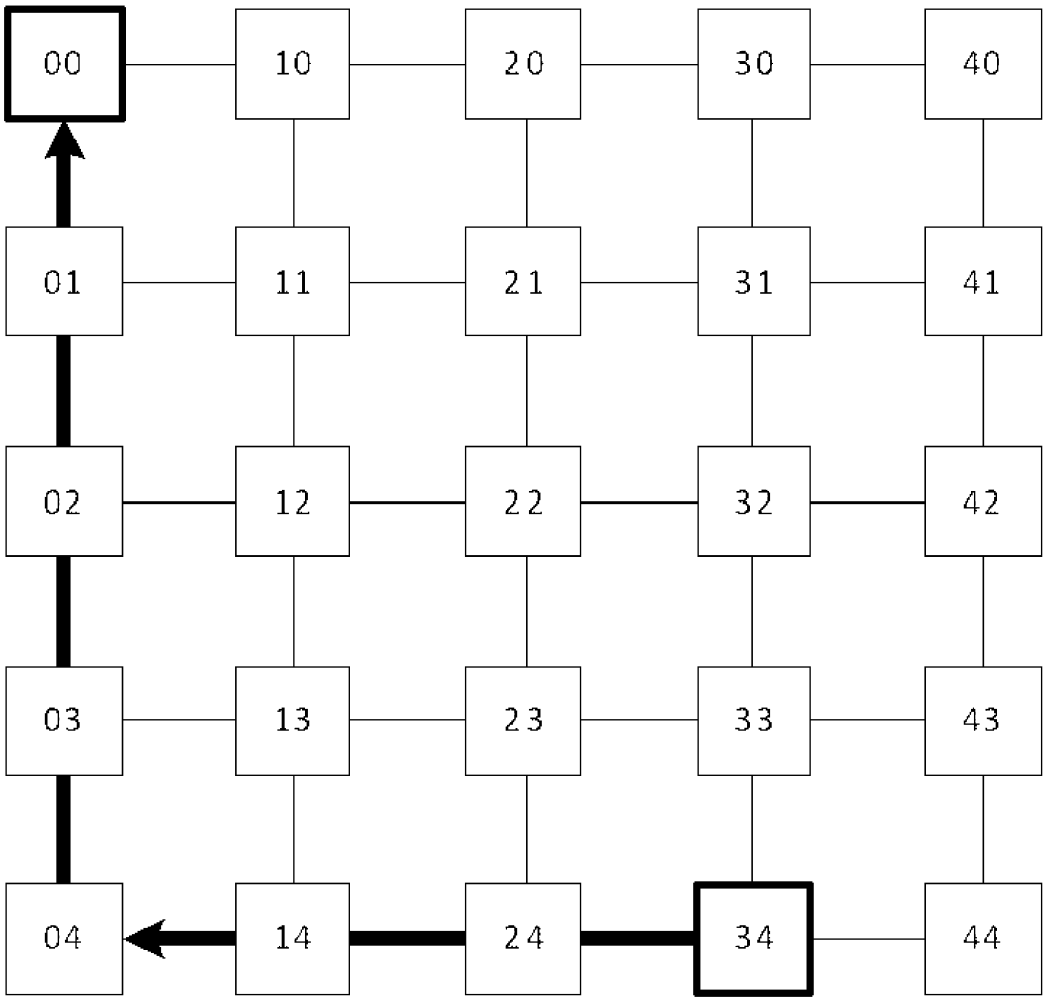
도면1c



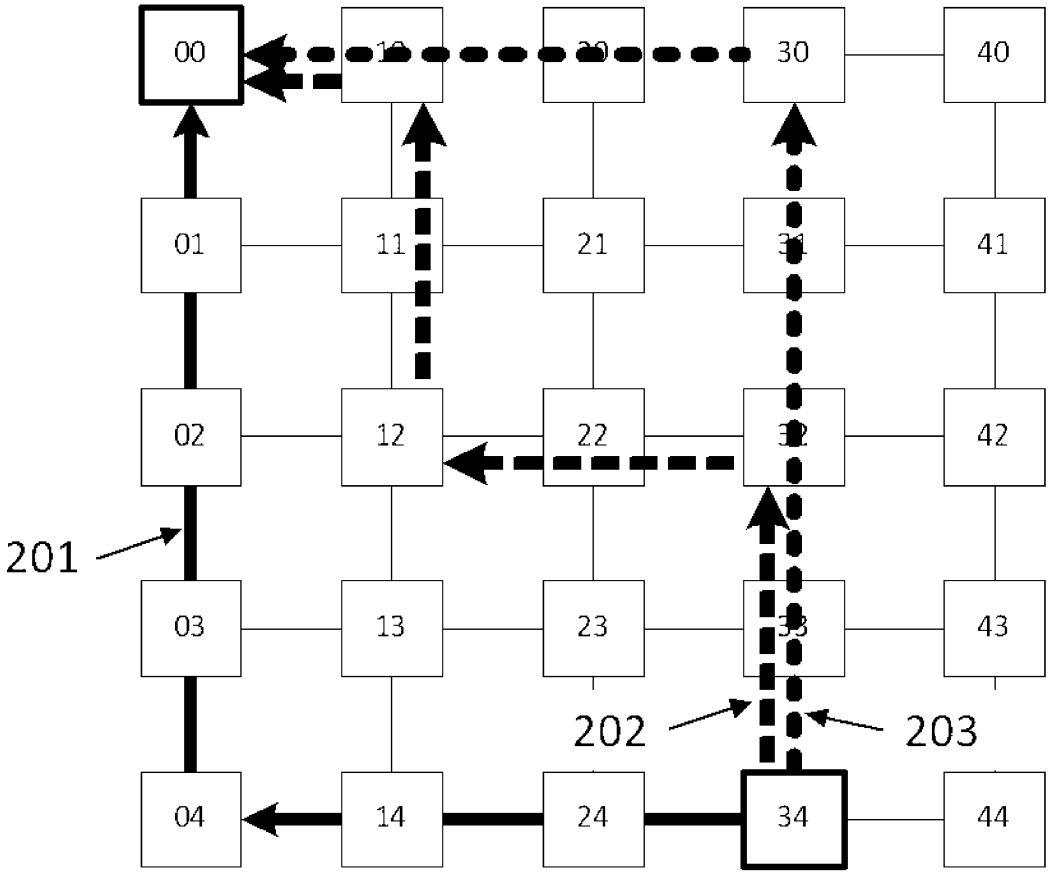
도면1d



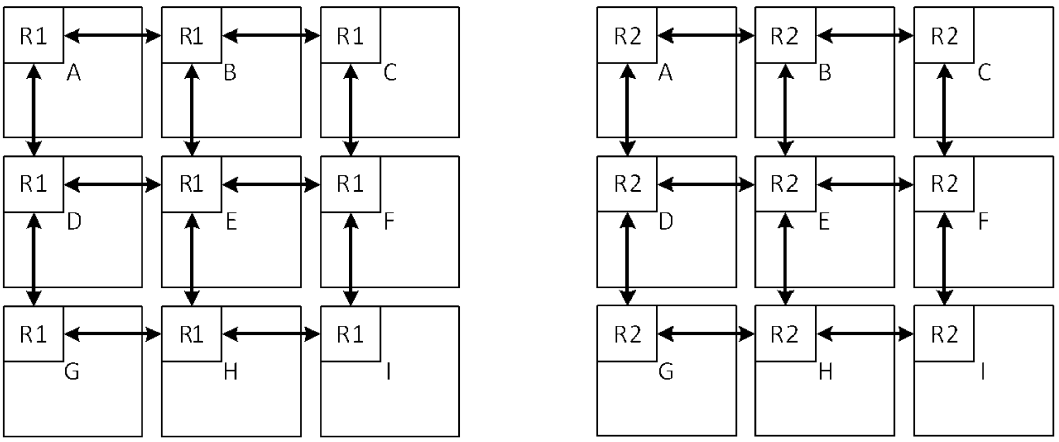
도면2a


00
10
20
30
40
01
11
21
31
41
02
12
22
32
42
