(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl.[7]
G06F 9/46

(45) 공고일자 2004년06월23일
(11) 등록번호 10-0437227
(24) 등록일자 2004년06월14일

(21) 출원번호	10-2001-7006200	(65) 공개번호	10-2001-0080468
(22) 출원일자	2001년05월16일	(43) 공개일자	2001년08월22일
번역문 제출일자	2001년05월16일		
(86) 국제출원번호	PCT/SE1999/002063	(87) 국제공개번호	WO 2000/29941
(86) 국제출원출원일자	1999년11월12일	(87) 국제공개일자	2000년05월25일

(81) 지정국

국내특허 : 알바니아, 아르메니아, 오스트리아, 오스트레일리아, 아제르바이잔, 보스니아-헤르체고비나, 바베이도스, 불가리아, 브라질, 벨라루스, 캐나다, 스위스, 리히텐슈타인, 중국, 쿠바, 체코, 독일, 덴마크, 에스토니아, 스페인, 핀랜드, 영국, 그루지야, 헝가리, 이스라엘, 아이슬란드, 일본, 케냐, 키르기즈, 북한, 대한민국, 카자흐스탄, 세인트루시아, 스리랑카, 라이베리아, 레소토, 리투아니아, 룩셈부르크, 라트비아, 몰도바, 마다가스카르, 마케도니아, 몽고, 말라위, 멕시코, 노르웨이, 뉴질랜드, 슬로베니아, 슬로바키아, 타지키스탄, 투르크메니스탄, 터어키, 트리니다드토바고, 우크라이나, 우간다, 우즈베키스탄, 베트남, 폴란드, 포르투칼, 루마니아, 러시아, 수단, 스웨덴, 싱가포르, 가나, 감비아, 인도네시아, 시에라리온, 유고슬라비아, 짐바브웨, 크로아티아, 그레나다, 인도, 아랍에미리트, 남아프리카, 코스타리카, 도미니카연방, 모로코, 탄자니아,

AP ARIPO특허 : 케냐, 레소토, 말라위, 수단, 스와질랜드, 우간다, 가나, 감비아, 짐바브웨, 시에라리온, 탄자니아,

EA 유라시아특허 : 아르메니아, 아제르바이잔, 벨라루스, 키르기즈, 카자흐스탄, 몰도바, 러시아, 타지키스탄, 투르크메니스탄,

EP 유럽특허 : 오스트리아, 벨기에, 스위스, 리히텐슈타인, 독일, 덴마크, 스페인, 프랑스, 영국, 그리스, 아일랜드, 이탈리아, 룩셈부르크, 모나코, 네덜란드, 포르투칼, 스웨덴, 핀랜드, 사이프러스,

OA OAPI특허 : 부르키나파소, 베넹, 중앙아프리카, 콩고, 코트디브와르, 카메룬, 가봉, 기네, 말리, 모리타니, 니제르, 세네갈, 차드, 토고, 기네비쏘,

(30) 우선권주장

9803901-9	1998년11월16일	스웨덴(SE)
9902373-1	1999년06월22일	스웨덴(SE)

(73) 특허권자
텔레폰아크티에볼라게트 엘엠 에릭슨
스웨덴 스톡홀름 에스-126 25 텔레폰바겐 30

(72) 발명자
홀름베르크페르안데르스
스웨덴왕국스톡홀름에스-11842플린트바켄18

클링라르스-외르얀
스웨덴왕국쇠데르텔예에스-15152쿰멜베겐17

존슨스텐에드워드
스웨덴왕국파르스타에스-12342리스빅스가탄3

에게란트테르예
스웨덴왕국배름되에스-13956뵈르크빅스알레배크2

(74) 대리인 권동용
김기종
서장찬
최재철

심사관 : 김승조

(54) 다중 처리 시스템에서 개선된 결과 처리 방법 및 장치

요약

본 발명은 일반적으로, 작업이 다중 프로세서(30-1, 30-2, ..., 30-N)에 의해 병렬로 추론적으로 실행되는 다중 처리 시스템에 관한 것이다. 보다 대충적인 병렬 처리, 소위 작업 레벨 병렬 처리의 존재를 추론하고 작업의 병렬 실행을 방지하는 종속성이 검출되는 경우에만 순차 실행의 지원에 의해, 고도의 병렬 처리가 도출될 수 있다. 본 발명에 따르면, 전용 메모리 버퍼가 통신 메시지, 운영 시스템 호출 또는 새로운 작업 신호와 같은 추론적으로 실행되는 작업의 결과를 유지하도록 추론적으로 할당되고, 이들 결과는 할당된 메모리 버퍼에 추론적으로 기록된다. 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때, 할당된 메모리 버퍼에 관한 포인터가 전송된 포인터에 의해 메모리 버퍼에 액세스할 수 있는 입/출력 장치(10)에 전송된다. 이러한 방법으로, 메시지 및 신호를 추론적으로 전용 메모리 버퍼에 기록함으로써, 추가의 병렬 처리가 도출될 수 있다.

대표도

도 1

색인어

다중 프로세서, 입/출력 장치, 작업 분배기, 종속성 검사기, 작업 신호 메모리

명세서

기술분야

본 발명은 일반적으로 처리 기술에 관한 것이며, 특히 다중 프로세서 또는 처리 유닛을 갖는 처리 시스템에 관한 것이다.

