(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2017-0042600
(43) 공개일자 2017년04월19일

(51) 국제특허분류(Int. Cl.)
G06F 15/173 (2006.01) *H04L 12/24* (2006.01)
H04L 12/753 (2013.01) *H04L 29/08* (2006.01)
(52) CPC특허분류
G06F 15/17387 (2013.01)
H04L 41/0806 (2013.01)
(21) 출원번호 10-2017-7004368
(22) 출원일자(국제) 2015년08월11일
심사청구일자 없음
(85) 번역문제출일자 2017년02월16일
(86) 국제출원번호 PCT/US2015/044607
(87) 국제공개번호 WO 2016/028545
국제공개일자 2016년02월25일
(30) 우선권주장
14/461,614 2014년08월18일 미국(US)

(71) 출원인
어드밴스드 마이크로 디바이시즈, 인코포레이티드
미국 캘리포니아 94088-3453 서니베일 피.오.박스 3453 원 에이엠디 플레이스
(72) 발명자
제임스 마이클 이.
미국 캘리포니아 94040 마운틴 뷰 카멜리타 드라이브 195
프리커 진-필립
미국 캘리포니아 94040 마운틴 뷰 아이홀러 코드 1225
(74) 대리인
박장원

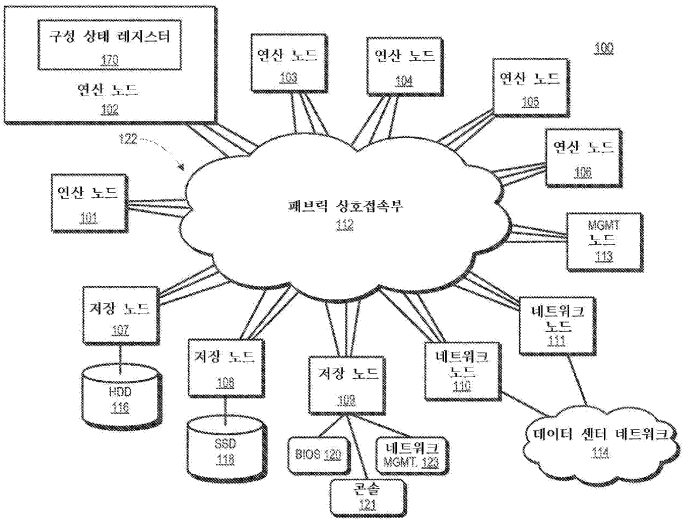
전체 청구항 수 : 총 14 항

(54) 발명의 명칭 **셀룰러 오토머턴을 이용한 클러스터 서버의 구성**

(57) 요 약

클러스터 컴퓨터 서버(100)는 시스템 리셋 또는 기타 구성 이벤트 후에 구성된다. 클러스터 컴퓨터 서버의 패브릭의 각 노드(101, 102)는 구성을 위해 셀룰러 오토머턴의 셀로서 채택되고, 이에 따라 특별 구성 네트워크가 중앙 관리 유닛으로부터의 구성 정보를 통신할 필요가 없다. 대신, 노드들은, 그 노드들에서의 소프트웨어 서비스들의 정상적인 실행 동안 메시지들을 통신하는 데 사용되는 동일한 패브릭 상호접속부(112)를 사용하여 구성 정보를 통신한다.

대 표 도 - 도1



(52) CPC특허분류
H04L 41/0886 (2013.01)
H04L 41/16 (2013.01)
H04L 45/48 (2013.01)
H04L 67/10 (2013.01)

명 세 서

청구범위

청구항 1

서버 시스템(100)으로서,

소프트웨어 서비스들의 실행 동안 메시지들을 경로설정하는 패브릭 상호접속부(112); 및

상기 서버 시스템을 위한 서비스들을 실행하도록 상기 패브릭 상호접속부에 연결된 복수의 연산 노드(101, 102)를 포함하되,

상기 복수의 연산 노드의 각각은 상기 패브릭 상호접속부를 이용하여 구성 정보(401, 402, 403, 404, 405)를 통신하도록 셀룰러 오토머턴의 셀로서 구성된, 서버 시스템.

청구항 2

제1항에 있어서, 상기 복수의 연산 노드의 각각은,

접속된 연산 노드들에서 대응하는 상태 필드들의 상태들(502, 503)의 천이에 응답하여 미리 정의된 상태들의 세트들 간에 상태 필드들의 세트의 천이를 실시하고,

상기 상태 필드들의 세트의 천이에 기초하여 경로설정 정보(610)를 수신하고, 및

상기 경로설정 정보에 기초하여, 수신된 메시지들(612)을 상기 패브릭 상호접속부를 통해 상기 복수의 연산 노드의 다른 연산 노드들에 경로설정하는, 서버 시스템.

청구항 3

제2항에 있어서, 상기 복수의 연산 노드의 각각은, 상기 상태 필드들(608)의 세트의 천이에 기초하여 스패닝 트리(spanning tree)에서의 자신의 접속된 연산 노드들 중 하나 이상에 대하여 자신의 위치를 식별하는, 서버 시스템.

청구항 4

제3항에 있어서, 상기 복수의 연산 노드의 각각은, 상기 스패닝 트리에 기초하여 경로설정 정보를, 상기 패브릭 상호접속부를 통해, 접속된 연산 노드에 통신하고, 상기 통신은 상기 상태 필드들(610)의 세트의 천이에 기초하는, 서버 시스템.

청구항 5

제3항에 있어서, 상기 복수의 연산 노드 중 적어도 하나는, 제1유형의 메시지 수신에 응답하여, 상기 상태 필드들(1001, 1002, 1003)의 세트의 천이에 기초하여 상기 제1유형의 메시지를 상기 스패닝 트리의 상기 적어도 하나의 연산 노드의 위치에 대하여 상기 스패닝 트리의 복수의 원위 노드 중 선택된 제1원위 노드에 통신하는, 서버 시스템.

청구항 6

제5항에 있어서, 상기 복수의 연산 노드 중 상기 적어도 하나는, 상기 복수의 원위 노드 중 상기 선택된 제1원위 노드로부터 상기 제1유형의 메시지를 후속 수신하는 것에 응답하여, 상기 상태 필드들(1004, 1005)의 세트의 천이에 기초하여 상기 제1유형의 메시지를 상기 복수의 원위 노드 중 선택된 제2원위 노드에 통신하는, 서버 시스템.

청구항 7

제6항에 있어서, 상기 복수의 연산 노드 중 상기 적어도 하나는, 상기 복수의 원위 노드 중 상기 선택된 제2원위 노드로부터 상기 제1유형의 메시지를 후속 수신하는 것에 응답하여, 상기 상태 필드들(1007)의 세트의 천이에 기초하여 상기 제1유형의 메시지를 상기 스패닝 트리의 상기 적어도 하나의 연산 노드 중 근위 노드에 통신

하는, 서버 시스템.

청구항 8

제5항에 있어서, 상기 복수의 연산 노드 중 상기 적어도 하나는, 상기 제1유형의 메시지의 페이로드가 상기 복수의 연산 노드 중 상기 적어도 하나를 대상으로 한다는 것을 식별함에 응답하여, 상기 상태 필드들의 세트의 천이에 기초하여 상기 복수의 연산 노드 중 상기 적어도 하나의 구성 레지스터에 상기 메시지의 페이로드를 저장하는, 서버 시스템.

청구항 9

제5항에 있어서, 상기 복수의 연산 노드 중 상기 적어도 하나는, 제2유형의 메시지의 수신에 응답하여, 상기 상태 필드들(1101, 1102)의 세트의 천이에 기초하여 상기 스패닝 트리의 상기 적어도 하나의 연산 노드의 위치에 대하여 상기 스패닝 트리의 상기 복수의 원위 노드의 각각에 상기 경로설정 정보를 동시에 통신하는, 서버 시스템.

청구항 10

제2항에 있어서, 상기 경로설정 정보는, 상기 패브릭 상호접속부의 상기 복수의 연산 노드 중 하나의 위치를 식별하는 노드 어드레스 정보인, 서버 시스템.

청구항 11

제1항에 있어서, 상기 복수의 연산 노드 중 제1연산 노드는, 상기 제1연산 노드의 접속된 연산 노드들의 상태들(501, 502)의 천이에 기초하여 접속되지 않은 제2연산 노드의 구성을 식별할 수 있는, 서버 시스템.

청구항 12

방법으로서,

메시지를 경로설정하도록 패브릭 상호접속부에 연결된 복수의 연산 노드를 포함하는 서버에서의 시스템 리셋에 응답하여, 상기 복수의 연산 노드 중 제1연산 노드(101)에서,

접속된 연산 노드들에서의 대응하는 상태 필드들의 상태들의 천이에 응답하여, 미리 정의된 상태들(501, 502)의 세트들 간에 상태 필드들의 세트의 천이를 실시하는 단계;

상기 상태 필드들(610)의 세트의 천이에 기초하여 경로설정 정보를 수신하는 단계; 및

상기 경로설정 정보(612)에 기초하여 상기 패브릭 상호접속부를 통해 상기 복수의 연산 노드들 중 다른 연산 노드들에 수신된 메시지들을 경로설정하는 단계를 포함하는, 방법.

청구항 13

제12항에 있어서, 상기 제1연산 노드에서, 상기 상태 필드들(608)의 세트의 천이에 기초하여 스패닝 트리에서의 자신의 접속된 연산 노드들 중 하나 이상에 대한 상기 제1연산 노드의 위치를 식별하는 단계를 더 포함하는, 방법.

청구항 14

제13항에 있어서, 상기 스패닝 트리에 기초하여 상기 패브릭 상호접속부를 통해 상기 제1연산 노드로부터 접속된 연산 노드로 경로설정 정보를 통신하는 단계를 더 포함하고, 상기 통신은 상기 상태 필드들(610)의 세트의 천이에 기초하는, 방법.

발명의 설명

기 술 분 야

[0001] 본 개시 내용은, 일반적으로 처리 시스템에 관한 것으로서, 더욱 구체적으로는 클러스터 서버의 구성에 관한 것이다.

배 경 기 술

[0002] 서버 시스템 등의 고성능 연산 시스템은 때때로 하나 이상의 패브릭 상호접속부에 의해 함께 접속된 연산 노드들을 사용하여 구현된다. 연산 노드들은, 파일 관리, 데이터베이스 관리, 문서 인쇄 관리, 웹 페이지 저장 및 표현, 컴퓨터 게임 서비스 등, 또는 이들의 조합의 지정된 서비스들을 수행하기 위해 소프트웨어 프로그램들을 실행한다. 다수의 연산 노드는, 상대적으로 대량의 데이터 처리를 용이하게 함과 동시에 연산 시스템의 직관적인 빌드업과 스케일링도 용이하게 한다. 패브릭 상호접속부는, 연산 노드들 간의 통신을 위한 백본을 제공하므로, 프로세서 성능에 상당한 영향을 끼칠 수 있다. 시스템 리셋 후에 패브릭 상호접속부를 사용하려면, 연산 노드들이 통상적으로 패브릭 상호접속부의 통신 체계를 구현하는 어드레스 및 경로설정(routing) 테이블로 구성되어야 한다.

도면의 간단한 설명

[0003] 본 개시 내용은, 첨부된 도면을 참조함으로써, 더욱 잘 이해될 수도 있고, 많은 특징들과 장점들이 통상의 기술자에게 명백해질 수도 있다. 상이한 도들에서 동일한 참조 부호를 사용하는 것은 유사하거나 동일한 항목들을 나타낸다.

도 1은 일부 실시예들에 따른 클러스터 연산 서버의 블록도이다.

도 2는 일부 실시예들에 따라 클러스터 연산 서버를 위해 구현된 네트워크 토폴로지의 일례를 도시하는 블록도이다.

도 3은 일부 실시예들에 따라 도 2의 네트워크 토폴로지에 있어서 연산 노드와 이에 접속된 연산 노드들에 대한 차원 관계를 도시하는 블록도이다.

도 4는 일부 실시예들에 따라 클러스터 연산 서버의 연산 노드에 대한 구성 상태 필드들을 도시하는 블록도이다.

도 5는 일부 실시예들에 따라 도 4의 구성 상태 필드들 중 하나에 대한 구성 상태들의 세트를 도시하는 블록도이다.

도 6은 일부 실시예들에 따라 클러스터 연산 서버의 패브릭을 구성하는 방법을 도시하는 흐름도이다.

도 7은 일부 실시예들에 따라 관리 유닛 노드를 포함하는 클러스터 연산 서버에 대한 패브릭 토폴로지의 일례를 도시하는 블록도이다.

도 8은 일부 실시예들에 따라 구성 동안 도 4의 노드들의 상태 천이들을 도시하는 블록도이다.

도 9는 일부 실시예들에 따라 클러스터 연산 서버의 노드들에 의해 자기 조직된 스패닝 트리(spanning tree)의 일례를 도시하는 블록도이다.

도 10은 일부 실시예들에 따라 도 8의 스패닝 트리에 걸쳐 CHAIN형 구성 메시지를 통신하는 일례를 도시하는 블록도이다.

도 11은 일부 실시예들에 따라 도 8의 스패닝 트리에 걸쳐 CHAIN형 구성 메시지를 통신하는 일례를 도시하는 블록도이다.

도 12는 일부 실시예들에 따라 클러스터 연산 서버의 노드들의 물리적 배열의 일례를 도시하는 블록도이다.

도 13은 일부 실시예들에 따라 클러스터 연산 서버의 연산 노드의 구현예를 도시하는 블록도이다.

도 14는 일부 실시예들에 따라 집적 회로(IC) 디바이스를 설계 및 제조하는 방법을 도시하는 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0004] 도 1 내지 도 14는 시스템 리셋 또는 기타 구성 이벤트 후에 클러스터 연산 서버의 패브릭을 구성하는 기술들을 도시한다. 패브릭의 각 노드는, 구성을 위해 셀룰러 오토머턴(automaton)의 셀로서 사용되므로, 특별 구성 네트워크가 중앙 관리 유닛으로부터 구성 정보를 통신할 필요가 없다. 대신에, 노드들은, 패브릭 상호접속부를 고정 토폴로지에 따라 네트워크 경로설정을 위해 미리 구성할 필요 없이, 그 노드들에서의 소프트웨어 서비스들의 정상 실행 동안 메시지들을 통신하는 데 사용되는 동일한 패브릭 상호접속부를 사용하여 구성 정보를 통신한다.

이는 서버 복잡성을 감소시키고 클러스터 연산 서버의 더욱 양호한 확장성을 제공한다.

[0005] 예를 들어, 클러스터 연산 서버는 패브릭 상호접속부를 사용하여 자신의 다양한 노드들에 접속한다. 정상적인 동작 동안, 연산 노드들이 소프트웨어 서비스들을 실행하면, 각 노드가 유니캐스트 메시지들을 서버의 서로 다른 노드에 통신하여 전체 서버 효율성과 소프트웨어 서비스의 품질을 향상시키는 것이 유용하다. 이에 따라, 정상 동작 동안, 연산 서버의 패브릭은, 각각의 노드가 그 패브릭에서 고정된 어드레스를 갖고 메시지 경로설정 기법에 대한 경로설정 규칙을 나타내는 경로설정 정보(예를 들어, 경로설정 테이블)를 국부적으로 저장하는 메시지 경로설정 기법을 구현한다. 패브릭의 토폴로지는 개별적인 연산 노드에서의 오류나 고장으로 인해 시간이 지남에 따라 변경될 수 있기 때문에, 연산 노드들의 각각에 고정된 경로설정 정보를 영구 저장하는 것은 통상적으로 실현될 수 없다. 이러한 토폴로지 변경은 고정된 경로설정 정보의 상당 부분을 무효화하여, 서버 성능을 감소시킨다. 이에 따라, 연산 서버가 각 시스템 리셋 후에 패브릭을 구성하는 것이 유용하므로, 패브릭을 구성하는 동안, 패브릭은, 1) 패브릭의 기능 노드들에 의해 정의된 바와 같이 패브릭의 토폴로지를 식별하고, 2) 각 기능 노드에 고유 어드레스와 경로설정 정보를 분배한다. 이 프로세스를 본원에서는 연산 서버의 "구성"(또는 패브릭의 구성)이라고 한다.

[0006] 구성을 수행하기 위해, 종래의 클러스터 연산 서버들은, 통상적으로 정상 동작 동안 노드들 간에 메시지를 통신하는 데 사용되는 패브릭 상호접속부와는 별도로 "대역외" 네트워크를 사용한다. 그러나, 많은 연산 노드들이 있는 서버 시스템들에서, 대역외 네트워크는 노드 어드레스와 경로설정 정보를 분배하고 그 외에는 패브릭 노드들의 각각을 구성하는 데 많은 시간을 필요로 한다. 또한, 대역외 네트워크 자체에는 모든 구성 메시지들을 개별 노드들로 경로설정하는 인프라스트럭처가 있어야 하므로, 연산 서버의 복잡성과 비용이 증가한다.