03
13
23
33
43
04
14
24
34
44

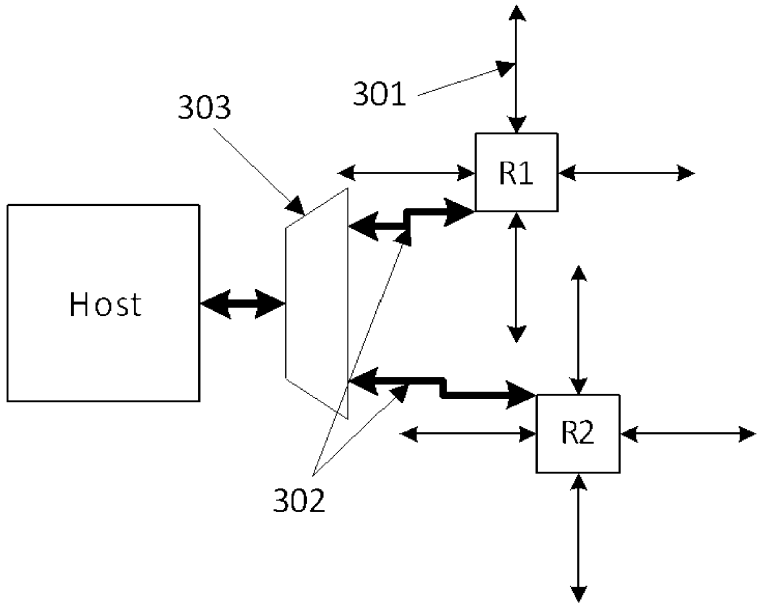
도면2b


00
30
40
01
11
21
41
02
12
22
42
201
03
13
23
43
202
203
04
14
24
34
44

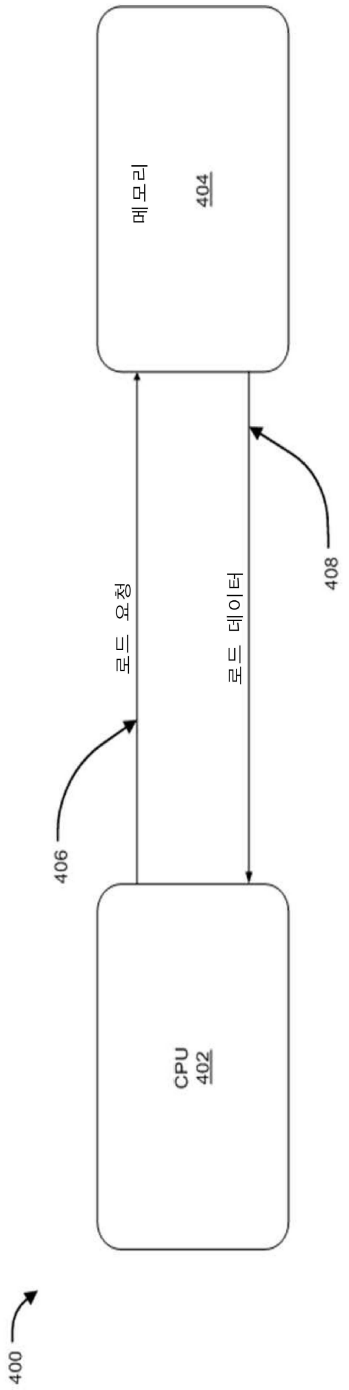