배경기술

컴퓨터 또는 처리 시스템에서 향상된 처리 용량의 요구가 증가함에 따라 더욱 고속이고 더욱 효율적인 프로세서가 요구되고 있다. 처리 용량을 증가시키는 종래의 방법은 보다 높은 클록 주파수 및 보다 고속의 메모리의 사용에 좌우된다.
특히 응용 소프트웨어를 변경시키지 않고 처리 시스템의 처리 용량을 증가시키는 공통의 방법은 인접한 명령 사이에 발견되는 정밀한 병렬 처리(fine-grained parallelism)를 탐색하는 소위 슈퍼스칼라 프로세서(superscalar processor)를 사용하는 것이다. 슈퍼스칼라 프로세서에서, 이 프로세서내의 기능 유닛이 여러 개의 명령을 병렬로 동시에 실행하도록 배치된다. 이를 종종 명령 레벨 병렬 처리라고 한다.
처리 용량을 증가시키는 또 다른 방법은 처리 시스템을 멀티프로세서 시스템으로서, 즉 처리 시스템을 병렬로 동작하는 다중 프로세서로 형성하는 것이다. 일부 처리 시스템은 대칭적인 멀티프로세서(SMP)로 공지되어 있는 구조를 통해 고성능 처리를 달성한다. 슈퍼스칼라 프로세서에 의해 달성되는 정밀한 병렬 처리와 대조적으로, SMP 구조는 현재의 프로그래밍 원리에 따라 설계된 프로그램에 명시적으로 규정되거나, 편집 중에 단일 프로세서 시스템 상에서의 순차 실행을 위해 설계된 프로그램으로부터 도출되는 대충적인 병렬 처리(coarse-grained parallelism)를 이용한다. SMP 구조에서, 여러 개의 태스크(task) 중 각각 하나는 여러 개의 프로세서 중의 각각 하나에서 실행된다. 이들 태스크는 상호 협력하는 프로세스 또는 스레드(thread)이다.
소위 멀티스칼라(multiscalar) 프로세서가 논문(Multiscalar Processors by Sohi, Breach and Vijaykumar, 22nd A

nnual International Symposium on Computer Architecture, Santa Margherita, Ligure, Italy(1995))에 기재되어 있다. 멀티스칼라 모델 실행에서, 제어 종속성에 관하여 정적 프로그램을 나타내는 제어 흐름도(CFG)는 태스크로 분할되고, 멀티스칼라 프로세서는 태스크내의 명령 중 어느 하나를 검사하기 위해 정지함 없이 태스크형의 단계를 가지고 추론적(speculative)으로 CFG를 다룬다. 태스크는, 태스크의 초기 프로그램 카운터를 처리 유닛에 전달함으로써 실행을 위해 처리 유닛의 집합 중 하나에 할당된다. 그 후, 다중 태스크는 처리 유닛에서 병렬로 실행할 수 있으므로, 사이클당 다수 명령의 집합 실행 레이트를 발생시킨다. 멀티스칼라 하드웨어의 기능은 CFG를 다루고, 태스크를 처리 유닛에 할당하며, 이들 태스크를 순차 실행의 형태로 실행하는 것이다. 부가적인 하드웨어가 추론적인 메모리 동작을 유지하고, 메모리 종속성의 위반을 검출하고, 필요한 경우 보정 동작을 개시하도록 제공된다. 헤드에서의 태스크만이 비추론적인 것으로 보증된 태스크이기 때문에, 헤드를 제외한 모든 유닛에 의해 실행되는 메모리 동작은 추론적이다.
미국 특허 5,832,262호는 통상적으로 하나 이상의 다른 프로세서에 의해 실행될 다중 태스크를 관리하도록 스케쥴링 프로세서와 관련하여 작업하는 실시간 하드웨어 스케쥴러(scheduler)에 관한 것이다. 스케쥴러는 2개의 실시간 하드웨어 스케쥴링 유닛을 통해 공유 메모리내의 원형 FIFO 큐우를 관리하며, 하나의 스케쥴링 유닛은 실행 대기 태스크를 관리하고, 다른 스케쥴링 유닛은 완료된 태스크를 관리한다. 스케쥴링 프로세서는 태스크 제어 블록을 생성하여 특정 메모리 위치에서 공유 메모리에 그 블록을 저장함으로써 새로운 태스크를 스케쥴링하도록 요구하는 동작을 한다. 그 후, 그 위치에 대한 포인터(pointer)가 스케쥴링 프로세서에 의해 실시간 하드웨어 스케쥴러로 송신된다. 슬레이브(slave) 프로세서는 실행 대기 태스크의 공유 메모리에서 위치에 대한 포인터를 검색할 수 있다. 그 후, 슬레이브 프로세서는 실제 태스크를 검색하도록 포인터를 사용하고, 태스크를 실행하며, 태스크 완료의 스케쥴러를 고지한다. 따라서, 상기 특허는 종래의 멀티태스킹 시스템에서의 간접 신호 처리에 관한 것이다.
미국 특허 5,781,763호는 명령이 추론적으로 실행되는 프로세서를 개시하고 있다. 이 프로세서는 소정의 동작에 전용되는 다수의 슈퍼스칼라 실행 유닛을 가짐으로써 명령 레벨 병렬 처리가 얻어진다.
미국 특허 5,560,029호는 멀티스레딩용으로 사용되는 동기화 코프로세서(coprocessor)를 갖는 데이터 처리 시스템을 개시하고 있다.
미국 특허 5,195,181호는 일반적으로, 메시지 처리에 관한 것으로, 특히 포인터를 사용함으로써 분리된 수신 및 송신 프로세서 사이에서 메시지를 전송하는 방법에 관한 것이다.
그러나, 더욱 효율적인 처리 시스템에 대한 요구가 여전히 존재한다.