[0007] 종래의 연산 서버와는 대조적으로, 본원에서 개시하는 기술들은, 연산 노드들의 각각에서 구성 필드들의 정의된 세트를 저장하기 위한 레지스터 또는 기타 저장 구조를 통합함으로써 노드 어드레스, 경로설정 정보, 및 기타 구성 정보를 통신하는 클러스터 연산 서버를 제공하며, 이에 의해, 각 필드가 대응하는 상태들의 제한된 세트 중 임의의 것을 취할 수 있다. 본원에서 구성 사이클이라고 하는 정의된 시간 간격에서, 서버의 각 노드는, 패브릭 상호접속부를 통해 자신에 접속된 다른 노드들의 각각에서의 구성 필드의 이전 상태 및 대응하는 구성 필드의 상태에 기초하여 (이전 상태로 유지되고 있거나 새로운 상태로 변하는) 자신의 구성 필드들의 각각의 상태를 설정한다. 노드는, 자신의 구성 필드들의 상태들에 기초하여, 구성 필드들의 상태들에 연관되고 미리 정의되어 있으며 저장된 구성 명령어들의 하나 이상의 세트를 실행한다. 따라서, 구성 필드들은, 연산 노드에서 실행되는 하나 이상의 작업을 정의하며, 자신의 접속된 노드들의 구성 레지스터들에 있는 대응 필드들의 상태들에만 의존한다. 따라서, 본원에서 개시하는 클러스터 연산 서버는, 대역외 구성 네트워크를 사용할 필요가 없으므로, 연산 서버 설계를 단순화하고 많은 수의 연산 노드들을 보다 신속하게 구성할 수 있다. 대신에, 본원에서 개시하는 기술들은, 이웃하는 노드들의 상태 변화들에 기초하여 구성 정보를 분배함으로써 가상 대역외 네트워크를 생성한다.

[0008] 일부 실시예들에서, 연산 노드들의 상태를 변경하는 것은, 토폴로지 분석과 구성 메시지 통신이라는, 클러스터 연산 서버에 대한 적어도 두 개의 구성 동작을 수행한다. 토폴로지 분석에 있어서, 연산 노드들의 구성 필드들의 상태는, 1) 기능하는 연산 노드들이 구성 메시지들의 통신을 위해 검출 및 준비되고 2) 연산 노드들이 스패닝 트리로 자기 조직되도록, 변한다. 일단 스패닝 트리가 조직되었다면, 패브릭 노드들은 구성 메시지들을 통신할 수 있으며, 노드들은 메시지 유형에 의해 암시되는 분배 규칙에 따라 메시지들을 자신들의 연결된 노드들에 분배한다. 클러스터 연산 서버의 하나 이상의 관리 유닛들은 스패닝 트리의 루트 연산 노드에서 구성 메시지를 개시함으로써 구성 정보를 주입하고, 각 노드에서의 분배 규칙에 의해 메시지가 스패닝 트리의 각 연산 노드에 도달한다. 이에 따라, 관리 유닛은, 경로설정 테이블, 노드 어드레스 정보 등의 구성 정보를 분배하여 정상 동작을 위한 노드를 준비할 수 있다.

[0009] 일부 경우에, 클러스터 연산 서버는, 스패닝 트리에서 오류가 있는 연산 노드의 검출 등의 정의된 오류 조건에 응답하여 토폴로지 분석에 다시 참여할 수 있다. 이어서, 연산 노드들은, 대응하는 구성 필드들을 대응하는 토폴로지 분석 상태들로 자동 복귀시키고 스패닝 트리를 재형성함으로써 스패닝 트리 토폴로지를 조정한다. 따라서 연산 노드들은 개별 연산 노드들의 오류 및 기타 오류를 자동으로 조정할 수 있다.

[0010] 용이하게 예시하기 위해, 서버의 구성은 도 1 내지 도 12를 참조하여 후술하는 바와 같이 클러스터 연산 서버의 예시적인 문맥에서 설명한다. 이러한 서버의 예로는 어드밴스트 마이크로 디바이시스사(Advanced Micro Devices, Inc.)의 씨마이크로(SeaMicro)(상표명) 부에 의해 제공되는 SM10000 시리즈 또는 SM15000 시리즈의 서버가 있다. 이하에서 일반적인 설명이 설명되어 있지만, 클러스터 연산 서버의 실시예들에 관한 추가 상세는

미국 특허 제7,925,802호 및 제8,140,719호에 개시되어 있으며, 이러한 문헌들의 전체 내용은 본원에 참고로 원용된다. 본 명세서에서 설명하는 기술들은 이러한 예시적인 문맥으로 제한되지 않고, 대신에 다양한 서버들 중 임의의 것에서 구현될 수도 있다. 또한, 이러한 기술들은 MAC 어드레스를 사용하는 이더넷 구현의 문맥으로 설명되지만, 이러한 기술들은 다양한 링크 계층 프로토콜들과 어드레싱 기법들 중 임의의 것으로 구현될 수도 있다.

[0011] 도 1은 일부 실시예들에 따른 클러스터 연산 서버(100)를 도시한다. 본원에서 "서버(100)"라고 하는 클러스터 연산 서버(100)는, 랙 유닛(RU) 시스템에 연산, 저장, 스위칭, 및 서버 관리를 함께 취하는 데이터 센터 플랫폼을 포함한다. 서버(100)는, 고-대역폭 저-지연 슈퍼컴퓨터 상호접속부를 포함하는 패브릭 상호접속부(112)에 의해 함께 링크된, 독립적인 저전력 연산 노드들(예를 들어, 연산 노드(101 내지 106)), 저장 노드들(예를 들어, 저장 노드(107 내지 109)), 네트워크 노드들(예를 들어, 네트워크 노드(110, 111)), 및 관리 노드들(예를 들어, 관리 유닛(113))의 병렬 어레이에 기초한다. 각 노드는, 효율적인 빌드업, 스케일링, 유지보수, 수리, 및 핫스왑 능력을 용이하게 하도록 인쇄 회로 기판(PCB) 기반 카드 또는 블레이드에 배치된 구성요소들을 포함하는 별도의 필드 교체가능 유닛(field replaceable unit: FRU)으로서 구현된다.

[0012] 연산 노드들은, 운영 체제(OS), 하이퍼바이저, 가상화 소프트웨어, 연산 애플리케이션 등을 비롯한 다양한 소프트웨어 프로그램들을 실행하도록 동작한다. 종래의 서버 노드들에서와 같이, 서버(100)의 연산 노드들은, 하나 이상의 프로세서, 및 하나 이상의 프로세서가 사용하기 위한 명령어들과 데이터를 저장하기 위한 시스템 메모리를 포함한다. 그러나, 종래의 서버 노드들과는 달리, 일부 실시예들에서, 연산 노드들은 저장 장치, I/O 제어기, 및 네트워크 인터페이스 카드(NIC) 등의 다양한 로컬 주변 장치들을 개별적으로 통합하지 않는다. 오히려, 서버(100)의 원격 주변 자원들은 연산 노드들 간에 공유되며, 이에 따라 I/O 제어기와 NIC 등의 서버 마더보드 상에서 통상적으로 발견되는 다수의 구성 요소가 연산 노드들로부터 제거될 수 있고, 패브릭 인터페이스 디바이스에 더하여 주로 그 하나 이상의 프로세서와 시스템 메모리를 남겨 둔다.

[0013] 시스템 리셋에 응답하는 구성 후에, 예를 들어, 주문형 집적 회로(ASIC)로서 구현될 수도 있는 패브릭 인터페이스 디바이스는, 서버(100)의 원격 공유 주변 자원들이 해당 프로세서의 로컬 주변 버스에 위치하는 각 프로세서에서 실행되는 OS에 보이도록, 그 공유 주변 자원들을 가상화하도록 동작한다. 이러한 가상화된 주변 자원들에는, 예를 들어, 대용량 저장 디바이스, 콘솔, 이더넷 NIC, 광섬유 채널 NIC, 인피니밴드(Infiniband)(상표명) NIC, HBA(저장 호스트 버스 어댑터), BIOS(기본 입력/출력 시스템), 유니버설 직렬 버스(USB) 디바이스, 파이어와이어(Firewire)(상표명) 디바이스, PCIe 디바이스, 사용자 인터페이스 디바이스(예를 들어, 비디오, 키보드, 및 마우스) 등이 있지만, 이러한 예들로 한정되지 않는다. 하드웨어에서의 원격 주변 자원들의 가상화와 공유는 원격 주변 자원들의 가상화를 연산 노드들에서 OS 및 기타 로컬 소프트웨어에 대하여 투명하게 만든다. 또한, 패브릭 인터페이스 디바이스를 통한 원격 주변 자원들의 가상화 및 공유는, 통상적으로 서버 마더보드에서 발견되는 다수의 구성요소 대신에 패브릭 인터페이스 디바이스를 사용할 수 있게 한다. 이는 각 연산 노드에서 구현되는 구성요소들의 수를 감소시키고, 이에 따라 연산 노드들은, 별도의 개별적인 주변장치 자원들을 구현하는 종래의 서버 블레이드들보다 적은 에너지를 소비하면서 더욱 작은 폼 팩터를 가질 수 있다.

[0014] 저장 노드와 네트워크 노드(총괄하여 "입력/출력(I/O) 노드"라고 함)는 하나 이상의 공유 주변 자원들을 관리하는 주변 디바이스 제어기를 구현한다. 이 제어기는, 연산 노드들의 패브릭 인터페이스 디바이스들과 협동하여 자원 관리자가 관리하는 주변 자원들을 가상화하고 공유한다. 예시를 위해, 저장 노드(107)는 하드 디스크 드라이브(HDD)(116)를 관리하고, 저장 노드(108)는 SSD(118)를 관리한다. 일부 실시예들에서, 임의의 내부 대용량 저장 디바이스는 임의의 프로세서를 장착할 수 있다. 또한, 대용량 저장 디바이스는 슬라이스들, 또는 "가상 디스크들"로 논리적으로 분리될 수도 있으며, 가상 디스크들의 각각은 단일 연산 노드에 할당될 수도 있고, 또는, 판독 전용 모드에서 사용된다면 대형 공유 데이터 캐시로서 다수의 연산 노드에 의해 공유될 수도 있다. 가상 디스크의 공유를 통해, 사용자는 운영 체제, 애플리케이션 소프트웨어, 및 캐시된 데이터 등의 공통 데이터를 전체 서버(100)에 대하여 한 번 저장하거나 갱신할 수 있다. I/O 노드들에 의해 관리되는 공유 주변 자원들의 다른 일례로, 저장 노드(109)는, 원격 BIOS(120), 콘솔/범용 비동기 송수신기(UART)(121), 및 데이터 센터 관리 네트워크(123)를 관리한다. 네트워크 노드들(110 및 111) 각각은 데이터 센터 네트워크(114)에 접속된 하나 이상의 이더넷 업링크들을 관리한다. 이더넷 업링크는, 탑-오브(top-of) 랙 스위치의 업링크 포트와 유사하며, 예를 들어, 데이터 센터 네트워크(114)의 엔드-오브-로우(end-of-row) 스위치 또는 코어 스위치에 직접 접속하도록 구성될 수 있다. 원격 BIOS(120)는 서버의 노드들의 일부 또는 모두에 대한 로컬 BIOS로서 동작하도록 대용량 저장 디바이스, NIC, 및 기타 주변 자원들과 동일한 방식으로 가상화될 수 있어서, 이러한 노드들이 각 노드에서 로컬 BIOS의 적어도 일부를 구현하지 않을 수 있다. 일부 실시예들에서, 서버의 노드들 각각은 시스템 리

셋에 응답하여 실행되는 로컬 BIOS를 포함한다. 로컬 BIOS를 실행함으로써, 각 노드가 본원에서 더 설명하는 구성 프로세스들에 참여할 수 있다. 구체적으로, 로컬 BIOS의 실행은, 노드의 상태에 따라 그 노드에서의 작업들의 실행을 제공하고, 노드의 이전 상태와 그 노드에 접속된 노드들의 상태에 따라 노드들을 서로 다른 상태들로 천이시키는 것을 제공한다.

[0015] 연산 노드들의 패브릭 인터페이스 디바이스, I/O 노드들의 패브릭 인터페이스들, 및 패브릭 상호접속부(112)는 연산 노드들의 연산 자원들을 I/O 노드들의 주변 자원들과 접속하는 패브릭(122)으로서 함께 동작한다. 이를 위해, 패브릭(122)은 분산형 스위칭 설비를 구현하고, 이에 의해, 패브릭 인터페이스 디바이스들과 패브릭 인터페이스들 각각은, 패브릭 상호접속부(112)의 양방향 링크들에 접속된 다수의 포트를 포함하고, 시스템 리셋에 응답하는 패브릭 상호접속부(112)의 구성 후에, 서버(100)의 노드들에서 구현된 결정론적 경로설정 로직에 따라 그 포트들 간에 패킷 트래픽을 경로설정하기 위한 링크 계층 스위치들로서 동작한다. "링크 계층"이라는 용어는 일반적으로 개방 시스템 상호접속(OSI) 모델의 데이터 링크 계층, 또는 계층(2)을 가리킨다는 점에 주목한다.

[0016] 패브릭 상호접속부(112)는, 백플레인, 인쇄 배선 기판, 마더보드, 케이블링 또는 다른 가요성 배선, 또는 이들의 조합 등의 고정된 또는 가요성 상호접속부를 포함할 수 있다. 또한, 패브릭 상호접속부(112)는 전기 신호, 광 신호, 또는 이들의 조합을 포함할 수 있다. 일부 실시예들에서, 패브릭 상호접속부(112)의 링크들은, PCIE(Peripheral Component Interconnect-Express) 표준, 래피드(Rapid) IO 표준, 로켓(Rocket) IO 표준, 하이퍼-트랜스포트(Hyper-Transport) 표준, 파이버채널(FiberChannel) 표준, 또는 기가비트 이더넷(GbE) 부착 유닛 인터페이스(XAUI) 표준 등의 이더넷 기반 표준 중 하나 이상에 따라 구현된 고속 양방향 직렬 링크들을 포함한다.

[0017] 노드를 구현하는 FRU가 통상적으로 도 12를 참조하여 후술되는 바와 같이 서버 박스 내의 하나 이상의 행에 물리적으로 배열되지만, 패브릭(122)은, 구성 중에, 다양한 메시 토폴로지들 또는 토러스, 다차원 토러스(k-ary n-큐브라고도 함), 트리, 팻 트리 등의 기타 네트워크 토폴로지들 중 임의의 것에서 노드들을 논리적으로 배열할 수 있다. 예시를 위해, 서버(100)는 본원에서 다차원 토러스 네트워크 토폴로지의 문맥으로 설명된다. 그러나, 설명되는 기술들은 본원에 제공되는 가이드라인을 이용하여 다른 네트워크 토폴로지들에 유사하게 적용될 수도 있다.

[0018] 연산 노드들(101 내지 106)의 각각은, 대응하는 연산 노드에 대한 구성 필드들의 세트를 저장하기 위한 구성 상태 레지스터(예를 들어, 연산 노드(101)에서의 구성 상태 레지스터(170))를 포함한다. 각 구성 필드는, 본원에서 더 설명하는 바와 같이 대응하는 연산 노드의 구성의 구체적인 양태에 대한 상태 정보를 저장한다. 예를 들어, 구성 필드들 중 하나는, 서버(100)의 노드들을 매핑하는 스패닝 트리에서 대응하는 연산 노드에 접속된 노드들에 대하여 그 대응하는 연산 노드의 위치를 나타내는 구성 정보를 저장할 수 있다. 동작시, 각 연산 노드는, 접속된 연산 노드들의 구성 상태 레지스터들의 구성 필드들을 주기적으로 체크하고, 이러한 필드들의 값들에 기초하여, 자신의 고유한 구성 상태 레지스터의 구성 필드들에서의 값들을 갱신한다. 연산 노드는, 자신의 구성 상태 레지스터의 구성 필드들의 값들에 기초하여, 구성 메시지들의 내부 처리, 자신의 접속된 노드들로의 구성 메시지들의 통신, 구성 메시지들에 응답하는 데이터 생성 등의 정의된 구성 동작들을 수행한다. 또한, 연산 노드는, 자신의 접속된 노드들에서의 천이를 야기하고 그러한 노드들에서의 상태들의 변화를 관찰함으로써, 접속되지 않은 노드들(원격 노드들)의 상태와 구성을 식별할 수 있고, 본 명세서에서 설명하는 바와 같이 패브릭을 통해 전파될 때 원격 노드들에서의 상태와 구성의 변화를 야기하는 메시지들을 생성할 수 있다.

[0019] 각 연산 노드를 구성하기 위해, 각 연산 노드의 구성 상태 레지스터들의 구성 필드들은, 관리 노드(113)에 의해 주입되는 구성 정보에 기초하여 시간이 경과함에 따라 전개된다. 각 구성 필드에 대한 상태 갱신은, 구성 필드들의 전개에 따라 각 연산 노드가 패브릭 상호접속부(112)의 토폴로지의 고유한 어드레스 및 각 노드의 고유한 어드레스에 따라 메시지들이 노드들 간에 경로설정될 수 있게 하는 경로설정 정보를 수신하도록 정의된다. 이에 따라, 연산 노드들은 서버(100)의 정상적(포스트 구성) 동작 동안 유니캐스트 메시지들의 경로설정을 위해 준비된다.