도면3a


R1
A
R1
B
R1
C
R1
D
R1
E
R1
F
R1
G
R1
H
R1
I
R2
A
R2
B
R2
C
R2
D
R2
E
R2
F
R2
G
R2
H
R2
I

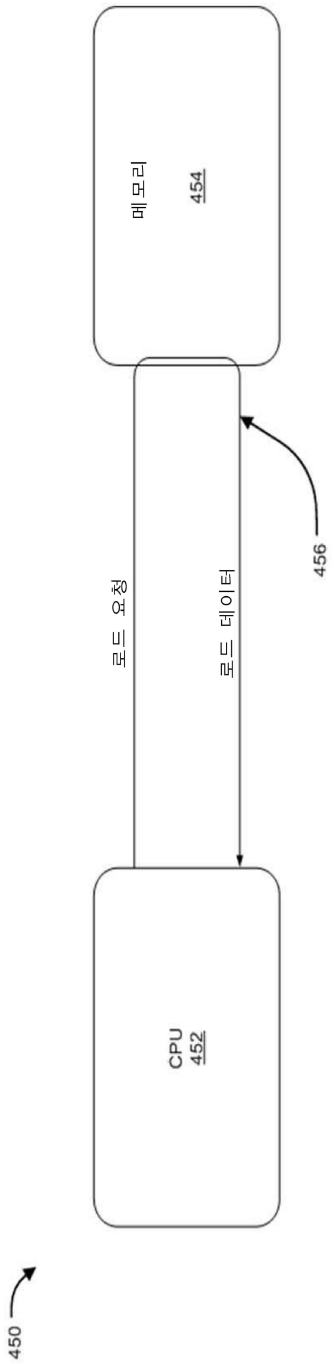
도면3b



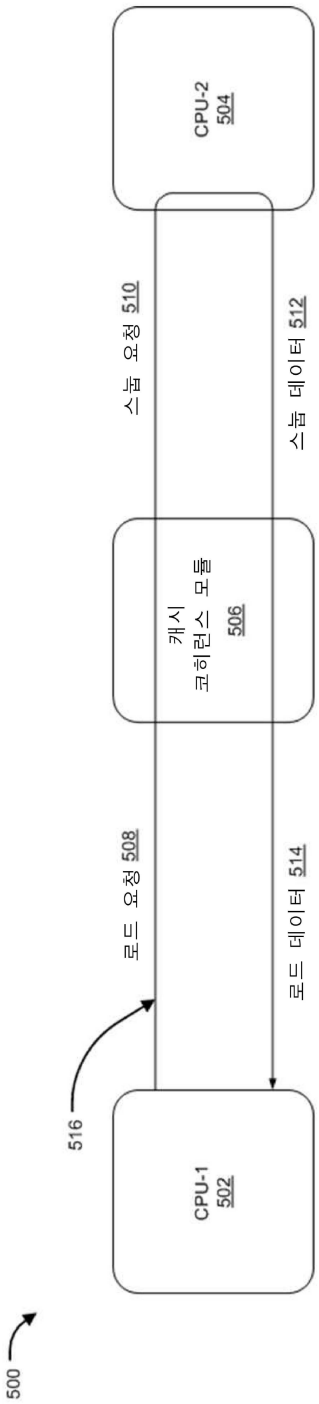