발명의 상세한 설명

일반적으로, 본 발명은 작업이 추론적으로, 병렬로 실행되는 다중 처리 시스템에 관한 것이다. 보다 대충적인 병렬 처리, 소위 작업 레벨 병렬 처리의 존재를 추론하고, 작업의 병렬 실행을 방지하는 종속성이 검출되는 경우에만 순차 실행의 지원에 의해, 고도의 병렬 처리가 도출될 수 있다. 이 방법은 상이한 프로세서에 의해 작업이 병렬로 실행되게 하지만, 실행된 작업은 순서대로 인도(commit)되는데, 즉 리타이어(retire)되어야 하고, 추론적으로 실행된 작업은 메모리 기록 및 신호 전송을 인도하도록 우선 순위를 대기한다.
작업의 추론적인 실행은 통상적으로, 추론이 성공적인지를 결정하기 위해 종속성을 검출하는 메커니즘이나 추론적으로 실행된 작업의 결과를 유지시키는 메커니즘을 필요로 한다. 추론이 성공적인 경우, 그 결과는 최종적으로 인도되고, 추론이 실패인 경우, 그 결과는 플러쉬(flush)되고 대응하는 작업이 재개된다. 통상적으로, 추론적으로 실행된 작업 결과는 중간 메모리 위치에 유지되고, 인도 시에, 결과는 중간 메모리 위치로부터 메인 메모리내의 실제 위치로 및/또는 외부 유닛으로 전송된다. 대안적으로, '실제 위치'로부터 구 데이터(old data)가 히스토리 로그(history log)에 저장되고, 추론적인 결과가 실제 위치에 기록된다. 추론이 실패인 경우, 실제 위치가 구 데이터를 복구하기 위해 히스토리 로그를 사용함으로써 회복된다. 추론이 성공적인 경우, 추론적인 데이터는 이미 실제 위치의 적절한 장소에 있다.
그러한 처리 시스템의 개발 중에, 추론 자체가 시스템 성능을 향상시키는 매우 효율적인 방법이지만, 추론적인 결과를 처리하는 메커니즘은 성능을 더욱 향상시키는 병목(bottleneck)을 구성한다.
본 발명은 이러한 및 다른 관련 문제점을 효율적인 방법으로 극복한다.
본 발명의 일반적인 목적은 고도의 병렬 처리를 제공할 수 있는 효율적인 다중 처리 시스템을 제공하는 것이다.
본 발명의 다른 목적은 다중 처리 시스템에서 결과를 처리하는 효율적인 방법을 제공하는 것이다.
이들 및 그외 다른 목적은 첨부한 특허 청구의 범위에 의해 규정된 바와 같은 본 발명에 의해 충족된다.
본 발명에 따르는 일반적인 개념은 추론적으로 실행된 작업의 결과를 유지시키는 사적 메모리 버퍼를 추론적으로 할당하고, 그 결과를 직접 할당된 메모리 버퍼에 추론적으로 기록하는 것이다. 인도할 때, 우선 순위가 추론적으로 실행된 작업에 할당되고, 할당된 메모리 버퍼에 관한 포인터만이 전달 포인터(transferred pointer)에 의해 메모리 버퍼로부터의 결과에 액세스할 수 있는 입/출력 장치에 전달된다. 이러한 메커니즘은 I/O 프로세서, 네트워크 제어기, 이더넷 회로 또는 작업 신호 스케쥴러와 같은 I/O 장치에 의해 액세스 가능한 전용 메모리 버퍼(private memory buffer)로의 통신 메시지, 운영 시스템 호출 및 작업 신호의 추론적인 기록을 지원한다. 이것은 대응하는 작업의 실행이 추론적으로 실행될 뿐만 아니라 메모리 버퍼로의 할당 및 기록도 추론적으로 실행되기 때문에, 더욱 많은 병렬 처리가 도출되는 것을 의미한다.
본 발명은 또한 포인터만이 회복할 필요가 있고 개별 데이터의 회복이 필요로 되지 않기 때문에, 종속성이 검출될 때 고속 복구를 가능하게 한다.
프로세서는 추론적인 모드에서 통신 메시지, 운영 시스템 호출 또는 새로운 작업 신호에 관한 모든 정보를 전송할 수 있으므로, 정확하게 데이터의 송신을 확산시키고 대역폭 요건을 저하시킨다.