[0020] 도 2는 일부 실시예들에 따라 k-ary n-큐브 또는 다차원 토러스로서 배치된 네트워크 토폴로지의 서버(100)의 구성의 일례를 도시한다. 도시한 예에서, 서버(100)는 깊이 3을 갖는(즉, k=n=3) 3차원(30) 토러스 네트워크 토폴로지(본원에서 "토러스 네트워크(200)"라 함)를 구현한다. 이에 따라, 서버(100)는, 3개의 직교 차원(X, Y, Z)으로 형성된 링들의 네트워크에 배열된 총 27개 노드를 구현하며, 각 노드는 서로 다른 3개의 링의 멤버이며, 차원들의 각각당 하나의 링이 해당한다. 각 노드는 패브릭 상호접속부(112)(도 1 참조)의 양방향 직렬 링크를

통해 최대 6개의 이웃하는 노드들에 접속된다. 토러스 네트워크(200)에서의 각 노드의 상대 위치는 도 2에서 위치 튜플(tuple)(x, y, z)에 의해 식별되며, 여기서, X, Y, Z는 X, Y, Z 차원의 연산 노드의 위치를 각각 나타낸다. 이처럼, 노드의 튜플(x, y, z)은, 또한, 토러스 네트워크(200) 내의 어드레스로서 기능할 수도 있으며, 따라서, 위치 튜플(x, y, z)에 의해 표현되는 위치에서 목적지 노드로 패킷들을 경로설정하기 위한 소스 경로설정 제어부로서 기능할 수 있다.

[0021] 일부 실시예들에서, 시스템 리셋에 응답하여, 패브릭 상호접속부(112)는, 하나 이상의 미디어 액세스 컨트롤(MAC) 어드레스가 주어진 노드에 일시적으로 또는 영구적으로 연관되도록 각 노드를 구성한다. 이렇게 연관된 MAC 어드레스의 일부 또는 전부는 위치 튜플(x, y, z)을 직접적으로 나타낼 수도 있으며, 이는 토러스 네트워크(200)의 목적지 노드의 위치가 결정되게 하고 패킷의 목적지 MAC 어드레스에 기초하여 소스 경로설정되게 할 수 있다. 구성 동안, 튜플 변환을 위치시키기 위한 MAC 어드레스의 분산형 룩업 테이블은, 목적지 MAC 어드레스에 기초하여 목적지 노드의 위치의 식별을 용이하게 하도록 노드들에서 캐싱될 수도 있다.

[0022] 도시된 X, Y, 및 Z 차원은 네트워크에서 각 노드의 위치를 설명하는 논리적 차원을 나타내지만 반드시 각 노드의 물리적 배치를 나타내는 물리적 차원을 나타내는 것은 아니라는 점을 인식할 것이다. 예를 들어, 토러스 네트워크(200)에 대한 3D 토러스 네트워크 토폴로지는, 백플레인 상의 하나 이상의 행에 또는 랙에 물리적으로 배열된 네트워크의 노드들을 갖는 패브릭 상호접속부(112)의 와이어링을 통해 구현될 수 있다. 즉, 토러스 네트워크(200)의 주어진 노드의 상대 위치는, 연산 노드의 물리적 위치보다는 그 노드에 접속되어 있는 노드들에 의해 정의된다. 일부 실시예들에서, 패브릭(122)(도 1 참조)은, 3D 토러스 네트워크 토폴로지를 구현하도록 패브릭 상호접속부(112)를 통해 함께 와이어링된 복수의 소켓을 포함하고, 그 노드들의 각각은, 토러스 네트워크(200)의 노드의 위치가 FRU가 삽입되는 소켓에 의해 지시되게끔, 패브릭 상호접속부(112)에 의해 사용되는 소켓들에 연결하도록 구성된 필드 교체가능 유닛(FRU)을 포함한다.

[0023] 서버(100)에서, 패브릭 상호접속부(112)의 구성 후에, 노드들 간에 통신되는 메시지들은 하나 이상의 패킷으로 세그먼트화되고, 이러한 패킷들은 소스 노드와 목적지 노드 간의 경로설정 경로를 통해 경로설정된다. 경로설정 경로는 0개, 1개, 또는 1개보다 많은 중간 노드를 포함할 수도 있다. 전술한 바와 같이, 각 I/O 노드를 포함하는 각 노드는, 패브릭 상호접속부(212)의 대응하는 링크들에 접속된 노드의 포트들 간에 패킷들을 경로설정하기 위한 링크 계층 스위치를 구현하는, 패브릭 상호접속부(112)에 대한 인터페이스를 포함한다. 일부 실시예들에서, 이들의 구성 후에, 분산형 스위치들은, 패브릭 데드락(deadlock)을 피하는 데 일조하는 엄격한 결정론적 차원-순서 경로설정 기법(즉, 다른 차원으로 이동하기 전에 1차원에서 토러스 네트워크(200)를 완전히 횡단함) 등의 소스 경로설정 또는 소스 경로설정된(source routed) 기법을 이용하여 패브릭(122)에 걸쳐 패킷들을 경로설정하도록 동작한다. 엄격한 결정론적 차원-순서 경로설정의 일례를 예시하도록, 위치(0, 0, 0)의 노드로부터 위치(2, 2, 2)로 송신되는 패킷은, 노드(0, 0, 0)로부터 노드(1, 0, 0)로 X 차원에서 초기에 송신된다면, X 차원에서 노드(2, 0, 0)로 계속되며, 이에 따라, Y 평면에서 노드(2, 0, 0)로부터 노드(2, 1, 0)로 이동한 후, 노드(2, 2, 0)로 이동하고, 이어서 Z 평면에서 노드(2, 2, 0)로부터 노드(2, 2, 1)로 이동한 후 노드(2, 2, 2)로 이동한다. 평면들이 소스와 목적지 간에 완전히 횡단되는 순서는 미리 구성될 수도 있으며 각 노드마다 다를 수도 있다.

[0024] 또한, 토러스 네트워크(200)의 노드들 간에 다수의 경로가 있으므로, 패브릭(212)은, 구성 프로세스 동안, 주 경로 장애가 있는 경우 패킷 트래픽이 2차 경로를 횡단하도록 구성될 수 있다. 패브릭(212)은, 또한, 패킷 클래스와 가상 채널을 구현하여 링크 대역폭을 보다 효과적으로 활용하고 패킷 루프를 제거하도록 구성될 수 있으므로, 링크 레벨 루프 방지 및 스패닝 트리 프로토콜 등의 리던던시 프로토콜을 필요로 하지 않을 수 있다.

[0025] 종래에, 소정 유형의 노드들은 서버에서 소프트웨어 서비스의 정상적인 실행 동안 자신들의 경로설정 능력이 제한되도록 구성된다. 예를 들어, 연산 노드들은 패킷의 소스 노드와 패킷의 목적지 노드 간의 패킷의 경로설정 경로에 존재하는 중간 노드들로서 기능 수 있는 반면, I/O 노드들은 패킷을 다른 노드들로 경로설정하는 중간 노드가 아닌 소스 노드 또는 목적지 노드로서만 동작하도록 구성된다. 도시된 실시예에서, 각 I/O 노드는 연산 노드들과 유사한 방식으로 패킷들을 경로설정하도록 구성되며, 이에 따라 모든 노드들이 유사한 경로설정 능력을 제공한다.

[0026] 패브릭(122)은 다양한 패킷 경로설정 및 기술 프로토콜을 구현하도록 구성될 수도 있다. 예를 들어, 각 노드의 스위치에서 큰 버퍼가 필요하지 않도록, 패브릭(122)은, 구성 후에, 흐름 제어 디지트("플릿") 기반 스위칭을 사용할 수 있으며, 이에 따라 각 패킷이 플릿들의 시퀀스로 세그먼트화된다. 헤더 플릿이라고 하는 제1플릿은, 패킷의 경로(즉, 목적지 어드레스)에 대한 정보를 보유하고, 패킷에 연관된 모든 후속 플릿에 대한 경로설정 거

동을 설정한다. 헤더 플릿 뒤에는, 데이터의 실제 페이로드를 포함하는 0개 이상의 바디 플릿이 뒤따른다. 테일 플릿이라고 하는 최종 플릿은, 경로설정 경로의 모든 중간 노드들뿐만 아니라 소스와 목적지 노드들에도 할당된 자원들을 해제하도록 어떤 기록을 수행한다. 이어서, 이러한 플릿들은, 패킷 레벨로 버퍼와 채널 대역폭을 할당하는 컷스루(cut-through) 경로설정 또는 플릿 레벨로 버퍼와 채널 대역폭을 할당한 웜홀 경로설정을 이용하여 토러스 네트워크(200)를 통해 경로설정될 수도 있다. 웜홀 경로설정은 토러스 네트워크(200)에서 가상 채널을 사용할 수 있는 장점이 있다. 가상 채널은, 경로의 다음 홉(hop)을 위한 현재 노드의 출력 채널 및 가상 채널의 상태(예를 들어, 유휴, 자원 대기, 또는 활성)를 포함하는 채널을 통해 패킷의 플릿 처리를 조정하는 데 필요한 상태를 유지한다. 가상 채널은, 또한, 현재 노드에서 버퍼링된 패킷의 플릿에 대한 포인터 및 다음 노드에서 사용가능한 플릿 버퍼의 수를 포함할 수도 있다.

[0027] 도 3은 패브릭(122)의 노드(301) 및 그 노드에 접속된 노드들(302, 303, 304, 305, 306 및 307)을 도시한다. 노드(301)는 대응하는 통신 포트를 통해 각 노드(302 내지 307)에 접속된다. 일부 실시예들에서, 노드(301)는 3D 토러스 토폴로지에서 노드들의 예상되는 상대 위치에 따라 그 노드들(302 내지 307)의 각각을 내부적으로 식별한다. 따라서, 노드(302)는 노드(301)에 의해 "+X" 노드로서 식별되어, 노드(302)가 노드(301)에 대한 3D 토러스의 X 평면을 따라 한 방향으로 있을 것으로 예상된다. 노드(305)는 노드(301)에 의해 "-X" 노드로서 식별되어, 노드(302)가 노드(301)에 대한 3D 토러스의 X 평면을 따라 노드(302)의 반대 방향으로 있을 것으로 예상된다. 유사한 이유로, 노드(303)는 노드(301)에 의해 "+Y" 노드로서 식별되고, 노드(306)는 노드(301)에 의해 "-Y 노드"로서 식별되고, 노드(304)는 노드(301)에 의해 "+Z" 노드로서 식별되고, 노드(307)는 노드(301)에 의해 "-Z" 노드로서 식별된다.

[0028] 시스템 리셋에 응답하여, 노드(301)는 자신의 구성 필드들의 각각의 현재 상태를 나타내는 토큰들을 패브릭 상호접속부(122)를 통해 자신에 접속된 노드들(302 내지 307)에 주기적으로 송신한다. 또한, 노드(301)는, 접속된 노드들(302 내지 307)의 각각에 대한 구성 필드들의 각각의 대응하는 상태들을 나타내는, 접속된 노드들(302 내지 307)의 각각으로부터의 토큰들을 패브릭 상호접속부(122)를 통해 수신한다. 구성 필드들의 현재 상태 및 접속된 노드들(302 내지 307)에서의 구성 필드들의 상태에 기초하여, 노드(301)는 자신의 고유한 구성 필드들 상태의 상태들을 조정하고, 조정된 상태에 의해 요구되는 임의의 처리 동작을 실행한다.

[0029] 도 4는 일부 실시예들에 따른 구성 상태 레지스터(CSR)(170)의 일례를 도시한다. 도시한 예에서, CSR(170)은, 어드레스 및 배향 필드(401), 스패닝 트리 상태 필드(402), 인터럽트 포워딩 상태 필드(403), 웨이브 메시지 상태(404), 및 체인 메시지 상태(405)를 포함하는 다수의 구성 필드에 대한 값들을 저장한다. 구성 필드들(401 내지 405)의 각각은 해당 필드의 상태를 나타내는 값을 저장한다. 각 구성 사이클 동안, 연산 노드(102)는 각 구성 필드의 상태에 기초하여 구성 동작을 취한다. 또한, 각 구성 사이클의 종료까지, 연산 노드(102)는 자신의 보정된 노드들의 각각에서 대응하는 구성 필드들의 값들에 기초하여 구성 필드들(401 내지 405) 각각을 갱신한다.

[0030] 예시하자면, 어드레스 및 배향 필드(401)는 연산 노드(102)에 연관된 FRU의 배향을 나타내는 정보를 저장한다. 시스템 리셋 후에, 연산 노드(102)는 어드레스 및 배향 필드(401)에 값을 설정하여 FRU의 배향이 알려지지 않았음을 나타낸다. 접속된 노드들 중 하나에서의 어드레스 및 배향 필드가 대응하는 FRU에 대한 특정 배향을 나타내는 것에 응답하여, 연산 노드(102)는 어드레스 및 배향 필드(401)를 갱신하여 연산 노드(102)에 관련된 FRU가 접속된 노드의 FRU와 동일한 배향을 갖는다는 것을 나타낸다.

[0031] 스패닝 트리 상태 필드(402)는, 연산 노드(102)가 패브릭 상호접속부(112)의 노드들에 대한 스패닝 트리에 합류할 준비가 되어 있는지 여부를 나타내는 정보를 저장하고, 일단 합류하면, 노드 스패닝 트리에서의 접속된 노드들에 대한 자신의 위치를 저장한다. 예를 들어, 시스템 리셋 후에, 연산 노드(102)는 스패닝 트리 상태 필드(402)에서의 값을 IDLE 상태로 설정하여, 연산 노드(102)가 아직 스패닝 트리에 합류하지 않았음을 나타낸다. 접속된 노드들 중 하나에서의 스패닝 트리 상태 필드가 READY 상태에 있는 것에 응답하여, 연산 노드(102)는 스패닝 트리 상태 필드(402)를 READY 상태로 설정하여, 그 연산 노드가 스패닝 트리에 합류할 준비가 되어 있음을 나타낸다. 연산 노드(102)는, 접속된 노드들의 각각에서의 스패닝 트리 상태 필드의 상태들의 후속 변화에 기초하여, 자신에 접속된 노드들에 대한 스패닝 트리의 연산 노드의 위치를 나타내도록 스패닝 트리 상태 필드(402)의 상태를 전개한다. 본원에서 더 설명되는 바와 같이, 서버(100)의 연산 노드들의 각각에서 스패닝 트리 상태 필드들의 전개로부터 발생하는 스패닝 트리는 각 노드를 구성하도록 구성 메시지들을 통신하는 데 사용된다.

[0032] 인터럽트 포워딩 상태 필드(403)는 연산 노드(102)의 접속된 노드들로부터 수신되거나 그 노드들로 송신되는 인터럽트 메시지들의 상태를 나타내는 정보를 저장한다. 예를 들어, 일부 실시예들에서, 연산 노드(102)가 패브릭

상호접속부(112)의 노드들에 대한 스패닝 트리에 합류한 후에, 연산 노드는, 자신의 접속된 노드들에서의 인터럽트 포워딩 상태 필드들의 상태에 기초하여 인터럽트 포워딩 상태 필드(403)의 상태를 설정하여, 그러한 접속된 노드들로부터 수신되는 인터럽트의 상태를 반영하고, 자신의 접속된 노드들 중 하나로부터 인터럽트 메시지를 수신한 경우, 자신의 접속된 노드들 중 다른 하나에 인터럽트 메시지를 포워딩하였는지 여부, 인터럽트 메시지에 대한 응답이 수신되었는지 여부 등을 반영한다. 또한, 연산 노드(102)는, 포워딩 필드(403)의 상태에 기초하여 수신된 메시지를 다른 접속된 노드들에 포워딩하는 것을 포함하여, 포워딩 상태 필드(403)의 상태에 기초하여 수신된 인터럽트 메시지 및 응답을 처리한다. 이에 따라, 인터럽트 포워딩 필드(403)는, 개별 노드들의 직접 어드레싱을 사용하지 않고 패브릭 상호접속부(112)를 통해 인터럽트 메시지들의 포워딩을 위한 저-지연 기구를 제공한다.

[0033] 웨이브 메시지 상태 필드(404) 및 체인 메시지 상태 필드(405) 각각은 본원에서 더 설명되는 바와 같이 특정 구성 메시지 유형의 처리 상태를 나타내는 값을 각각 저장한다. 구성 동안, 연산 노드(102)는, 메시지들이 정의된 프로토콜에 따라 처리되는 것을 보장하도록 각 유형의 수신된 메시지에 기초하여 필드들(404 및 405)의 각각의 상태를 전개한다.

[0034] 일부 실시예들에서, 연산 노드(102)는, 연산 노드(102)에 의해 실행될 때, BIOS 코드나 다른 구성 정보에 의해 정의된 대응하는 공식들에 따라 구성 필드들(401 내지 405)의 각각을 갱신하는 그 BIOS 코드나 다른 구성 정보를 저장한다. 이는, 각 구성 필드의 상태 정보가 식에 의해 표시된 바와 같이 정의된 프로세스에 따라 전개되어 각 노드가 정의된 구성 프로세스에 따라 구성되는 것을 보장한다. 식들은 구성 노드 또는 다른 제어 노드에 의한 상태 정보의 직접 통신 없이 구성 필드들의 전개를 제공하기 때문에, 특별한 대역의 구성 네트워크가 필요 없게 된다.

[0035] 구성 필드의 전개를 제공하기 위한 식의 일반식은 아래와 같다:

[0036] $$S_{i,n+1} = f(\{S_{j,n} | j \in N(i)\})$$

[0037] 여기서, $S_{i,n}$은 구성 사이클 n에서의 노드 S에 대한 구성 필드의 상태이고, N(i)는 노드 S에 대하여 주목하는 접속된 셀들이다. 특정 함수와 N(i)는 각 구성 필드마다 다를 수 있으며, N(i)는 구성 필드들 중 하나 이상의 상태에 따라 변할 수 있어서, 구성 필드의 더욱 복잡한 전개를 제공한다.