도면4a



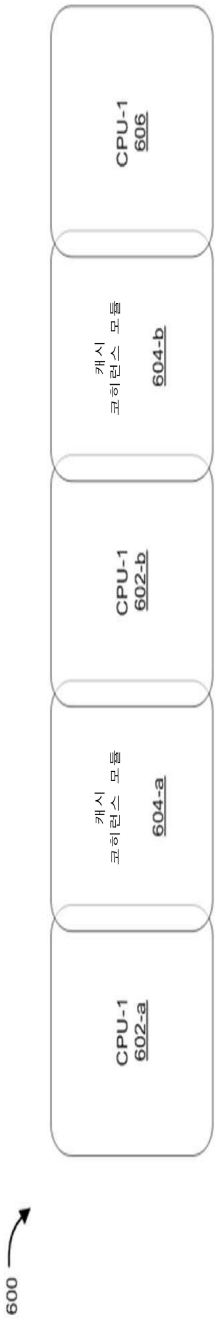
도면4b



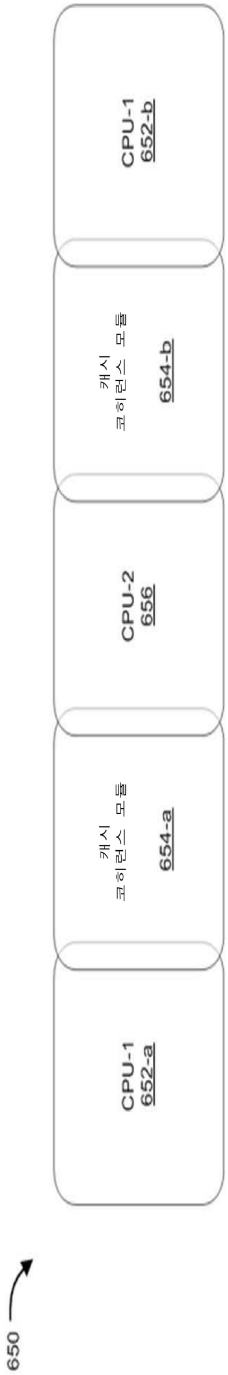
도면5


CPU-2
504
스눕 요청 510
스눕 데이터 512
캐시
코히런스 모듈
506
로드 요청 508
로드 데이터 514
516
CPU-1
502
500