특수용 하드웨어를 사용하는 하드웨어 수행 추론에 있어서, 추론적인 변수의 상태를 유지시키고 이들 하드웨어 큐우 상으로의 압력을 감소시키기 위해 사용되는 하드웨어 큐우를 통과하도록 할당된 메모리 버퍼에 기록할 필요가 없다. 표준 재고(off-the-shelf) 마이크로프로세서를 사용하는 소프트웨어 수행 추론에 있어서, 추론적인 기록을 행하는 코드가 실질적으로 감소된다.
메인 메모리내에 할당된 전용 메모리 버퍼에 직접 새로운 기록 및 객체의 추론적인 기록이 지원될 수 있다는 것을 알 수 있다. 추론적으로 실행된 작업에 의해 발생되는 새로운 기록 또는 객체는 전용 메모리 버퍼에 추론적으로 할당되고, 인도할 때, 우선 순위가 추론적으로 실행된 작업에 할당되며, 새로운 기록 또는 객체가 미리 메인 메모리내의 적절한 장소에 있고 추가적인 전송은 불필요하다.
그러나, 본 발명에 따르는 메커니즘은 일반적으로, 변수의 추론적인 기록에 대해 맞추어지지 않는데, 그 이유는 변수의 비추론적인 상태의 위치가 전용 메모리 위치가 아니라 공유 메모리 위치이어야 하기 때문이다. 본 발명은 아래의 장점을 제공한다:
- 고도의 병렬 처리가 제공되고;
- 추론적인 실행에 대한 비용이 최소화되며 - 추론적으로 할당된 메모리 버퍼로의 액세스가 롤백(roll-backs) 및 종속성 검사를 지원할 필요가 없다;
- 대역폭 요건이 감소되며;
- 기록 큐우내의 메모리 영역이 절감되어 처리 유닛이 정지되는 것을 방지한다;
- 포인터만 전달되기 때문에 인도 우선 순위에서 분담(off-loading)이 더욱 빠르다;
- 발생된 통신 메시지 및 작업 신호가 1회만 기록된다.
본 발명에 의해 제공되는 다른 장점은 본 발명의 실시예에 관한 아래의 설명을 통해서 이해할 것이다.
본 발명은 추가의 목적 및 장점과 함께 첨부하는 도면과 함께 이루어지는 이하의 설명을 참조함으로써 가장 잘 이해하게 될 것이다.

도면의 간단한 설명

도 1은 본 발명에 따르는 일반적인 처리 시스템의 간략화한 도면.
도 2는 본 발명에 따라 추론적으로 실행되는 작업에 의해 발생되는 결과를 처리하는 방법의 개략적인 흐름도.
도 3은 본 발명의 바람직한 실시예에 따르는 처리 시스템의 개략도.
도 4는 본 발명의 다른 바람직한 실시예에 따르는 처리 시스템의 개략도.

실시예

도면을 통하여, 동일한 참조 기호가 대응하거나 유사한 구성 요소에 사용된다.
본 발명을 더욱 잘 이해하기 위해, 본 발명에 따르는 추론적인 결과에 대한 일반적인 원리를 도 1을 참조하여 이제 설명한다.
도 1은 본 발명에 따르는 범용 처리 시스템의 간략화한 도면이다. 처리 시스템은 입/출력(I/O) 장치(10), 작업 분배기(20), 다중 프로세서 또는 처리 유닛(30-1 내지 30-N), 종속성 검사기(35) 및 메인 메모리(40)를 포함한다.
I/O 장치(10)는 신호 및 메시지를 수신 및 전송한다. 작업 분배기(20)는 I/O 장치(10)로부터 작업 신호를 수신하여 작업 신호를 작업의 병렬 실행을 위해 프로세서(30-1 내지 30-N)에 할당한다. 병렬 실행 중에, 작업 중 하나에는 인도 우선 순위가 할당되어 비추론적으로 실행되는 반면에, 다른 작업은 추론적으로 실행된다. 프로세서(30)는 상업용 대칭 멀티프로세서에서 하드웨어에서 전적으로 실현되는 실행 파이프라인에 이르는 임의의 종류의 처리 유닛일 수 있다. 종속성 검사기(35)는 추론적으로 실행된 작업 및 추론적인 실행이 성공적인지의 여부를 결정하도록 인도 우선 순위를 갖는 작업 사이의 데이터 종속성을 검출하도록 구성된다. 실행 중에, 상기 검사기(35)는 판독 및 기록의 프로그램 순서가 유지되는지를 검사하도록 데이터 액세스에 대한 종속성 검사를 행한다. 종속성 검사는 다수의 방법 중 어느 하나로 실현될 수 있다. 예를 들어, 후술되는 바와 같은 어드레스 비교 방법 도는 가변 마커(marker) 방법에 의해 실현될 수 있다.
본 발명에 따르면, 다수의 메모리 버퍼가 통신 메시지, 운영 시스템(OS) 호출 및 추론적으로 실행된 작업에 의해 발생되는 새로운 작업 신호와 같은 결과용으로 메인 메모리(40)에 할당된다. 바람직하게는, 메모리 버퍼는 버퍼의 풀(pool)(45-1 내지 45-N)로 분할되고, 각 프로세서(30)는 자체의 버퍼의 풀(45)을 갖는다. 편의상, 각 버퍼의 풀은 도 1에 표시되어 있는 바와 같이, 메인 메모리에 링크된 리스트로서 구현된다.
추론적으로 실행된 작업에 의해 발생되는 통신 메시지 또는 그와 유사한 것이 대응하는 프로세서(30)에 할당된 버퍼의 풀(45)내의 전용 메모리 버퍼에 추론적으로 할당되고, 그 메시지는 도 1에 표시되어 있는 바와 같이 할당된 전용 버퍼로 기록된다. 추론적인 실행은 메시지의 실제 전송을 포함하지는 않는다. 이것은 상기 메시지가 대응하는 작업이 인도 우선 순위를 얻을 때까지 할당된 버퍼로부터 통신 채널로 핸드오버되지 않는 것을 의미한다. 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때, 할당된 메모리 버퍼에 대한 포인터가 I/O 장치(10)에 전송된다. 메인 메모리(40)는 I/O 장치(10) 및 프로세서(30-1 내지 30-N)에 의해 공유되므로, I/O 장치(10)는 전송된 포인터에 의해 통신 메시지를 검색할 수 있다.
포인터는 다수의 방법 중 하나로 I/O 장치에 전송될 수 있다. 예를 들어, 포인터를 I/O 장치로 전송하는 3개의 기본적인 절차는 아래와 같다:
ⅰ) 인도 시에, 포인터가 관련 프로세서로부터 I/O 장치로 직접 기록된다;