[0038] 도 5는 일부 실시예들에 따른 연산 노드(102)의 구성 필드에 대한 상태의 세트(500)의 일례를 도시한다. 세트(500)는 구성 필드에 저장된 상이한 값에 의해 각각 표현되는 상태들(501, 502, 503 및 504)을 포함한다. 상태들(501 내지 504) 간의 화살표는 연산 노드(102)에 접속된 노드들 중 하나 이상의 대응하는 구성 필드들에서의 값의 변화로부터 발생하는 상태들 간의 천이를 나타낸다.

[0039] 예시하자면, 일부 실시예들에서, 세트(500)는 도 1의 스패닝 트리 상태(402)에 대한 상태들의 세트이다. 시스템 리셋에 응답하여, 스패닝 트리 상태(402)는 IDLE 상태를 나타내는 상태(501)에 배치된다. IDLE 상태에서, 연산 노드(102)는 그 IDLE 상태를 나타내는 토큰을 자신의 접속된 노드들에 주기적으로 송신한다.

[0040] 대응하는 접속된 노드가 READY 상태에 진입한 것을 나타내는 토큰(설명을 위해 "TOPO" 토큰이라 함)을 자신의 접속된 노드들 중 적어도 하나로부터 수신함에 응답하여, 연산 노드(102)는 스패닝 트리 상태(402)를 READY 상태를 나타내는 상태(502)로 천이시킨다. READY 상태에 있는 동안, 스패닝 트리에서의 접속된 노드들의 상대 위치를 나타내는, 자신의 접속된 노드들 중 하나로부터 스패닝 트리 정보를 수신함에 응답하여, 연산 노드(102)는 스패닝 트리 상태(402)를 상태(503)로 천이시킨다. 상태(503)에서, 연산 노드(102)는 자신의 접속된 노드들에 대한 스패닝 트리에서의 자신의 위치를 식별한다. 예를 들어, 일부 실시예들에서, 연산 노드는 TOPO 토큰을 송신한 접속된 노드를 스패닝 트리의 루트에 더욱 가까운(더욱 근위에 있는) 노드로서 식별한다. 따라서, TOPO 토큰을 송신하는 노드는 스패닝 트리에 대한 "근위 노드"로서 연산 노드(102)에 의해 식별된다. 연산 노드(102)는 READY 상태(502)로 다시 천이할 수 있다.

[0041] 또한, READY 상태(502)에서, 연산 노드(102)는, 연산 노드(102)가 접속된 노드들 중 하나 이상에 대한 근위 노드들로서 확립되었음을 나타내는 토큰들을 자신의 접속 노드들로부터 수신할 수 있다. 이에 응답하여, 연산 노드(102)는 스패닝 트리 상태(402)를 상태(504)로 천이시킨다. 스패닝 트리 상태(402)가 상태(504)인 경우, 연산 노드(102)는 자신의 접속된 노드들 중 어느 것이 스패닝 트리의 "원위 노드"인지를 나타내는 정보를 저장한다. 자신의 근위 노드와 원위 노드들을 식별함으로써, 연산 노드(102)는 스패닝 트리의 전체 토폴로지를 식별하지 않고 자신의 접속된 노드들에 대하여 스패닝 트리에서 자신의 위치를 식별한다. 이는 패브릭(122)의 노드들의

각각에서의 구성을 간략화한다.

[0042] 일부 경우에, IDLE 상태(501)에서, 연산 노드(102)는, 다중 노드들의 각각이 IDLE 상태(501)로부터 READY 상태(502)로 천이하였음을 나타내는 TOPO 토큰들을 자신의 접속된 노드들 중 다수의 노드로부터 동시에 수신할 수 있다. 이에 응답하여, 연산 노드(102)는, READY 상태로 천이하고, BIOS 코드에 반영되어 있는 미리 정의된 협약에 따라, 스패닝 트리에서 자신의 근위 노드로서 TOPO 토큰을 송신한 접속된 노드들 중 하나를 식별한다. 예를 들어, 일부 실시예들에서, 각 노드는 카운터를 포함하고, 이에 의해, 카운터는 시스템 리셋에 응답하여 초기에 0으로 설정된다. READY 상태(502)로의 천이에 응답하여, 노드는 자신의 카운터를 증분하고, 증분된 값을, 자신의 접속된 노드들에 송신하는 TOPO 토큰에 통합된 트리-깊이 필드로서 통신한다. 유휴 상태(501)에 있는 동안 단일 TOPO 토큰을 수신한 것에 응답하여, 노드는 자신의 카운터를 트리-깊이 필드의 값으로 설정한다. 이에 따라, 각 노드의 카운터는 스패닝 트리에서 노드의 깊이를 나타낸다. IDLE 상태에 있는 동안 다수의 TOPO 토큰을 수신한 것에 응답하여, 노드는, 최저 값을 갖는 트리-깊이 필드를 갖는 TOPO 토큰을 선택하고, 그 값을 증분하고, 증분된 값을 자신의 고유한 카운터에 저장하고, 대응하는 접속된 노드가 스패닝 트리에서의 인접 노드임을 나타내는 정보를 저장한다. 이에 따라, 각 노드는, 스패닝 트리에 하나의 근위 노드만을 갖지만, 다수의 원위 노드들 가질 수 있다.

[0043] READY 상태(502)에 있는 동안, 연산 노드(102)는, 자신의 근위 노드가 소정의 종류의 장애를 겪었음을 나타내는 토큰을 수신하거나 그 외에는 식별할 수 있다. 이에 응답하여, 연산 노드(102)는 IDLE 상태(501)로 복귀한다. 후속하여, 연산 노드는, 자신의 다른 노드들 중 하나가 READY 상태에 있음을 나타내는 토큰을 수신하여, 연산 노드(102)가 READY 상태(502)로 복귀할 수 있게 하고 다른 접속된 노드들에 대해 스패닝 트리에서 자신의 위치를 재확립할 수 있게 한다. 따라서, 패브릭(122)의 노드들은 구성 동안 개별 노드들의 장애에 적응할 수 있어서, 구성 프로세스의 견고성을 향상시킨다.

[0044] 세트(500)는 연산 노드(102)에 대한 구성 필드들 중 하나의 구성 필드만에 대한 상이한 상태들을 나타내며, 각 구성 필드는 도 5에 도시된 상태들과는 상이한 대응하는 상태들의 고유한 세트를 가질 수 있음을 인식할 것이다. 또한, 연산 노드(102)는 자신의 접속된 노드들에서 대응하는 구성 필드들의 상태들에 기초하여 각 구성 사이클에서 자신의 구성 필드들 중 하나보다 많은 구성 필드의 상태를 조정할 수 있다. 예를 들어, 일부 경우에, 연산 노드(102)는, 동일한 구성 사이클 동안, 자신의 인터럽트 포워딩 상태의 변화를 나타내는 토큰 및 자신의 웨이브 메시지 상태의 변화를 나타내는 토큰을 자신의 접속된 노드들 중 하나로부터 수신할 수 있다. 이에 따라, 구성 사이클 동안, 연산 노드(102)는, 토큰들에 응답하여, 자신의 인터럽트 포워딩 상태 및 자신의 웨이브 메시지 상태를 갱신할 수 있고, 양측 필드에 대한 갱신 상태들에 대응하는 구성 동작들을 취할 수 있다. 이는 동일한 구성 사이클 동안, 연산 노드(102)의 다수의 양태의 구성을 가능하게 하여, 구성 프로세스의 효율을 향상시킨다.

[0045] 도 6은 일부 실시예들에 따라 서버를 구성하는 방법(600)의 흐름도를 도시한다. 설명을 위해, 방법(600)은 도 1 내지 도 5에 설명된 패브릭(122)에서의 예시적인 구현예에 대하여 설명한다. 블록(602)에서, 서버(100)는, 소프트웨어 리셋, 리셋 스위치의 활성화 등에 응답하여 서버(100)에서 사이클링되는 전력에 응답하여 발생할 수 있는 것과 같은 시스템 리셋을 경험한다. 시스템 리셋에 응답하여, 패브릭(122)의 각 노드는 리셋된다. 블록(604)에서, 패브릭(122)의 노드들의 각각은, 본원에서 구성 클록이라고 하는 로컬 클록을 자신의 접속된 노드들의 구성 클록들과 동기화시킨다. 각 노드는 자신의 구성 클록에 기초하여 자신의 구성 필드들의 상태들 간의 천이를 제어한다. 일부 실시예들에서, 접속된 노드들의 구성 클록들은 각 노드 내에서 진행되는 피드백 프로세스를 통해 동기화된다. 피드백 프로세스는, 노드의 구성 클록이 정의된 허용 오차 내에서 자신의 접속된 노드들과 동기화되게 하는 것이다. 정의된 허용 오차는 모든 노드들의 구성 클록들이 전체 허용 오차 내에서 동기화되도록 정의된다. 전체 허용 오차는, 모든 노드가 임의의 필요한 상태 천이를 완료했을 것으로 예상되는 주기적인 인스턴스를 확립한다. 이러한 주기적인 인스턴스들에 의해 정의된 시간을 본원에서 "구성 사이클"이라고 한다. 구체적으로, 구성 사이클들은 패브릭(122)의 노드들에서의 상태 천이를 위한 전체 타이밍 구조를 확립하고, 이에 의해, 각 노드에서의 상태 천이들은 구성 사이클의 시작시 또는 구성 사이클의 시작 후에 개시될 것으로 예상되며, 두 개의 임의의 주어진 노드의 구성 클록들이 완전히 동기화되지 않을 수 있더라도 동일한 구성 사이클의 종료 시 또는 종료 전에 완료될 것으로 예상된다.

[0046] 블록(606)에서, 패브릭(122)의 노드들은 도 5와 관련하여 전술한 바와 유사한 방식으로 IDLE 상태에서 READY 상태로 천이한다. 일부 실시예들에서, 본원에서 더 설명하는 바와 같이, 패브릭(122)은 패브릭(122)의 구성을 제어하는 관리 유닛(MU)이라고 하는 적어도 하나의 프로세서를 포함한다. MU는, 패브릭(122)의 각 노드를 구성하기 위해, 경로설정 테이블, 노드 어드레스 등의 구성 정보를 포함하는 메모리에 접속된다. MU는 패브릭(122)의

연산 노드들 중 하나에 접속된다. 시스템 리셋에 응답하여, MU는 자신의 고유한 BIOS 코드를 실행하여 연산 노드를 IDLE 상태에서 READY 상태로 천이하는 커맨드를 자신의 접속된 노드들에 송신한다. 이는 패브릭(122)의 각 기능 노드가 결국 READY 상태로 천이하게 한다.

[0047] 노드들이 READY 상태로 천이함에 따라, 블록(608)에서, 노드들은 도 5와 관련하여 전술한 바와 같이 스패닝 트리로 자기 조직된다. 블록(610)에서, MU는 CHAIN 및 WAVE 메시지들 등의 메시지들을 사용하여 구성 정보를 각 노드에 분배한다. 구성 정보의 예는, 각 노드가 노드들 간의 하나 이상의 유니캐스트 메시지를 통해 다른 임의의 노드와 통신할 수 있도록, 각 노드에 대한 어드레스, 각 노드에 대한 경로설정 테이블 등을 포함한다. 블록(612)에서, 패브릭(122)의 노드들은 소프트웨어 서비스를 실행하고, 이 실행은, 블록(610)에서 분배된 어드레스, 경로설정 테이블, 및 기타 구성 정보에 기초하여 노드들 간에 메시지를 송신하는 것을 포함한다.

[0048] 도 7은 일부 실시예들에 따라 패브릭 상호접속부를 통해 접속된 연산 노드들(602 내지 613)을 포함하는 패브릭(700)을 도시한다. 패브릭(700)은 전술한 패브릭(122)과 유사한 방식으로 동작하지만, 설명을 위해 2차원 토폴로지로 배열된다. 패브릭(700)에 대해 본 명세서에서 설명하는 원리 및 기술은 3D 토폴로지(예를 들어, 3D 토러스)를 갖는 패브릭에도 적용된다는 것을 인식할 것이다. 도 7의 예시에서, 패브릭(700)은 노드(702)에 접속된 단일 MU(701)를 포함한다. MU(701)는 노드들(702 내지 713)의 각자의 READY 상태로의 천이를 개시함으로써 소프트웨어 서비스의 실행을 위한 패브릭(700)의 구성을 제어한다. 또한, 일단 노드들(702 내지 713)이 각각 준비 상태가 되면, MU는 패브릭(700) 전체에 걸쳐 구성 메시지들의 전파를 개시함으로써 각 노드를 구성한다. 이러한 기술들은 도 8 내지 도 10의 예들을 참조하면 더 잘 이해될 수 있다.

[0049] 도 8은 일부 실시예들에 따라 노드들(702 내지 713)이 IDLE 상태에서 READY 상태로 천이하는 예시적인 시퀀스를 도시한다. 이러한 천이는 노드들(702 내지 713)이 패브릭(700)을 통한 구성 메시지들의 전파를 제어하는 스패닝 트리로 자기 조직되게 한다. 도 8은 구성 사이클들(801 내지 806)의 시퀀스를 묘사함으로써 천이를 도시한다. IDLE 상태의 노드들은 교차 음영 없이 원으로서 표시된 한편, READY 상태의 노드들은 교차 음영으로서 표시된다.

[0050] 구성 사이클(801) 전에, 패브릭(700)은 MU(701)의 초기화 및 노드들(702 내지 713)에서의 구성 클록들의 동기화를 야기하는 시스템 리셋을 겪었다. 구성 사이클(801)에서, MU(701)는 스패닝 트리 상태 필드를 READY 상태로 천이시키도록(예를 들어, 노드(702)의 지정된 레지스터에 기입함으로써) 노드(702)에 커맨드를 발행한다. 이에 따라, 구성 사이클(802)에서, 노드(702)는 자신의 스패닝 트리 상태 필드를 준비 상태로 천이시키고, 따라서 노드들(703, 704, 706)에 TOPO 토큰을 발행한다. 구성 사이클(803)에서, 노드들(703, 704, 706) 각각은 각자의 스패닝 트리 상태 필드를 IDLE 상태에서 READY 상태로 천이시켰으며, 따라서, 그들의 접속된 노드들(노드들(705, 707, 609))에 TOPO 토큰을 발행한다. 이에 따라, 구성 사이클(804)에서, 노드들(705, 707, 708)은 IDLE 상태에서 READY 상태로 천이하였고, 그들의 접속된 노드들(노드들(708, 710, 712))에 TOPO 토큰을 발행한다. TOPO 토큰에 응답하여, 노드들(708, 710, 712)은 구성 사이클(805)에 의해 자신들의 스패닝 트리 상태 필드를 READY 상태로 천이시키고, 따라서 그들의 접속된 노드들(711, 713)에 TOPO 토큰을 발행한다. 이에 응답하여, 노드들(711, 713)은 구성 사이클(806)에서 READY 상태로 천이한다. 따라서, 도 8의 도시된 예에서, 노드들(702 내지 713)은, MU(401)가 노드들 중 하나의 노드(즉, 노드(702))에 커맨드를 발행하는 것에 기초하여 자신들의 스패닝 트리 상태 필드들을 모두 IDLE 상태에서 READY 상태로 천이한다. 이는 개별 메시지들을 대역외 네트워크를 통해 각 노드에 송신함으로써 구성을 위해 MU(401)가 각 노드를 개별적으로 준비해야 하는 종래의 시스템에 비해 MU(401)에서의 오버헤드를 감소시킨다.

[0051] 도 5와 관련하여 전술한 바와 유사한 방식으로, 노드들(702 내지 713)은, READY 상태로 천이함에 따라, 자신들의 접속된 노드들 중 하나 이상에 대하여 스패닝 트리에서 자신들의 위치를 식별함으로써 스패닝 트리로 자기 조직된다. 예시적인 스패닝 트리(900)가 도 9에 도시되어 있다. 스패닝 트리(900)는 도 8의 예시적인 시퀀스에 의해 형성된 스패닝 트리를 나타내지 않을 수도 있지만, 대신에, 노드들(702 내지 713) 중 다른 노드들이 형성되는 스패닝 트리에서 상이한 수의 원위 노드들을 가질 수 있음을 나타내도록, 다른 패브릭 토폴로지를 위한 스패닝 트리를 나타낸다는 점에 주목한다. 따라서, 스패닝 트리(900)의 예시에서, 노드(702)는 두 개의 원위 노드인 노드(702, 704)를 갖는 한편, 노드(706)는 단일 원위 노드(710)를 갖고, 노드(707)는 3개의 원위 노드(711, 712, 713)를 갖는다. 그러나, 노드들(702 내지 713)의 각각은 대응하는 단일 근위 노드만을 갖는다.

[0052] 구성 메시지들은, MU(701)로부터 노드들(702 내지 713) 중 하나 이상으로 구성 정보를 분배하도록 스패닝 트리(900)의 토폴로지를 따라 전파될 수 있다. 특히, 노드들(602 내지 613)의 각각은 상이한 메시지 타입들의 처리를 관리하기 위해 메시지 타입에 대응하는 구성 필드의 상태를 관리한다. 도 10은 CHAIN 메시지 유형을 사용하

여 구성 정보를 통신하기 위한 예시적인 기술을 도시한다. 체인 메시지를 수신한 것에 응답하여, 노드들(702 내지 713)의 각각은, 각자의 CHAIN 메시지 상태 필드를, 자신의 원위 노드들이 있는 경우 체인 메시지를 자신의 원위 노드들 중 하나에 제공하는 상태에 둔다. 노드가 스패닝 트리(700)에서 하나보다 많은 원위 노드를 갖는다면, 이 노드는, 자신의 원위 노드들의 각각에 고정된 번호를 할당하고, 자신의 CHAIN 메시지 상태를, 제인 메시지가 아직 송신되지 않은 최저 번호의 접속된 노드에 CHAIN 메시지를 송신하는 상태로 천이시킨다. 노드는, 자신의 모든 원위 노드들에 메시지가 송신되었음을 식별하면, 자신의 CHAIN 메시지 상태를, 그 노드가 자신의 근위 노드에 메시지를 제공하는 상태로 천이시킨다. 이에 의해, CHAIN 메시지는 스패닝 트리(700)의 깊이 탐색을 수행하여, 루트 노드(702)로 복귀되는 메시지로 탐색이 완료된다. 거기에서, CHAIN 메시지 또는 그 메시지의 페이로드가 추가 처리를 위해 MU(701)에 제공될 수 있다.