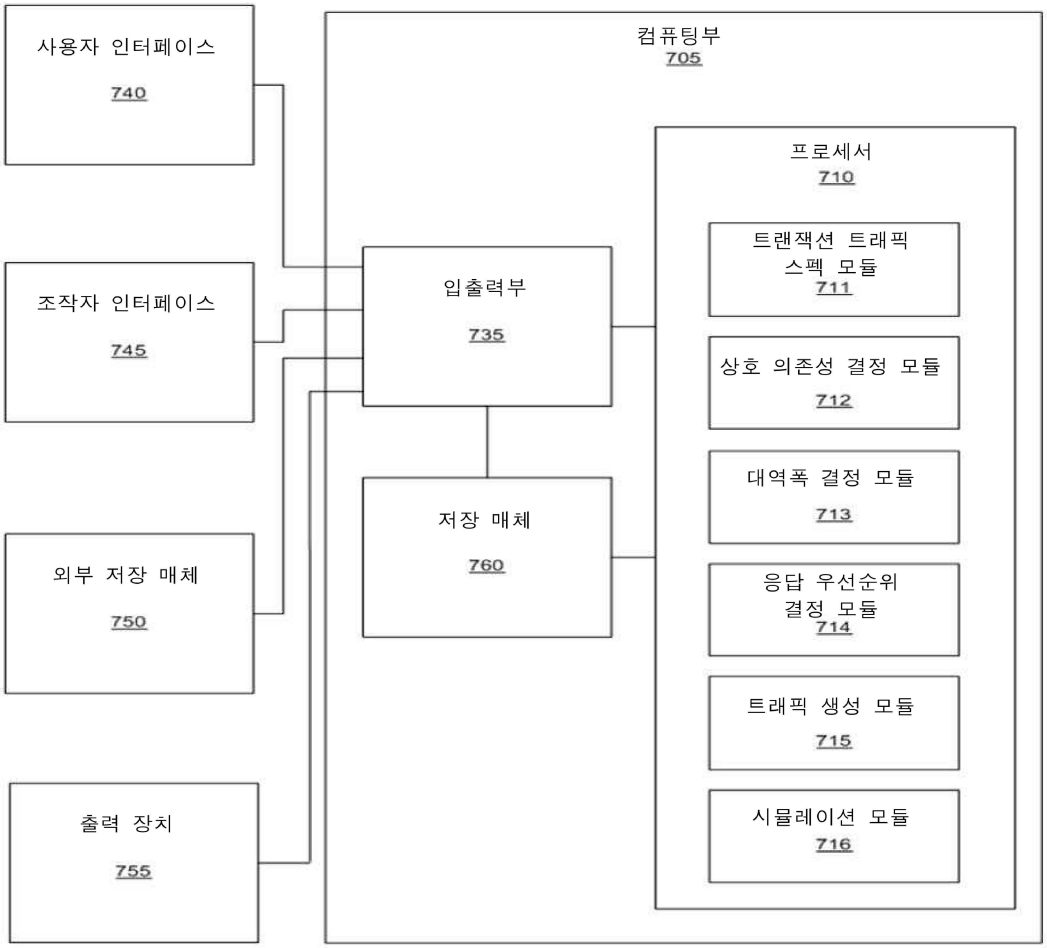
도면6a


CPU-1
606
캐시
코히런스 모듈
604-b
CPU-1
602-b
캐시
코히런스 모듈
604-a
CPU-1
602-a
600

도면6b


650
CPU-1
652-a
캐시
코히런스 모듈
654-a
CPU-2
656
캐시
코히런스 모듈
654-b
CPU-1
652-b

도면7


사용자 인터페이스
740
조작자 인터페이스
745
외부 저장 매체
750
출력 장치
755
컴퓨팅부
705
입출력부
735
저장 매체
760
프로세서
710
트랜잭션 트래픽 스펙 모듈
711
상호 의존성 결정 모듈
712
대역폭 결정 모듈
713
응답 우선순위 결정 모듈
714
트래픽 생성 모듈
715
시뮬레이션 모듈
716