ⅱ) 인도 시에, 포인터가 I/O 장치에 의해 포인터에 대해 검사된 메모리 수단에 기록된다.
ⅲ) 포인터가 중간 메모리 위치에 추론적으로 기록되고, 인도 시에, 포인터가 중간 메모리 위치로부터 I/O 장치로 전송된다.
추론이 실패임을 나타내는 종속성이 검출되는 경우, 할당된 메모리 버퍼는 간단하게 할당 취소되어 고속 복구를 가능하게 한다.
추론적인 실행 뿐만 아니라 통신 메시지, 새로운 작업 신호 및 OS 호출에 대한 버퍼의 추론적인 할당을 지원함으로써, 병렬 처리가 더욱 많이 이용된다.
본 발명은 또한, 기능적인 프로그래밍 언어로 새로운 레코드(records) 또는 객체 지향형 프로그래밍 언어로 새로운 객체의 추론적인 기록에 적용 가능하다. 이 경우에, 추론적으로 실행되는 작업에 의해 발생되는 새로운 레코드 또는 객체는 메인 메모리내의 전용 메모리 버퍼에 추론적으로 할당되어 전용 버퍼에 기록된다. 추론이 실패하는 경우, 전용 버퍼는 간단히 할당 취소된다. 반면에, 추론이 성공하는 경우, 새로운 레코드 또는 객체가 인도 시에 메인 메모리내의 적절한 위치에 미리 있어, 추가의 전송은 불필요하게 된다. 새로운 레코드/객체는 공유 메모리에 대한 대응하는 포인터를 핸드오버함으로써 예컨대, 포인터를 공유 메모리내의 변수로 기록함으로써 이용될 수 있다. 레코드/객체는 실행에 사용될 수 있고, 더 이상 필요 없게 될 때 간단히 폐기되고, 대응하는 메모리 버퍼는 할당 취소된다. 포인터는 또한, 외부 유닛으로 전송될 수도 있다.
객체 및 기록은 종종 통신 메시지 및 작업 신호보다 더욱 가변적인 크기를 가지며, 레코드/객체용 메모리 버퍼는 편리하게도 '힙(heap)'이라고 칭해지는 메모리 영역으로부터 할당된다. 통상의 프로그래밍 모델에 있어서, 예컨대, 유닛스 시스템에서 실행하는 C 프로그램의 경우에, 통상 프로세스당 하나의 힙이 존재하므로, 상이한 프로그램이 상호 데이터를 파괴할 수 없다. 그 후, 각 메모리 버퍼는 할당된 힙에 복귀되어야 한다.
본 발명에 따르는 메커니즘은 할당될 때 메모리 버퍼가 프리(전용)라는 사실을 이용하고, 결국 파괴될 수 있는 버퍼내에 필수불가결한 상태가 없다.
본 발명은 통상적으로 비추론적으로 실행되는 작업이 공유 메모리 위치에 유지되는 변수의 상태를 변경할 수 있어야 하기 때문에, 변수에 맞추어지지 않는다. 따라서, 추론적인 변수의 상태는 중간 위치에 일시적으로 저장되고 인도 시에 비추론적인 상태로 반전된 상태로 중간 위치로부터 실제 위치로 전송된다.
추론적으로 실행되는 작업이 임의의 가변 상태 뿐만 아니라 새로운 메시지, 신호 또는 기록/객체를 발생하지 않는 경우, 종속성 검사를 구현하는 임의의 메커니즘은 할당이 추론적으로 행해질지라도 메모리 버퍼가 한번에 하나의 작업에만 할당될 수 있기 때문에, 바이패스될 수 있다. 이것은 통상적으로 종속성 검사가 임시 코드를 갖는 판독 및 기록 명령에 의해 수행되는 소프트웨어 수행 추론에 대해 필수적이다.
단일 프로세서 시스템에 기록된 응용 소프트웨어가 표준 멀티프로세서 환경으로 이동되거나 그러한 환경에서 재사용되는 경우에, 응용 소프트웨어는 재편집 또는 그에 상당하는 것을 통해 변환될 수 있다. 예를 들어, 순차적으로 프로그램된 응용 소프트웨어는 종속성 검사용 코드 및 작업의 적절한 폐기 또는 롤백을 가능하게 하도록 변경된 변수의 카피(copy)를 저장하고나서 소프트웨어를 재편집(recompile)하는 코드를 포함하는 추론적인 실행을 지원하는 적절한 코드를 부가함으로써 자동으로 변환될 수 있다. 표준 멀티프로세서 시스템의 운영 시스템 또는 가상 머신도 또한 추론적인 실행을 지원하도록 변경될 수 있다. 예를 들어, 종속성 검사용 코드를 실행할 때 종속성이 검출되는 경우, 제어가 관련 작업을 폐기시키는 운영 시스템/가상 머신으로 전송될 수 있다.