[0053] 도 10은 일부 실시예들에 따라 패브릭 전체에 걸쳐 CHAIN 메시지의 전파의 일부를 나타내는 구성 사이클들의 세트를 도시한다. 구성 사이클(1001)에서, 노드(702)는 MU(601)로부터 체인 메시지를 수신한다. 이에 응답하여, 노드는, 메시지가 노드(703)로서 식별된 최저 번호의 원위 노드에 송신되어야 한다는 것을 식별하고, 따라서 메시지를 노드(703)에 대응하는 출력 포트에 통신한다. 구성 사이클(1002)에서, CHAIN 메시지는 노드(703)에 전달되었고, 노드(703)는 자신의 접속된 원위 노드들(705)을 CHAIN 메시지를 아직 수신하지 않은 최저 번호의 원위 노드로서 식별한다. 따라서, 노드(703)는 체인 메시지를 노드(705)에 제공한다. 구성 사이클(1003)에서, 노드(705)는 노드(708)를 자신의 최저 번호의 원위 노드로서 식별하고 체인 메시지를 노드(708)에 제공한다. 구성 사이클(1004)에서, 노드(708)는, 자신이 원위 노드를 갖지 않음을 식별하고 따라서 체인 메시지를 자신의 근위 노드(705)에 통신한다. 구성 사이클(1005)에서, 노드(705)는 체인 메시지를 아직 수신하지 않은 최저 번호의 원위 노드가 노드(709)임을 식별한다. 이에 따라, 노드(705)는 체인 메시지를 노드(709)에 제공한다. 구성 사이클(1006)에서, 노드(709)는, 자신이 원위 노드를 갖지 않음을 결정하고 따라서 체인 메시지를 근위 노드(705)에 제공한다. 노드(705)는, 체인 메시지들이 자신의 원위 노드들 모두에 제공되었다는 것을 결정하고, 따라서 자신의 근위 노드(703)에 메시지를 제공한다. 체인 메시지는, 체인 메시지 또는 그 메시지의 페이로드를 MU(701)에 제공할 수 있는 루트 노드(702)에 도달할 때까지 스패닝 트리(900)의 토폴로지에 따라 패브릭을 계속 횡단한다.

[0054] 체인 메시지는 노드들(702 내지 713) 중 하나 이상에 구성 정보를 송신 및 수신하도록 구성 노드에 의해 사용될 수 있다. 예를 들어, 일부 실시예들에서, CHAIN 메시지는 어드레스 정보, 경로설정 테이블 정보, 또는 기타 구성 정보 등의 구성 정보의 페이로드를 포함한다. CHAIN 메시지는, 또한, 노드들(702 내지 713) 중 특정 노드를 식별하는 필드를 페이로드 정보의 타겟으로서 포함한다. 일부 실시예들에서, MU는 튜플(x, y, z)을 갖는 타겟 노드의 상대 위치를 메시지에 포함시킴으로써 메시지의 타겟을 식별한다. 각 노드는, 자신의 접속된 노드들 중 하나에 CHAIN 메시지를 전송할 때, 통신 노드에 대한 수신 노드의 상대 위치에 기초하여 튜플의 값을 조정한다. 예를 들어, 수신 노드가 통신 노드에 대하여 "+X" 노드인 경우, 통신 노드는 튜플의 x 값에 대하여 1을 뺄 수 있다. 따라서, 튜플은 목적지에 도달하면 값(0, 0, 0)을 갖는다. 각 노드는, 체인 메시지를 수신하면, 체인 메시지의 노드 식별자를 체크하고, 튜플 값이 (0, 0, 0)이면, 페이로드를, 해당 노드에서 실행되는 BIOS 코드에 따라 추가 처리될 수 있는 곳인, 자신의 구성 레지스터들 중 하나에 저장한다. 일부 실시예들에서, 이 추가 처리는, 타겟 노드가 응답성 페이로드를 스패닝 트리(700)의의 다음 노드에 제공하기 전에 CHAIN 메시지에 저장할 수 있는 응답성 페이로드를 생성한다. 체인 메시지는 결국 루트 노드(702)로 복귀하고 거기로부터 MU(701)로 복귀하기 때문에, 체인 메시지는, MU(701)로부터 타겟 노드로 정보를 통신하고 타겟 노드로부터 MU(701)로 복귀 정보를 통신하기 위한 기술을 제공한다. 또한, 구성 정보의 이러한 통신은, MU(701)가 타겟 노드로의 직접적 경로를 결정하는 것 없이 및 노드들(702 내지 713) 중 어느 것도 타겟 노드의 위치에 의해 정의된 특별한 경로설정 경로를 따라 구성 메시지를 경로설정하는 것 없이 수행된다. 이는 소프트웨어 서비스의 실행 동안 나중에 사용되는 동일한 패브릭 상호접속부를 통해 구성 메시지가 통신될 수 있게 하여, 노드들(702 내지 713) 간에 정의된 경로설정 경로를 따라 메시지를 통신할 수 있게 한다.

[0055] 일례를 통해 예시하자면, MU(701)는, 구성 정보를 노드(708)에 통신하기를 원하면, 구성 사이클(1001)에서 구성 정보를 갖는 CHAIN 메시지를 노드(702)에 제공한다. 구성 사이클(1004)에 의해, CHAIN 메시지가 노드(708)에 도달하였다. 이에 응답하여, 노드(708)는, 자신이 CHAIN 메시지에 대한 타겟 노드임을 식별하고, 따라서 자신의 구성 레지스터들 하나 이상에 체인 메시지에 대한 페이로드 정보를 저장하고, 임의의 응답 정보를 생성하고, 응답 정보를 CHAIN 메시지의 페이로드에 저장한다. 노드(708)는, 구성 사이클(1005)에서, 변형된 체인 메시지를 원위 노드(705)에 제공한다. CHAIN 메시지는, CHAIN 메시지가(노드(708)로부터의 임의의 응답 정보를 포함하는) 메시지의 페이로드를 MU(701)에 제공하는 노드(702)로 복귀할 때까지 후속 구성 사이클들에 걸쳐 스패닝 트리를 계속 횡단한다.

[0056] 도 11은, 일부 실시예들에 따라 WAVE 메시지가 스패닝 트리(900)를 통해 어떻게 전파되는지를 나타내는 구성 사이클들의 시퀀스를 도시한다. 각 노드는, WAVE 유형 메시지의 수신에 응답하여, 자신의 WAVE 메시지 상태 필드를, 노드가 자신의 원위 노드들에 웨이브 메시지를 이전에 제공했는지 여부를 식별하는 상태로 두고, 제공하지 않았다면, 노드가 자신의 모든 원위 노드들에 메시지를 제공하는 WAVE 상태 메시지 필드로 천이한다. 메시지가 자신의 원위 노드들에 이전에 제공되었다면, 노드는, 자신의 WAVE 상태 메시지 필드를, 자신의 원위 노드들 모두로부터의 WAVE 메시지에 대한 응답을 기다리는 상태로 천이시킨다. 노드는, 일단 모든 원위 노드들로부터 WAVE 메시지에 대한 응답을 수신하였다면, 자신의 WAVE 메시지 상태 필드를, 노드가 자신의 근위 노드들에 웨이브 메시지를 제공하는 상태로 천이시킨다. 따라서, 도 11의 예에서, 구성 사이클(1101)에서, 노드(702)는 MU(701)로부터 웨이브 메시지를 수신한다. 이에 응답하여, 노드(702)는 자신의 원위 노드들(703, 704) 모두에 WAVE 메시지를 제공한다. 구성 사이클(1102)에서, 노드들(703 및 704) 각각은, 웨이브 메시지가 자신들의 원위 노드들에 이전에 제공되지 않았다고 결정한다. 이에 따라, 노드(703)는 자신의 원위 노드(705 및 706)에 WAVE 메시지를 제공하고, 노드(704)는 자신의 원위 노드(707)에 WAVE 메시지를 제공한다. 유사한 방식으로, 구성 사이클(1103)에서, 노드(705)는 자신의 원위 노드들(708, 709)에 WAVE 메시지를 제공하고, 노드(706)는 자신의 원위 노드(710)에 WAVE 메시지를 제공하고, 노드(707)는 자신의 원위 노드들(711, 712. 713)에 웨이브 메시지를 제공한다.

[0057] 구성 사이클(1104)에서, 노드들(708 내지 713)의 각각은 스패닝 트리(800)에서 원위 노드들을 갖지 않음을 식별한다. 이에 따라, 노드들(708 내지 713)의 각각은 자신의 대응하는 근위 노드에 WAVE 메시지를 제공한다. 예를 들어, 노드(709)는 자신의 근위 노드(705)에 WAVE 메시지를 제공한다. 구성 사이클(1105, 1106)에서, WAVE 메시지는 구성 사이클(1106)에서 MU(701)로 복귀할 때까지 스패닝 트리(900)를 따라 원위측으로 계속 진행한다.

[0058] 도 12는 일부 실시예들에 따른 서버(100)의 노드들의 예시적인 물리적 배열을 도시한다. 도시된 예에서, 패브릭 상호접속부(112)(도 1)는 플러그인 소켓들(1204)의 하나 이상의 행 또는 다른 집합(aggregation)을 갖는 하나 이상의 상호접속부(1202)를 포함한다. 상호접속부(1202)는 백플레인, 인쇄 배선 기판, 마더보드, 케이블링, 또는 다른 가요성 배선, 또는 이들의 조합 등의 고정된 또는 가요성 상호접속부를 포함할 수 있다. 또한, 상호접속부(1202)는 전기 신호, 광 신호, 또는 이들의 조합을 구현할 수 있다. 각 플러그인 소켓(1204)은 FRU(1206 내지 1211) 등의 하나 이상의 FRU를 상호접속부(1202)와 접속하도록 동작하는 카드-에지 소켓을 포함한다. 각 FRU는 서버(100)의 대응 노드를 나타낸다. 예를 들어, FRU(1206 내지 1209)는 연산 노드들을 포함할 수도 있고, FRU(1210)는 네트워크 노드를 포함할 수도 있고, FRU(1211)는 저장 노드를 포함할 수 있다. 하나 이상의 FRU(1206 내지 1211)는 또한 대응하는 관리 유닛을 포함할 수도 있다.

[0059] 각 FRU는 PCB 상에 배치된 구성요소들을 포함하고, 이에 의해 구성 요소들은 PCB의 금속 층들을 통해 상호접속되며, FRU에 의해 표현되는 노드의 기능을 제공한다. 예를 들어, 이 예에서 연산 노드인 FRU(1206)는, 하나 이상의 프로세서 코어(1122)를 포함하는 프로세서(1220), DRAM 듀얼 인라인 메모리 모듈(DIMM) 등의 하나 이상의 메모리 모듈(1124), 및 패브릭 인터페이스 디바이스(1126)를 구현하는 PCB(1212)를 포함한다. 각 FRU는, 플러그인 소켓(1204)을 통해 FRU를 상호접속부(1202)에 접속하도록 동작하는 소켓 인터페이스(1240)를 더 포함한다.

[0060] 상호접속부(1202)는, 상호접속부(1202)가 FRU를 링들에 접속하고 링들을 도 3의 토러스 네트워크(300) 등의 2D 또는 3D 토러스 네트워크 토폴로지에 접속하게끔 동작하도록, 플러그인 소켓들(1204) 간에 데이터 통신 경로의 이점을 가진다. FRU는 FRU(1206)의 패브릭 인터페이스 디바이스(1226) 등의 자신의 대응하는 패브릭 인터페이스를 통해 이러한 데이터 통신 경로를 이용한다. 소켓 인터페이스(1230)는, X차원 링(예를 들어, 핀(0, 1)에 대한 링-X_IN 포트(1232) 및 핀(2, 3)에 대한 링-X_OUT 포트(1234)), Y차원 링(예를 들어, 핀(4, 5)에 대한 링-Y_IN 포트(1136) 및 핀(6, 7)에 대한 링-Y_OUT 포트(1238)) 및 Z차원 링(예를 들어, 핀(8, 9)에 대한 링-Z_IN 포트(1240) 및 핀(10, 11)에 대한 링-Z_OUT 포트(1242))을 위한 포트 인터페이스로서 기능하도록 플러그인 소켓(1204)의 대응하는 전기 접촉부들에 전기적으로 접속하는 전기 접촉부들(예를 들어, 카드 에지 핀들)을 제공한다. 도시한 예에서, 각 포트는 PCIE 레인(lane)의 입력 포트 또는 출력 포트 중 하나를 포함하는 차동 송신기이다. 통상의 기술자는, 추가 레인 또는 추가 포트를 수용하도록 포트가 추가 TX/RX 신호 핀을 포함할 수 있음을 이해할 것이다.

[0061] 도 13은 일부 실시예들에 따라 도 1의 서버(100)에서 구현된 연산 노드(1300)를 도시한다. 연산 노드(1300)는 도 1의 연산 노드들 101-106) 중 하나에 대응한다. 도시한 예에서, 연산 노드(1300)는, (도 12의 프로세서(1320), 하나 이상의 메모리 모듈(1224), 및 패브릭 인터페이스 디바이스(1226)를 각각 나타내는) 프로세서(1302), 시스템 메모리(1304), 및 패브릭 인터페이스 디바이스(1306)를 포함한다. 프로세서(1302)는 하나 이상의 프로세서 코어(1308) 및 노스브리지(1210)를 포함한다. 하나 이상의 프로세서 코어(1308)는, 중앙 처리 유닛

(CPU) 코어, 그래픽 처리 유닛(GPU) 코어, 디지털 신호 처리 유닛(DSP) 코어 등의 다양한 유형의 프로세서 코어들 또는 이들의 조합 중 임의의 것을 포함할 수 있고, x86 명령어 세트 아키텍처 또는 어드밴스드 RISC 머신(ARM) 아키텍처 등의 다양한 명령어 세트 아키텍처들 중 임의의 것을 구현할 수도 있다. 시스템 메모리(1204)는 DRAM 모듈, SRAM 모듈, 플래시 메모리, 또는 이들의 조합 등의 하나 이상의 메모리 모듈을 포함할 수 있다. 노스브리지(1310)는 하나 이상의 코어(1308), 시스템 메모리(1304), 및 패브릭 인터페이스 디바이스(1306)를 상호접속한다. 패브릭 인터페이스 디바이스(1306)는, 일부 실시예들에서, 주문형 집적 회로(ASIC), 필드 프로그래머블 게이트 어레이(FPGA), 마스크 프로그래머블 게이트 어레이, 프로그래머블 로직 등의 집적 회로 디바이스에서 구현된다.

[0062] 종래의 연산 시스템에서, 노스브리지(1310)는, 노스브리지(1310)(이에 따라 프로세서 코어(1308))와 로컬 주변 자원들을 관리하는 하나 이상의 로컬 I/O 제어기 간의 인터페이스로서 동작할 사우스브리지에 접속된다. 그러나, 전술한 바와 같이, 일부 실시예들에서, 연산 노드(1300)는 로컬 주변 자원들 또는 그들의 I/O 제어기들을 유지하지 않고, 대신에 서버(100) 내의 다른 노드들에서 공유 원격 주변 자원들을 사용한다. 이러한 구성이 프로세서(1302)에서 실행되는 소프트웨어에 대하여 투명해지도록, 패브릭 인터페이스 디바이스(1306)는, 패브릭 인터페이스 디바이스(1306)의 하드웨어가 사우스브리지를 에뮬레이트(emulate)하고 이에 따라 노스브리지(1310)에 대하여 로컬 주변 자원들에 접속된 로컬 사우스브리지로서 보이도록, 연산 노드에 할당된 원격 주변 자원들을 가상화한다.

[0063] 이를 위해, 패브릭 인터페이스 디바이스(1306)는, I/O 버스 인터페이스(1312), 가상 네트워크 제어기(1314), 가상 저장 제어기(1316), 패킷 포맷터(1318), 및 패브릭 스위치(1320)를 포함하는 NIC(1319)를 포함한다. I/O 버스 인터페이스(1312)는, 로컬 I/O 버스(1324)를 통해 노스브리지(1310)에 접속되며, 로컬 I/O 버스(1324) 상에 있는 것으로 보이는 가상화된 주변 자원들에 어드레싱된 요구를 인터셉트하고 I/O 버스 인터페이스(1312)에 의해 가상 표현되는 주변 자원들의 원격 위치로 인해 잠재적으로 더욱 긴 지연이 있는 로컬 주변 자원과 동일한 방식으로 그 요구에 응답함으로써 각 로컬 프로세서 코어(1208)에 대한 가상 엔드포인트로서 기능한다.