도 2는 본 발명에 따르는 추론적으로 실행된 작업에 의해 발생되는 결과를 처리하는 방법의 개략적인 흐름도이다. 아래의 단계의 시퀀스는 컴퓨터 또는 프로세서 시스템의 소자에 의해 실행될 수 있다. 여러 단계가 특수용 하드웨어에 의해, 프로세서에 의해 실행되는 프로그램 명령에 의해, 또는 이들의 조합에 의해 실행될 수 있다. 단계 (101)에서, 작업은 병렬로 추론적으로 실행되고, 인도 우선 순위는 프로그래밍 모델에 의해 허용돠는 순서에 따라 한번에 하나씩 작업에 연속적으로 할당된다. 예를 들어, 작업은 전체적인 생성 순서에 따라서 인도, 즉 리타이어된다. 단계 (102)에서, 전용 메모리 버퍼가 추론적으로 실행되는 작업의 결과를 유지시키기 위해 추론적으로 할당되고, 단계 (103)에서, 추론적인 결과가 할당된 메모리 버퍼에 기록된다. 실행 중에, 종속성 검사가 단계 (104)에 표시되어 있는 바와 같이 데이터 변수 액세스에서 행해진다. 종속성이 추론적으로 실행되는 작업에서 검출되지 않는 경우(N), 대응하는 메모리 버퍼에 대한 포인터는, 인도 우선 순위가 최종적으로 추론적으로 실행되는 작업에 할당될 때 I/O 장치로 전달된다(단계 105). 반면에, 종속성이 검출되는 경우(Y), 대응하는 작업은 플러쉬되고, 할당된 메모리 버퍼가 할당 취소되며(단계 106), 작업의 실행이 재개된다(단계 101로 복귀).
도 3은 본 발명의 바람직한 실시예에 따르는 간략화된 처리 시스템을 도시하는 개략도이다. 이 처리 시스템은 네트워크 제어기(10)의 형태의 I/O 장치, 프로세서(30-1, 30-2), 종속성 검사기(35) 및 메인 메모리(40)를 포함한다. 표준 네트워크 제어기(10)의 일 예는 AMD PC net-FAST 칩(AM79C971)과 같은 이더넷 제어기이다. 네트워크 제어기(10)는 통상적으로, 착신 및 발신 메시지 및 신호를 처리하는 송신기 및 수신기 섹션을 갖는다. 수신기 섹션은 착신 메시지를 저장하는 다수의 수신 버퍼(41) 및 버퍼에 대한 포인터를 저장하는 대응하는 수신 메모리 링(42)을 포함하거나 이와 결합되어 있다. 통상적으로, 작업은 네트워크 제어기(10)에 의해 수신 버퍼(41)로 기록된 메시지에 대한 포인터용의 수신 링(42)을 검사하기 위해 새로운 메시지가 도달할 때 인터럽트 시에 또는 주기적으로 프로세서 중 하나에서 개시된다. 메시지에 대한 포인터가 발견되는 경우, 작업 신호가 생성되어 프로세서 중 하나에 의해 실행하기 위해 다른 작업 신호와 함께 스케쥴링된다. 도 3에는 단지 2개의 프로세서만 도시되어 있지만, 병렬 작업 처리를 위해 2개 이상의 프로세서를 사용하는 것이 가능하다는 것은 명백하다.
추론적인 메시지 '전송'을 위해, 바람직하게는 추론적으로 실행되는 작업에 의해 발생되는 메시지 뿐만 아니라 다른 신호 및 통상적으로 메모리 어드레스의 형태로 중앙 포인터 리스트를 프리 버퍼에 유지시키는데 사용되는 메인 메모리(40)내의 전용 메모리 버퍼의 공통 풀(common pool)이 존재한다. 각 프로세서(30)는 공통 풀내에 프리 버퍼에 대

한 포인터의 자체의 소위 '프리 리스트'를 가지므로, 버퍼의 공통 풀을 소위 송신 버퍼의 다수의 소형 프로세서 전용 풀(45-1, 45-2)로 논리적으로 분할한다. 버퍼의 소형 전용 풀(45-1, 45-2)은 프리 리스트 및 중앙 포인터 리스트에 적절한 변화를 줌으로써 소형 풀내의 고/저 워트마크(watermark)에서 공통 풀로부터 더 많은 버퍼로 채워질 수 있거나/공통 풀로 버퍼를 해제(release)할 수 있다.
프리 리스트(32-1, 32-2)는 각각 프로세서(30-1, 30-2)내의 물리적인 리스트이거나 메인 메모리(40)에 할당된다. 추론적으로 실행되는 작업에 의해 발생되는 새로운 메시지에 대한 메모리 버퍼의 추론적인 할당은 대응하는 프로세서(30)와 관련된 포인터의 프리 리스트(32)의 헤드를 참고함으로써 행해질 수 있다. 프리 리스트(32)의 헤드 엔트리에 저장된 포인터는 할당된 메모리 버퍼를 지시하고, 메시지는 할당된 버퍼에 추론적으로 기록된다. 소위 '후속 포인터'가 프리 리스트내의 후속 엔트리를 지시하도록 진행된다.
추론이 실패하는 경우, 할당된 메모리 버퍼는 프리 리스트의 헤드에 '후속 포인터'를 복귀시킴으로써 할당 취소된다. 추론이 성공하는 경우, 프리 리스트의 헤드 엔트리에 저장된 포인터는 인도 시에 프리 리스트(32)로부터 메인 메모리(40)내의 송신 메모리 링(46)으로 전송된 후, 리스트로부터 제거되므로 후속 포인터에 의해 표시되는 엔트리는 프리 리스트의 새로운 헤드로 된다.
메모리 링(46)은 프로세서(30) 및 네트워크 제어기(10)에 의해 공유되는 메모리에 할당되는 디스크립터 링(descriptor ring)으로 구성되는 것이 유용하다. 메시지의 실제 전송은 인도 시에 대응하는 프로세서(30)의 프리 리스트(32)로부터 디스크립터 링(46)으로 디스크립터를 전송함으로써 개시된다. 바람직하게는, 각 디스크립터는 a) 메시지 버퍼에 대한 어드레스(실제 포인터)로의 어드레스, b) 메모리 버퍼의 길이 및 c) 버퍼의 상태를 나타내는 상태 정보를 포함한다. 예를 들어, 네트워크 제어기(10)는 유효 디스크립터용 디스크립터 링을 주기적으로 폴링한다. 네트워크 제어기(10)가 유효 디스크립터를 발견한 경우, 디스크립터의 어드레스 필드로 지시되는 메모리 버퍼에 저장된 메시지를 전송한다.
디스크립터 링을 사용하여 버퍼 관리 시의 일반적인 정보에 대해, AMD PC net-FAST 칩 AM79971 데이터 시트 58-62 페이지를 참조하라.
프로세서당 하나의 풀을 사용하는 것이 최적이고, 이것이 플러쉬 시에 버퍼를 복귀시키는 매우 간단한 절차를 제공하며, 버퍼의 할당 및 버퍼 복귀가 여러 개의 여러 프로세서에 의해 병렬로 실행될 수 있고, 공통 풀에 액세스하는데 동기화는 필요 없다. 또한, 복귀된 버퍼는 동일한 프로세서에 의해 재사용되어, 프로세서당 하나의 데이터 캐시가 있을 때 높은 캐시 히트비(cache hit ratio)를 발생시킨다.
도 4는 본 발명의 또 다른 바람직한 실시예에 따르는 처리 시스템을 도시하는 개략도이다. 이 처리 시스템은 기본적으로, 작업 신호 스케쥴러의 형태의 I/O 장치(10) 및 처리 유닛(50)을 포함한다.
작업 신호 스케쥴러(10)는 입/출력 유닛(11), 작업 신호 메모리(12), 포인터 메모리(13), 메모리 인터페이스(14), TPU 유닛(To-Processors Unit)(15), FPU 유닛(From-Processors Unit)(16) 및 제어 유닛을 포함한다. 작업 신호 스케쥴러(10)는 외부 유닛으로부터 뿐만 아니라 처리 유닛(50)으로부터의 작업 신호를 수신하여 처리를 위해 작업 신호를 스케쥴링한다.
입/출력 유닛(11)은 지역 프로세서 및 다른 주변 장치와 같은 외부 유닛으로부터 작업 신호를 수신하여 이 신호를 메모리 인터페이스(14)를 통해 작업 신호 메모리(12)에 분포시킨다. 외부 유닛으로부터의 작업 신호가 통상적으로, 외부 이벤트로부터 발생하기 때문에, 시스템은 이벤트 구동형으로 간주될 수 있다. 작업 신호 메모리(12)는 특수 용도 메모리이거나, 처리 유닛의 메인 메모리(40)에 할당된다. 어느 경우에나, 작업 신호 메모리는 스케쥴러(10) 및 처리 유닛(50)에 의해 공유된다. 작업 신호 메모리는 외부 유닛에 작업 신호 및 통신 메시지를 저장하는 메모리 소자의 풀을 조절한다. 바람직하게는, 작업 신호 스케쥴러(10)는 신호 자체 대신에 작업 신호 및 통신 메시지에 대한 포인터로 대부분의 자신의 동작을 실행한다. 따라서, 각 작업 신호는 작업 신호 메모리(12)내의 메모리 소자에 대한 포인터와 관련되고, 대응하는 포인터는 대응하는 작업 신호가 작업 신호 메모리(12)에 기록될 때 메모리 인터페이스(14)를 통해 포인터 메모리(13)에 전송된다.
도 4에서, 단일 라인은 통상적으로, 포인터 신호(P)를 나타내고, 이중 라인은 작업 신호/통신 메시지(S) 또는 데이터 신호(D)를 나타낸다.
포인터 메모리(13)는 통상적으로 (도 4에 파선으로 된 박스로 표시된) 다수의 분리된 메모리 큐우로 분할된다. 하나의 큐우 대신에 여러 개의 큐우를 사용함으로써, 우선 순위를 조정하는 것이 가능하다; 상이한 우선 순위 레벨을 갖는 포인터가 상이한 메모리 큐우에 저장된다. 새로운 작업 신호가 작업 신호 스케쥴러(10)에 도달할 때, 작업 신호의 헤더에 포함되는 우선 순위 레벨 정보가 분석되고 대응하는 포인터가 위치되는 메모리 큐우가 식별된다. 이어서, 작업 신호(S)가 작업 신호 메모리(12)에 기록되고, 포인터(P)가 포인터 메모리(13)내의 선택된 큐우에 위치된다. 포인터 메모리(13)내의 큐우는 통상적으로, 우선 순위의 순서로 서비스되는 선입선출형(FIFO) 큐우로서 구성된다.
도달 순서로 및 우선 순위 레벨 정보에 따라서 포인터(P)를 저장함으로써, 스케쥴링 순서가 포인터 메모리(13)에서 얻어진다. 처리 유닛(50)이 새로운 작업 신호를 수락할 수 있을 때, 제어 유닛(17)은 최상위 우선 순위 레벨의 메모리 큐우내의 가장 오래된 포인터(P)가 TPU 유닛(15)으로 전송되게 한다. TPU 유닛(15)은 관련 작업 신호를 인출하도록 포인터(P)에 의해 표시되는 작업 신호 메모리(12)내의 메모리 소자를 판독하고, 검색된 작업 신호(S)를 처리 유닛(50)에 전송한다. 이러한 방법으로, 작업 신호 메모리(12)내의 작업 신호는 포인터 메모리(13)의 스케쥴링 순서에 따라 처리 유닛(20)에 전송된다.
바람직하게는, 작업 신호 메모리(12) 및 포인터 메모리(13)는 공통 메모리 예컨대, 메인 메모리(40)에서 수행된다.
바람직하게는, 작업 신호 스케쥴러(10)에서의 인터럽트 처리는 최상위 우선 순위 레벨의 작업 신호가 스케쥴러(10)로부터 처리 유닛(50)으로 전송되는 경우, 처리 유닛(50)내의 작업 인터럽트를 개시할 수 있어야 한다.