[0064] I/O 버스 인터페이스(1312)가 노스브리지(1310)에 물리적 인터페이스를 제공하는 동안, 보다 높은 레벨의 응답이 가상 네트워크 제어기(1314) 및 가상 저장 제어기(1316)에 의해 생성된다. 데이터 센터 네트워크(114)(도 1)에 접속된 이더넷 NIC 등의 외부 네트워크에 접속된 네트워크 주변장치에 대한 I/O 버스(1324)를 통해 송신된 요구는, I/O 버스 인터페이스(1312)에 의해 가상 네트워크 제어기(1314)에 경로 설정되는 한편, 저장 요구는 I/O 버스 인터페이스(1312)에 의해 가상 저장 제어기(1316)로 경로설정된다. 가상 네트워크 제어기(1314)는 예를 들어 이더넷 프로토콜에 기초하여 착신 및 발신 요구들의 처리를 제공한다. 가상 저장 제어기는, 예를 들어, 직렬 ATA(SATA) 프로토콜, 직렬 부착 SCSI(SAS) 프로토콜, 범용 직렬 버스(USB) 프로토콜 등에 기초하여 착신 및 발신 요구들의 처리를 제공한다.

[0065] 도 1 내지 도 11에 관련하여 전술한 바와 같이 연산 노드들의 각각의 구성 후에, 연산 노드(1300)는 요구를 생성하는 소프트웨어 서비스를 실행한다. 요구는, 가상 네트워크 제어기(1314) 또는 가상 저장 제어기(1316)에 의해 처리된 후에, 요구를 하나 이상의 패킷으로 캡슐화하는 패킷 포맷터(1318)에 포워딩된다. 이어서, 패킷 포맷터(1318)는 요구를 위한 물리적 주변 자원을 관리하는 I/O 노드의 패브릭 어드레스 또는 다른 위치 식별자를 결정한다. 패킷 포맷터(1318)는, 요구가 캡슐화된 하나 이상의 패킷의 헤더에 식별된 패브릭 어드레스(여기서는 "패브릭 ID"라고 함)를 부가하고, 그 패킷을 전송을 위해 NIC(1319)의 패브릭 스위치(1320)에 제공한다.

[0066] 도시한 바와 같이, 패브릭 스위치(1320)는 복수의 포트를 구현하고, 각 포트는 패브릭 상호접속부(112)의 상이한 링크와 인터페이싱한다. 도 2의 3x3 토러스 네트워크(200)를 사용하여 설명하기 위해. 연산 노드(1300)가 (1,1,1)에 있는 노드를 나타내는 것이라고 가정한다. 이 예에서, 패브릭 스위치(1320)는, 자신을 7개의 양방향 링크에 연결하기 위한 적어도 7개의 포트를 갖고, 이러한 양방향 링크는, 패킷 포맷터(1318)에 대한 내부 링크; (0,1,1)에 있는 노드에 대한 외부 링크; (1,0,1)에 있는 노드에 대한 외부 링크; (1,1,0)에 있는 노드에 대한 외부 링크; (1,2,1)에 있는 노드에 대한 외부 링크; (2,1,1)에 있는 노드에 대한 외부 링크; 및 (1,1,2)에 있는 노드에 대한 외부 링크이다. 연산 노드(1200)의 구성 후에, 패브릭 스위치(1320)의 포트들 간의 데이터의 스위칭 제어는, 패킷에 의해 표시되는 목적지 어드레스(즉, 목적지 패브릭 ID)에 기초하여 출구 포트를 특정하는 집적된 결정론적 스위칭 로직 및 서버(100)에서 구현되는 결정론적 경로설정에 기초하여 결정된다. 각 연산 노드의 목적지 패브릭 ID는 구성 중에 각 노드에 분배될 수 있다. 예를 들어, MU는 전술한 바와 같이 일련의 CHAIN 메시지를 통해 자신의 목적지 패브릭 ID를 각 연산 노드에 분배할 수 있다.

[0067] MU에 의한 구성 후에 그리고 소프트웨어 서비스의 정상 실행 동안, 연산 노드(1300)는 수신된 패킷들을 다음과

같이 처리한다. 패킷의 목적지가 연산 노드(1300)인 다른 노드로부터 수신된 패킷들에 대해, 패브릭 스위치(1320)는 결정론적 경로설정 로직에 기초하여 패킷 포맷터(1318)에 접속된 포트에 인입 패킷을 경로설정한다. 이어서, 패킷 포맷터(1318)는, 패킷으로부터 응답/요구를 캡슐해제하고, 이를 요구에 포함된 유형 식별자에 기초하여 가상 네트워크 제어기(1314) 또는 가상 저장 제어기(1316)에 제공한다. 이어서, 요구를 수신하는 제어기는, 응답/요구를 처리하고 I/O 버스 인터페이스(1312)를 제어하여 그 요구를 노스브리지(1310)에 시그널링하며, 이때, 응답/요구는, 로컬 주변 자원으로부터의 응답 또는 요구인 것처럼 처리된다.

[0068] 연산 노드(1300)가 패킷에 대한 경로설정 경로의 중간 노드인 일시적 유니캐스트 패킷의 경우, 패브릭 스위치(1320)는, 일시적 패킷의 헤더로부터 목적지 어드레스(예를 들어, 튜플(x, y, z))를 결정하고, 결정론적 경로설정 로직에 의해 식별된 대응하는 출력 포트에 패킷을 제공한다. 일부 실시예들에서, 패브릭 스위치(1320)는 국부적으로 저장된 경로설정 테이블을 사용하여 목적지 어드레스를 결정한다. 구성 중에, MU는 전술한 바와 같이 CHAIN 메시지 또는 WAVE 메시지를 사용하여 경로설정 테이블을 각 작업 노드에 분배할 수 있다.

[0069] 전술한 바와 같이, 동작 노드(1300)를 구성하기 위한 BIOS의 일부는 마찬가지로 가상화된 주변 자원일 수 있다. 이러한 경우에, 패브릭 인터페이스 디바이스(1306)는, 로컬 I/O 버스(1224)를 통해 또는 별도의 로우 핀 카운트(LPC) 버스(1328)를 통해 노스브리지(1310)에 접속된 BIOS 제어기(1326)를 포함할 수 있다. 저장 및 네트워크 자원들과 마찬가지로, BIOS 제어기(1326)는, 패킷 포맷터(1318) 및 패브릭 스위치(1320)를 통해 BIOS 요구를 원격 BIOS를 관리하는 I/O 노드에 포워딩한 후 공급된 BIOS 데이터를 노스브리지(1310)에 제공하여 노스브리지(1310)로부터의 BIOS 요구에 응답함으로써, 로컬 BIOS를 에뮬레이트할 수 있다.

[0070] 도 13의 예시에서, 패브릭 인터페이스 디바이스(1306)는 구성 상태 레지스터(170)를 포함한다. 서버(100)의 구성 동안, 패브릭 인터페이스 디바이스(1306)는, 도 1 내지 도 12와 관련하여 상술한 바와 같이, 연산 노드(1300)에 접속된 노드들에서의 구성 상태 레지스터들의 대응하는 필드들의 갱신에 응답하여 구성 상태 레지스터(170)의 필드들을 갱신한다. 구성 레지스터(170)에서 각 필드의 상태에 기초하여, 패브릭 인터페이스 디바이스는, 자신의 접속된 노드들에 대한 스패닝 트리에서의 연산 노드(1300)의 위치를 식별하고, 통신된 구성 메시지들을 수신 및 처리하고, 다른 구성 동작을 수행한다.

[0071] 일부 실시예들에서, 전술한 기능 중 적어도 일부는 컴퓨터 판독가능 매체에 유형적으로 저장된 하나 이상의 소프트웨어 프로그램을 실행하는 하나 이상의 프로세서에 의해 구현될 수도 있으며, 이에 의해, 하나 이상의 소프트웨어 프로그램은, 실행시, 하나 이상의 프로세서가 전술한 하나 이상의 기능을 수행하게 하는 명령어들을 포함한다. 일부 실시예들에서, 전술한 장치와 기술들은, 도 1 내지 도 13을 참조하여 전술한 서버(100)의 특정 구성요소들(예를 들어, 패브릭 인터페이스 디바이스 또는 연산 노드) 등의 하나 이상의 집적 회로(IC) 디바이스(집적 회로 패키지 또는 마이크로 칩이라고도 함)를 포함하는 시스템에서 구현된다. 이러한 IC 디바이스들의 설계 및 제조에는, 전자 설계 자동화(EDA) 및 컴퓨터 지원 설계(CAD) 소프트웨어 도구들이 사용될 수도 있다. 이러한 설계 도구들은 통상적으로 하나 이상의 소프트웨어 프로그램으로서 표현된다. 하나 이상의 소프트웨어 프로그램은, 회로를 제조하기 위해 제조 시스템을 설계 또는 적응시키기 위한 프로세스의 적어도 일부를 수행하도록 하나 이상의 IC 디바이스의 회로를 나타내는 코드로 동작하게끔 컴퓨터 시스템을 조작하도록 컴퓨터 시스템에 의해 실행가능한 코드를 포함한다. 이 코드는, 명령어, 데이터, 또는 명령어와 데이터의 조합을 포함할 수 있다. 설계 도구 또는 제조 도구를 나타내는 소프트웨어 명령어는 통상적으로 연산 시스템에 액세스가능한 컴퓨터 판독가능 저장 매체에 저장된다. 유사하게, IC 디바이스의 설계 또는 제조의 하나 이상의 단계를 나타내는 코드는, 동일한 컴퓨터 판독가능 저장 매체 또는 다른 컴퓨터 판독가능 저장 매체에 저장되고 액세스될 수도 있다.

[0072] 컴퓨터 판독가능 저장 매체는, 명령어 및/또는 데이터를 컴퓨터 시스템에 제공하도록 사용 중에 컴퓨터 시스템에 의해 액세스가능한 임의의 저장 매체 또는 저장 매체들의 조합을 포함할 수도 있다. 이러한 저장 매체는, 광 매체(예를 들어, 컴팩트 디스크(CD), 디지털 버서타일 디스크(DVD), 블루-레이 디스크), 자기 매체(예를 들어, 플로피 디스크, 자기 테이프, 또는 자기 하드 드라이브), 휘발성 메모리(예를 들어, RAM 또는 캐시), 비휘발성 메모리(예를 들어, ROM 또는 플래시 메모리), 또는 미세전자기계 시스템(MEMS) 기반 저장 매체를 포함할 수 있지만, 이에 한정되지 않는다. 컴퓨터 판독가능 저장 매체는, 연산 시스템(예를 들어, 시스템 RAM 또는 ROM)에 내장될 수도 있고, 연산 시스템(예를 들어, 자기 하드 드라이브)에 고정적으로 부착될 수도 있고, 연산 시스템(예를 들어, 광학 디스크 또는 유니버셜 직렬 버스(USB) 기반 플래시 메모리)에 탈착가능하게 부착될 수도 있고, 또는 유선 또는 무선 네트워크(예를 들어, 네트워크 액세스가능 스토리지(NAS))를 통해 컴퓨터 시스템에 연결될 수도 있다.

[0073] 도 14는 하나 이상의 양태들을 구현하는 IC 디바이스의 설계 및 제조를 위한 예시적인 방법(1400)을 도시하는 흐름도이다. 상술한 바와 같이, 다음에 따르는 공정들의 각각에 대해 생성된 코드는, 대응하는 설계 도구 또는 제조 도구에 의한 액세스 및 사용을 위해 컴퓨터 판독가능 저장 매체에 저장되거나 구체화된다.

[0074] 블록(1402)에서는, IC 디바이스의 기능 명세를 생성한다. (종종 마이크로 아키텍처 명세(MAS)이라고도 하는) 기능 명세는, C, C++, 시스템C(SystemC), 시뮬링크(Simulink)(상표명), 또는 MATLAB(상표명)을 포함한 다양한 프로그램 언어들이나 모델링 언어들 중 임의의 것에 의해 표현될 수도 있다.

[0075] 블록(1404)에서, 기능 명세는 IC 디바이스의 하드웨어를 나타내는 하드웨어 설명 코드를 생성하는 데 사용된다. 일부 실시예들에서, 하드웨어 설명 코드는, IC 디바이스의 회로들의 형식적인 기술 및 설계를 위한 다양한 컴퓨터 언어들, 명세 언어들, 또는 모델링 언어들 중 임의의 것을 포함하는 적어도 하나의 하드웨어 기술 언어(HDL)를 사용하여 표현된다. 생성된 HDL 코드는, 통상적으로 IC 디바이스의 회로들의 동작, 회로들의 설계 및 조직, 및 시뮬레이션을 통한 IC 디바이스의 정확한 동작을 검증하는 테스트를 나타낸다. HDL의 예로는 AHDL(아날로그(Analog) HDL), 베릴로그(Verilog) HDL, 시스템베릴로그(SystemVerilog) HDL, 및 VHDL이 있다. 동기화된 디지털 회로들을 구현하는 IC 디바이스들의 경우, 하드웨어 설명자 코드는 동기화 디지털 회로들의 동작을 추상적으로 표현하는 레지스터 전송 레벨(RTL) 코드를 포함할 수도 있다. 다른 유형의 회로의 경우, 하드웨어 설명자 코드는 회로 동작을 추상적으로 표현하는 거동 레벨 코드를 포함할 수도 있다. 하드웨어 설명 코드에 의해 표현된 HDL 모델은 통상적으로 설계 검증을 통과하기 위해 시뮬레이션과 디버깅의 하나 이상의 라운드를 거친다.

[0076] 하드웨어 설명 코드에 의해 표현된 설계를 검증한 후에, 블록(1406)에서, 합성 도구를 사용하여 하드웨어 설명 코드를 합성하여 IC 디바이스의 회로의 초기 물리적 구현을 나타내거나 정의하는 코드를 생성한다. 일부 실시예들에서, 합성 도구는, 회로 디바이스 인스턴스들(예를 들어, 게이트, 트랜지스터, 저항기, 커패시터, 인덕터, 다이오드 등) 및 회로 디바이스 인스턴스들 간의 네트들 또는 접속부들을 포함하는 하나 이상의 네트리스트(netlist)를 생성한다. 대안으로, 네트리스트의 전부 또는 일부는 합성 도구를 사용하지 않고 수동으로 생성될 수 있다. 하드웨어 설명 코드와 마찬가지로, 네트리스트는, 하나 이상의 네트리스트들의 최종 세트가 생성되기 전에 하나 이상의 테스트 및 검증 프로세스를 거칠 수도 있다.

[0077] 대안으로, 개략적 편집기 도구를 사용하여 IC 디바이스의 회로의 개략도를 그릴 수 있고, 이어서 개략적 캡처 도구를 사용하여 결과 회로도를 캡처하고 회로도의 접속성과 구성요소들을 나타내는 (컴퓨터 판독가능 매체에 저장된) 하나 이상의 네트리스트를 생성할 수도 있다. 이어서, 캡처된 회로도는 테스트 및 검증을 위해 시뮬레이션의 하나 이상의 라운드를 거칠 수 있다.

[0078] 블록(1408)에서, 하나 이상의 EDA 도구는 블록(1406)에서 생성된 네트리스트를 사용하여 IC 디바이스의 회로의 물리적 레이아웃을 나타내는 코드를 생성한다. 이 프로세스는, 예를 들어, IC 디바이스의 회로의 각 요소의 위치를 결정하거나 고정하기 위해 네트리스트를 사용하는 배치 도구를 포함할 수 있다. 또한, 경로설정 도구는 배치 프로세스를 기반으로 하여 네트리스트(들)에 따라 회로 요소들을 접속하는 데 필요한 와이어들을 추가하고 경로설정한다. 그 결과 코드는 IC 디바이스의 3차원 모델을 나타낸다. 코드는 예를 들어 GDSII(그래픽 데이터베이스 시스템 II) 포맷 등의 데이터베이스 파일 포맷으로 표현될 수도 있다. 이러한 포맷의 데이터는 통상적으로 회로 레이아웃에 대한 기하학적 형상, 텍스트 레이블, 및 기타 정보를 계층적 형태로 나타낸다.

[0079] 블록(1410)에서, 물리적 레이아웃 코드(예를 들어, GDSII 코드)는, (예를 들어, 마스크 작업을 통해) IC 디바이스를 제조하는 제조 설비의 제조 도구들을 구성하거나 그 외에는 적응시키기 위한 물리적 레이아웃 코드를 사용하는 제조 설비에 제공된다. 즉, 물리적 레이아웃 코드는, 하나 이상의 컴퓨터 시스템으로 프로그래밍될 수도 있으며, 이러한 컴퓨터 시스템은 제조 설비의 도구들의 동작 또는 그 도구들의 내부에서 수행되는 제조 동작들을 전체적으로 또는 부분적으로 제어할 수도 있다.

[0080] 일반적인 설명에 있어서 전술한 활동들이나 요소들이 모두 필요한 것은 아니며, 특정 활동이나 디바이스의 일부가 필요하지 않을 수도 있으며, 전술한 것에 더하여, 하나 이상의 추가 활동이 수행될 수도 있거나 요소들이 포함될 수도 있다는 점에 주목한다. 또한, 활동들이 열거되는 순서가 반드시 그 활동들이 수행되는 순서는 아니다.

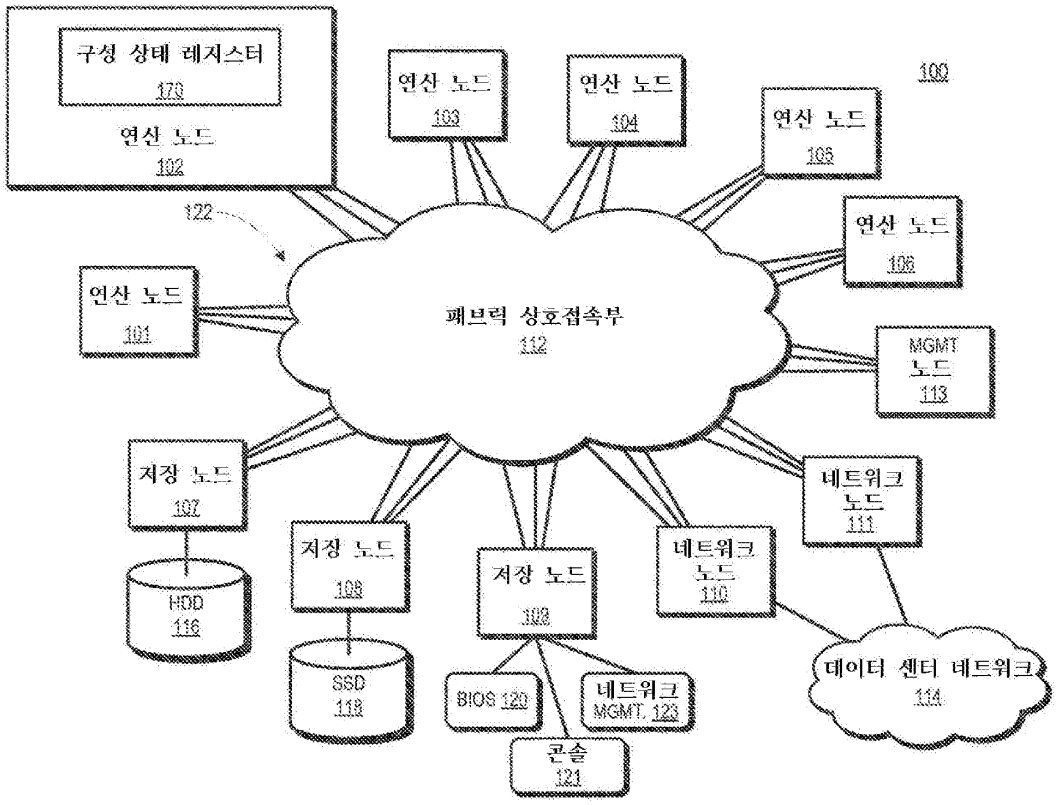
[0081] 또한, 그 개념들은 특정 실시예들을 참조하여 설명되었다. 그러나, 통상의 기술자는, 이하의 청구범위에 설명된 바와 같이 본 개시 내용의 범위를 벗어나지 않으면서 다양한 변형 및 변경이 이루어질 수 있음을 이해할 것이다. 이에 따라, 명세서 및 도면은 제한적인 의미라기보다는 예시적인 것으로 간주되어야 하며, 이러한 모든

수정을 본 개시 내용의 범위 내에 포함하고자 하는 것이다.

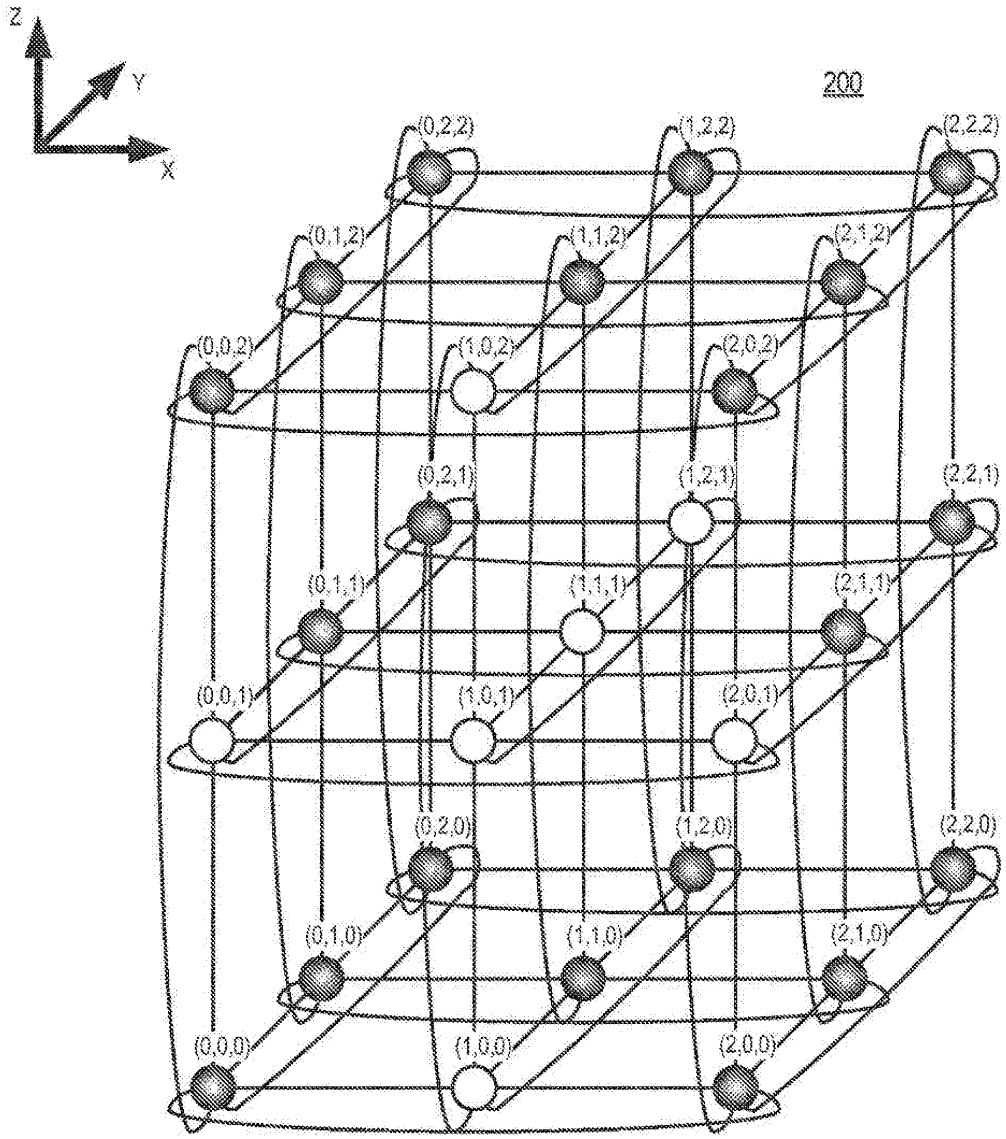
[0082] 이점들, 다른 장점들, 및 문제점들에 대한 해결 방안을 특정 실시예들과 관련하여 설명하였다. 그러나, 그 이점들, 장점들, 문제점들에 대한 해결 방안, 및 그 이점, 장점 또는 해결 방안을 발생시키거나 더욱 두드러지게 할 수도 있는 임의의 특징부(들)를, 모든 청구항들 또는 임의의 청구항의 중요 특징부, 필요 특징부, 또는 필수 특징부로서 해석되어서는 안 된다.

도면

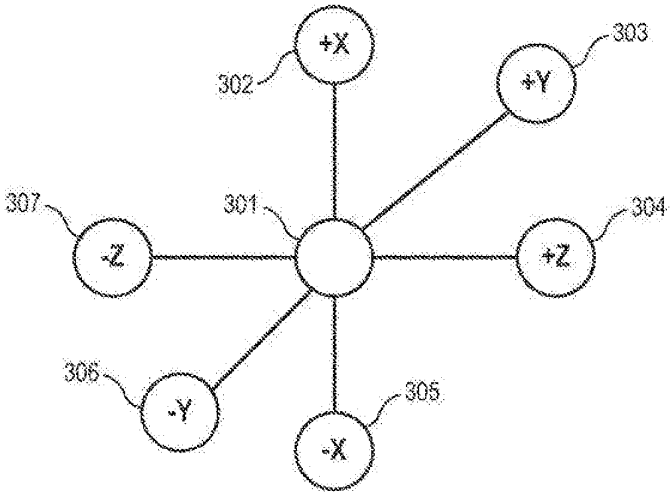
도면1



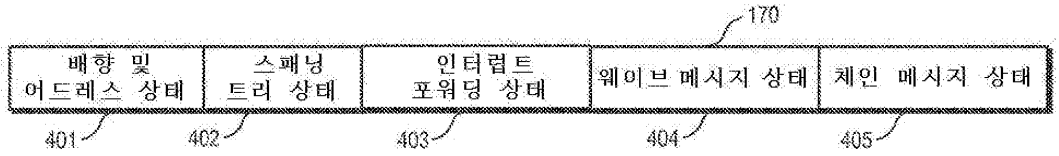
도면2


Z
Y
X
200
(0,2,2)
(1,2,2)
(2,2,2)
(0,1,2)
(1,1,2)
(2,1,2)
(0,0,2)
(1,0,2)
(2,0,2)
(0,2,1)
(1,2,1)
(2,2,1)
(0,1,1)
(1,1,1)
(2,1,1)
(0,0,1)
(1,0,1)
(2,0,1)
(0,2,0)
(1,2,0)
(2,2,0)
(0,1,0)
(1,1,0)
(2,1,0)
(0,0,0)
(1,0,0)
(2,0,0)

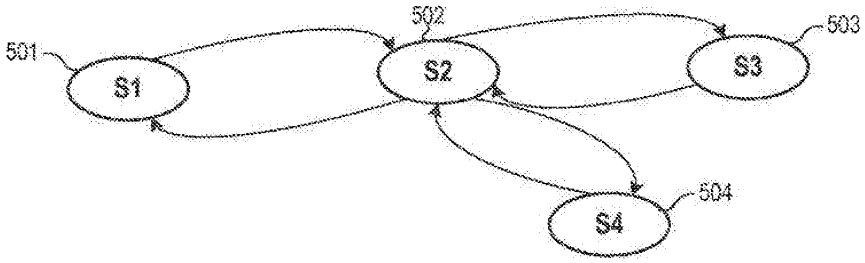
도면3


+X
302
+Y
303
307
301
304
-Z
+Z
306
-Y
305
-X

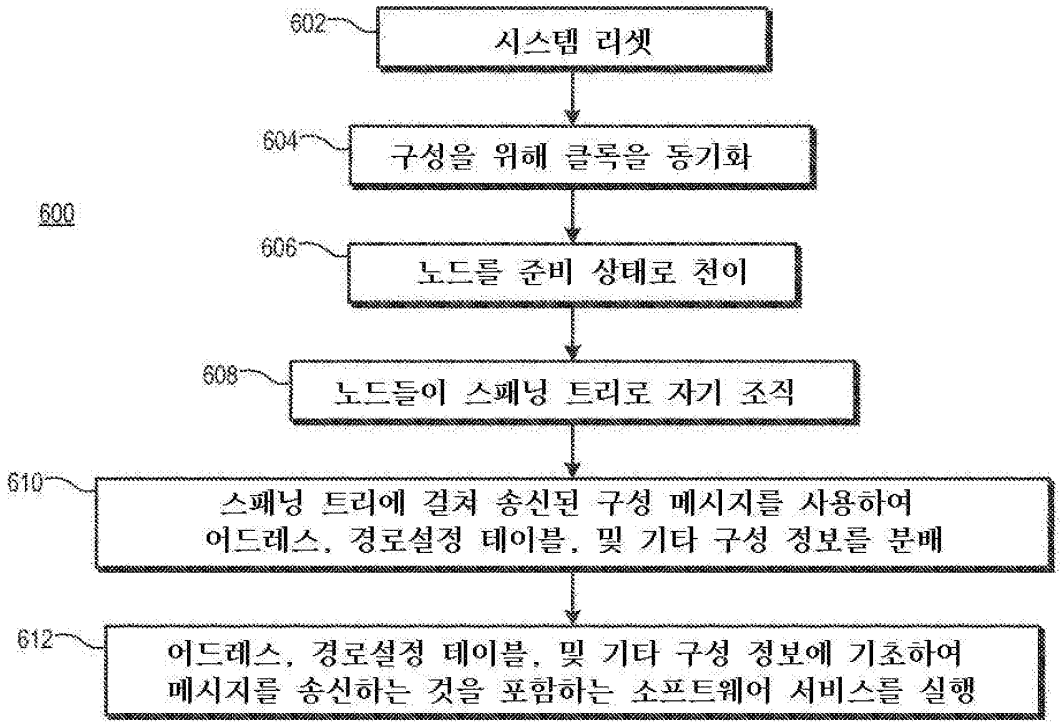
도면4


170
배향 및 어드레스 상태
스패닝 트리 상태
인터럽트 포워딩 상태
웨이브 메시지 상태
체인 메시지 상태
401
402
403
404
405

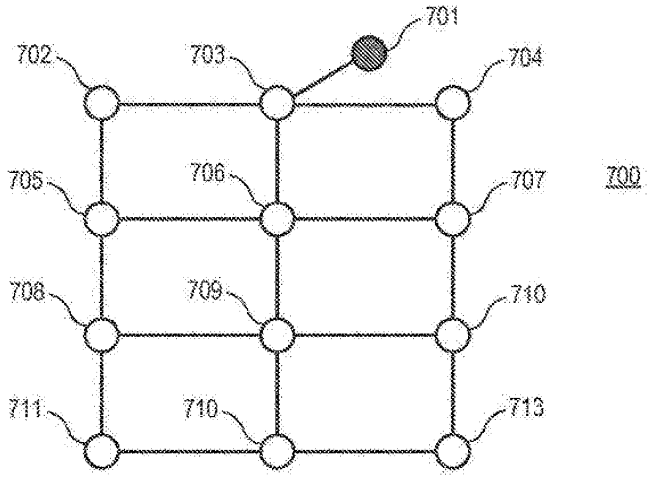
도면5


501
S1
502
S2
503
S3
504
S4

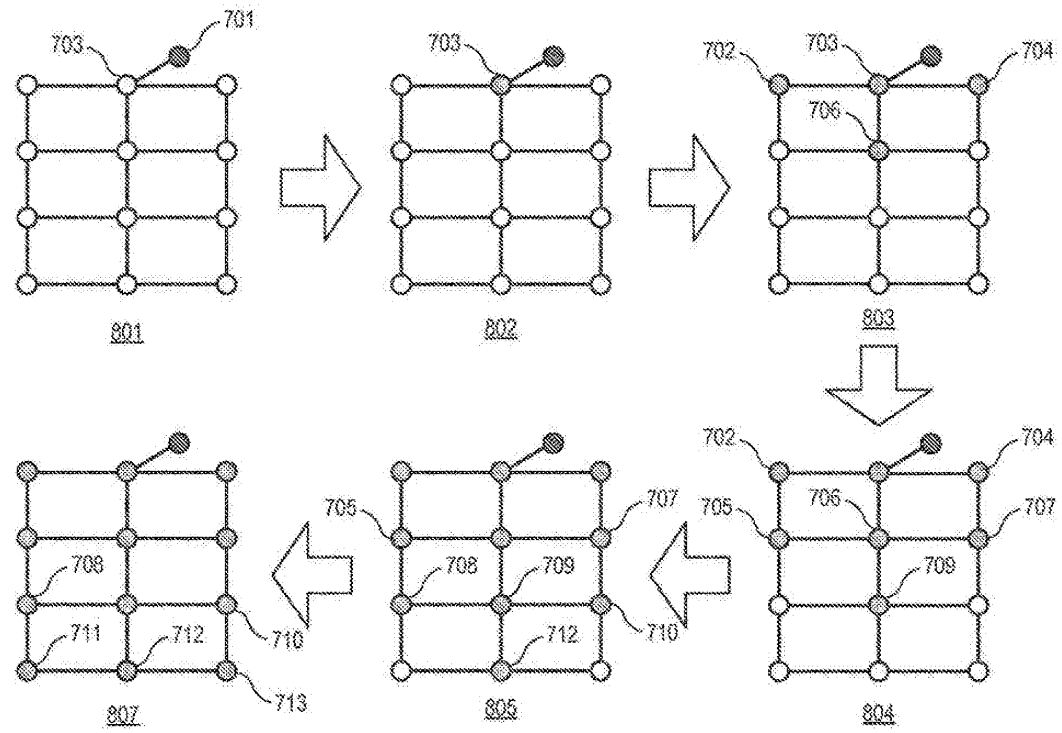
도면6


600
602
시스템 리셋
604
구성을 위해 클록을 동기화
606
노드를 준비 상태로 천이
608
노드들이 스패닝 트리로 자기 조직
610
스패닝 트리에 걸쳐 송신된 구성 메시지를 사용하여
어드레스, 경로설정 데이블, 및 기타 구성 정보를 분배
612
어드레스, 경로설정 데이블, 및 기타 구성 정보에 기초하여
메시지를 송신하는 것을 포함하는 소프트웨어 서비스를 실행

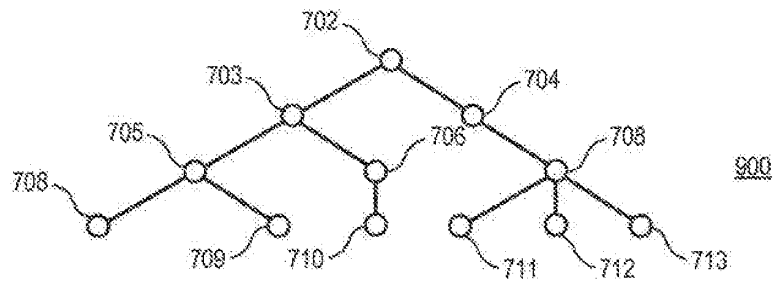
도면7


701
702
703
704
705
706
707
700
708
709
710
711
710
713

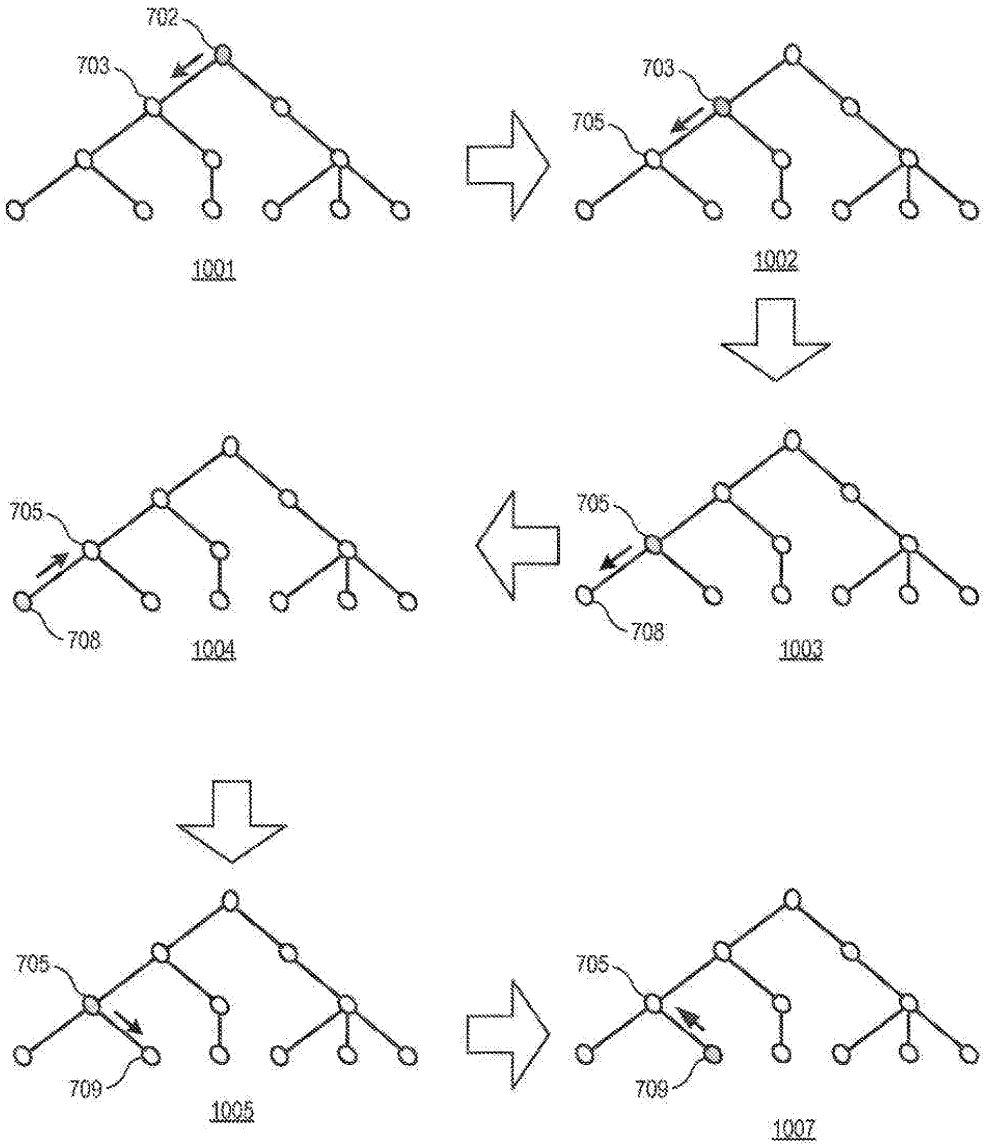
도면8


701
703
801
703
802
702
703
704
706
803
702
704
706
707
705
708
709
709
710
712
804
705
707
708
709
710
712
805
711
712
710
713
807

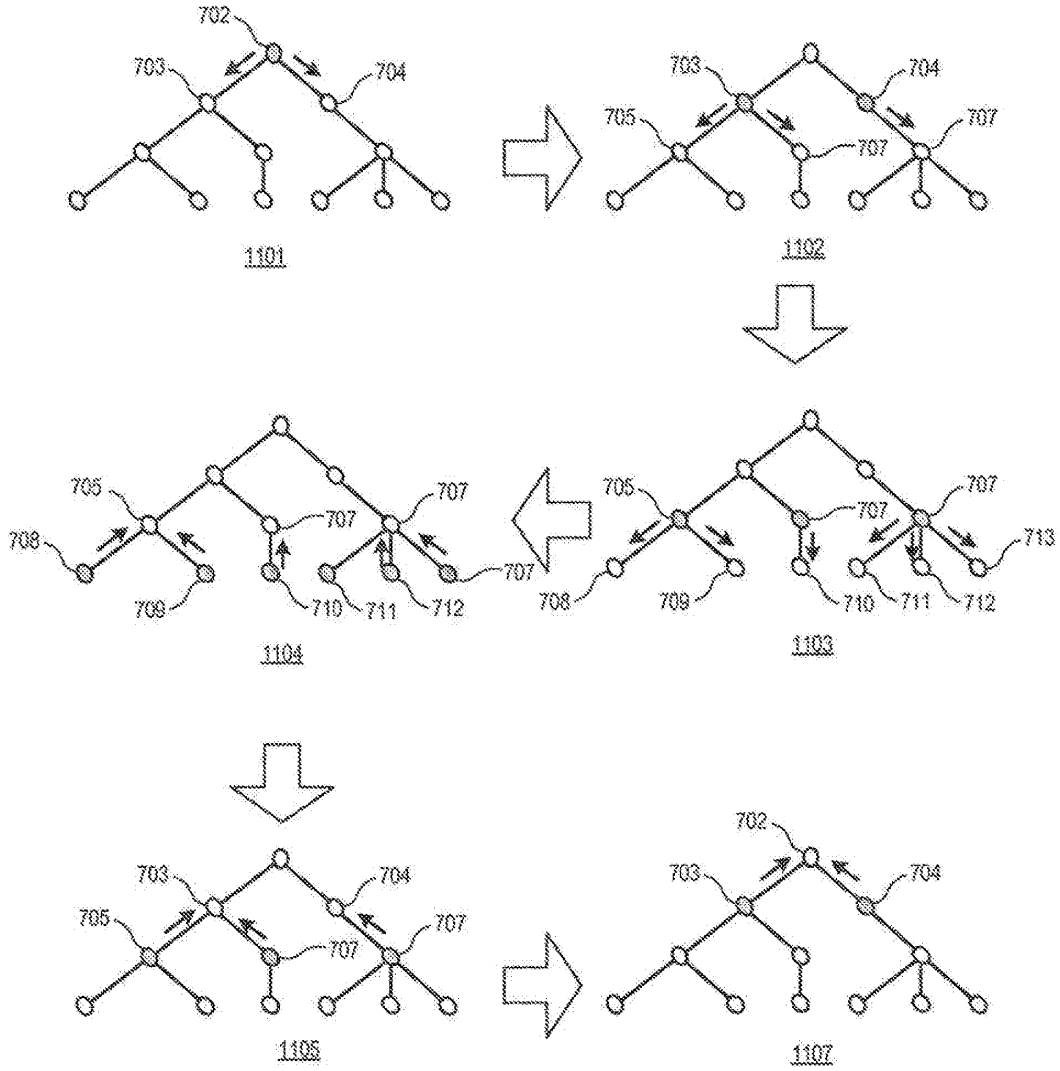
도면9


702
703
704
705
706
708
900
708
709
710
711
712
713

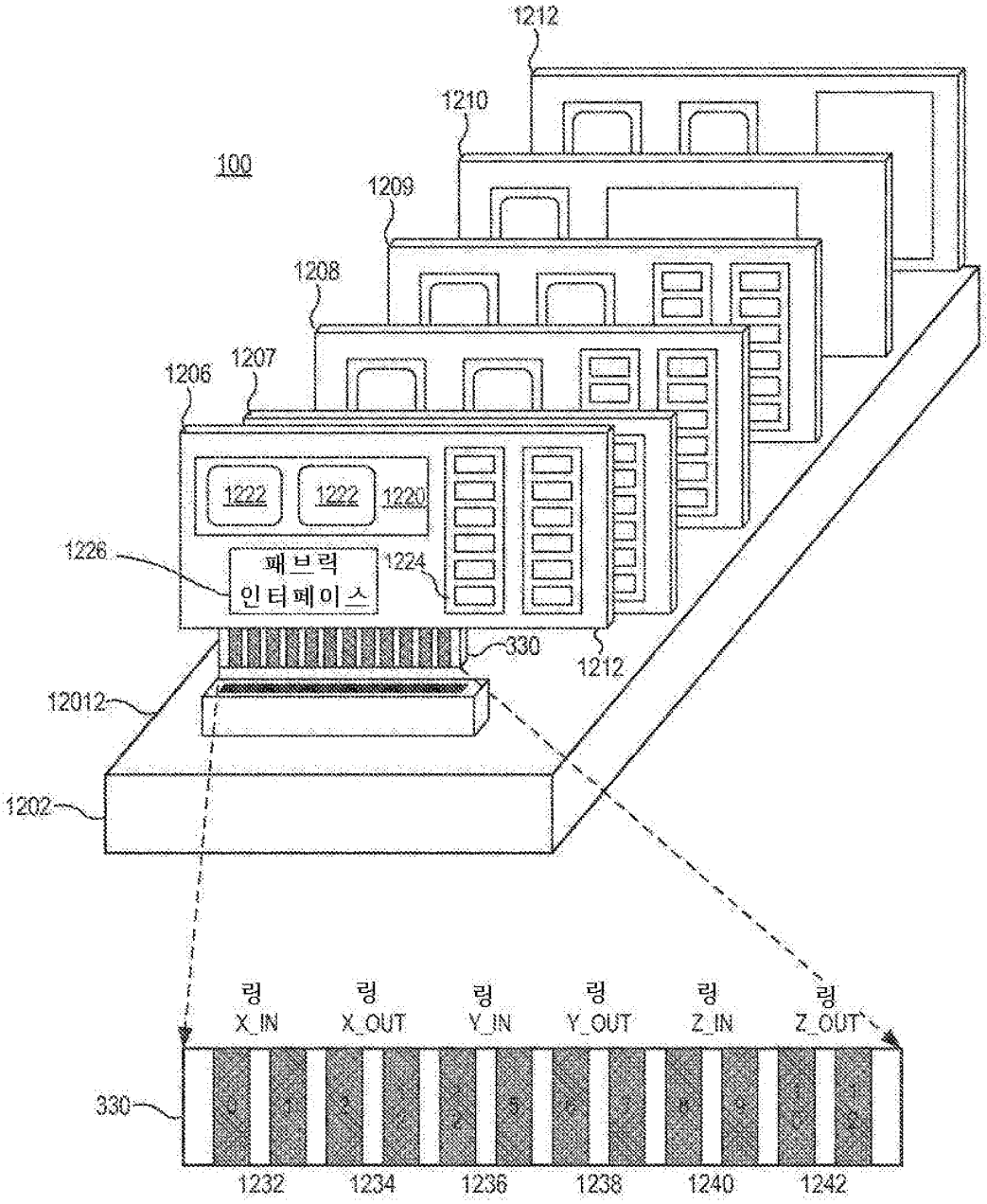
도면10


702
703
1001
703
705
1002
705
708
1004
705
708
1003
705
709
1005
705
709
1007

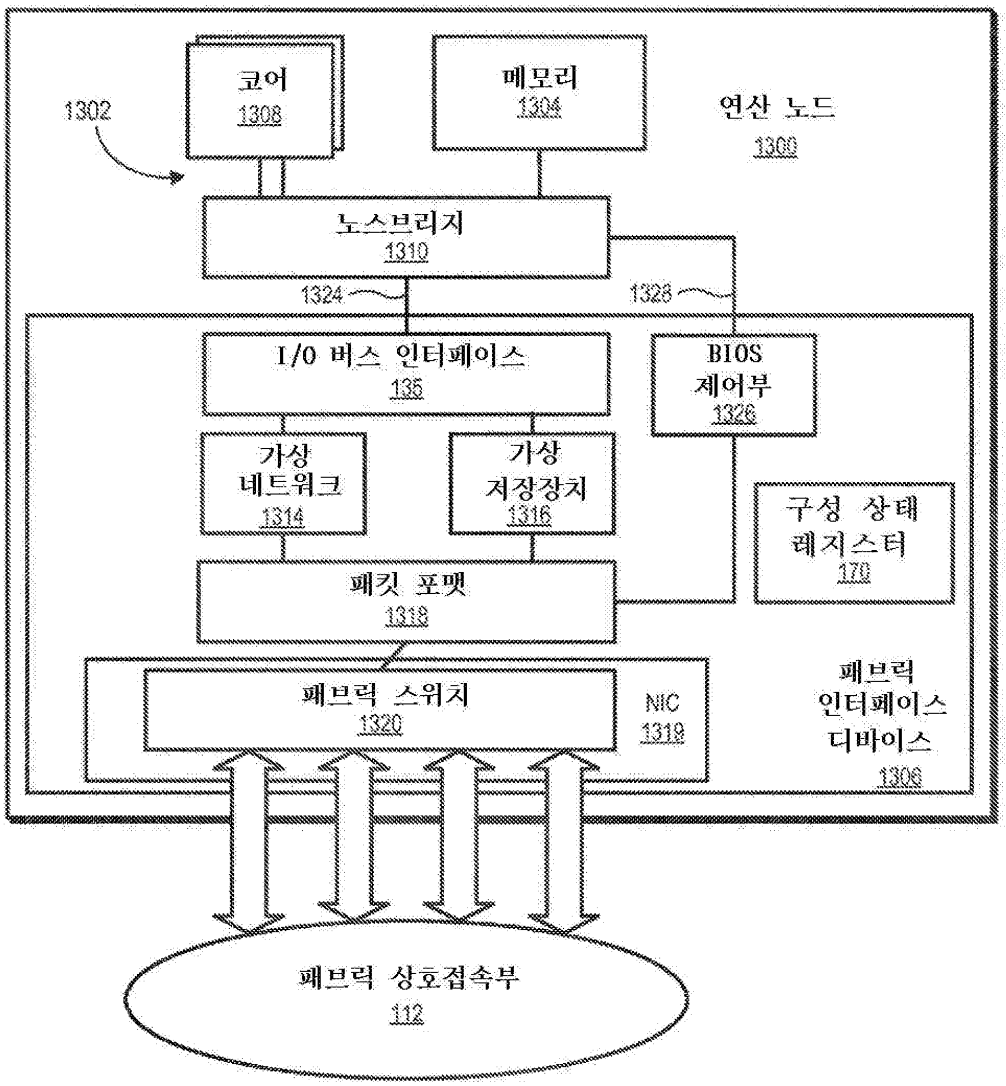
도면11


702
703
704
1101
703
704
705
707
707
1102
705
707
707
708
709
710
711
712
713
1103
705
707
707
708
709
710
711
712
1104
703
704
705
707
707
1105
702
703
704
1107

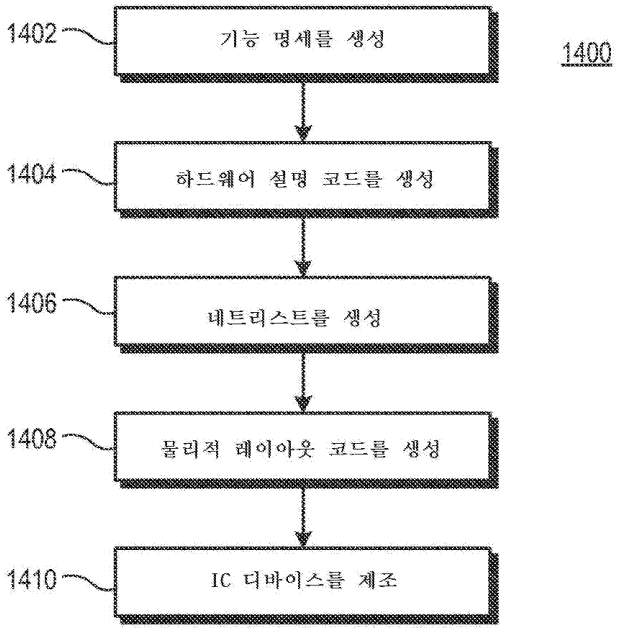
도면12


1212
1210
100
1209
1208
1206
1207
1222
1222
1220
1226
패브릭
인터페이스
1224
330
1212
12012
1202
링
X_IN
링
X_OUT
링
Y_IN
링
Y_OUT
링
Z_IN
링
Z_OUT
330
1232
1234
1236
1238
1240
1242

도면13


코어
1308
메모리
1304
연산 노드
1300
1302
노스브리지
1310
1324
1328
I/O 버스 인터페이스
136
BIOS
제어부
1326
가상
네트워크
1314
가상
저장장치
1316
구성 상태
레지스터
170
패킷 포맷
1318
패브릭 스위치
1320
NIC
1319
패브릭
인터페이스
디바이스
1306
패브릭 상호접속부
112

도면14


1402
기능 명세를 생성
1400
1404
하드웨어 설명 코드를 생성
1406
네트리스트를 생성
1408
물리적 레이아웃 코드를 생성
1410
IC 디바이스를 제조