바람직하게는, 처리 유닛(50)은 한번에 하나의 우선 순위 레벨의 작업 신호를 처리한다. 작업 신호 스케쥴러(10)로부터의 상위 우선 순위 레벨의 신호는 하위 레벨의 신호를 인터럽트한다. 예를 들어, 작업 신호 스케쥴러(10)는 처리 유닛에 인터럽트 요구를 전송함으로써 인터럽트를 개시할 수 있고, 그 후, 처리 유닛은 현재 실행 중인 작업을 인터럽트한다.

처리 유닛(50)은 작업 신호 큐우(20), 복수의 처리 소자/프로세서(30-1 내지 30-4), 데이터 종속성을 조정하고 추론적으로 실행되는 작업의 결과를 일시적으로 저장하는 결합된 종속성 검사기 및 기록 큐우 장치(35), 및 프로그램 저장부(43) 및 데이터 저장부(44)로 분리되는 메모리 시스템(40)을 포함한다. 이 예에서, 처리 소자(30-1 내지 30-4)는 바람직하게는 실행 파이프라인의 형태이다. 따라서, 처리 소자는 이하에서는 실행 파이프라인이라고 한다.

작업 신호 스케쥴러(10)로부터의 작업 신호는 작업 신호를 저장하는 다수의 저장 위치를 갖는 작업 큐우(20)에서 버퍼링된다. 각 작업 신호는 작업 큐우(20)의 각각의 저장 위치에 부가 정보와 함께 저장된다.

일반적으로, 각 작업 신호는 헤더 및 신호 주요부를 포함한다. 관리 정보에 추가하여, 헤더는 통상적으로 프로그램 저장부(43)내에 소프트웨어 코드에 대한 포인터를 포함하고, 작업 신호의 주요부는 대응하는 작업의 실행을 위해 필요한 입력 오퍼랜드(operand)를 포함하여 작업 신호를 자기 구속적으로 만든다. 바람직하게는, 소프트웨어 코드는 하나 이상의 순람표를 통해 지시된다. 이것은 포인터가 순람표 엔트리를 지시하고 차례로 소프트웨어 코드를 지시하는 것을 의미한다. 신호 주요부는 지역 프로세서 도는 다른 프로세서와 같은 외부 유닛으로부터의 신호 메시지일 수 있다. 작업은 신호 헤더에 의해 규정되는 명령 스트림으로 정의될 수 있고, 작업은 작업 신호의 수신으로 시작하여 종료 작업 루틴의 호출에 의해 종료한다. 그러나, 작업 신호 자체는 통상적으로 어떠한 명령도 포함하지 않고, 프로그램 저장부(43)에 저장되는 소프트웨어 코드내의 명령에 대한 포인터 및 명령의 실행에 필요한 오퍼랜드만 포함한다.

바람직하게는, 실행 파이프라인(30-1 내지 30-4)은 상이한 작업을 병렬로 독립적으로 실행하기 위해 작업 큐우(20)내의 상이한 저장 위치로부터 작업 신호를 인출한다. 실행 파이프라인이 새로운 작업을 실행하기 시작하도록 프리일 때마다, 작업 큐우(20)는 할당되지 않은 작업 신호를 발견하도록 조사되어, 할당되지 않은 작업 신호가 파이프라인에 할당된다. 그 후, 작업 신호는 실행 파이프라인에서 처리되고, 대응하는 작업이 실행된다. 병렬 작업 실행 중에는 언제나, 작업 큐우(20)내의 단지 하나의 작업 신호가 인도 위치에 있어 작업 신호가 할당되는 실행 파이프라인이 대응하는 작업을 인도/리타이어할 수 있게 한다. 다른 실행 파이프라인내의 작업은 추론적으로 실행되고, 데이터 종속성이 종속성 검사기(35)에 의해 검출된 경우 플러쉬될 수 있다.

정보 흐름이 프로토콜에 의해 통제되는 시스템의 일반적인 요건은 특정 관련 이벤트가 검색된 순서로 처리되어야 한다는 것이다. 이것은 시스템이 어떻게 수행된다고 하더라도 시스템의 이형이 아니다. 작업 사이의 인도 순서는 통상적으로 처리 코어로의 도달에 의해 정해지고, 일반적으로 불변이다. 그러나, 상이한 우선 순위 레벨의 작업 신호를 처리하는 처리 시스템에서, 하위 우선 순위의 작업 신호 앞에 상위 우선 순위 레벨의 작업 신호를 위치시키는 것이 유용하므로, 동일한 우선 순위 레벨의 작업 사이의 인도 순서는 처리 유닛(50)으로의 도달에 의해 정해진다.

일반적으로, 각 실행 파이프라인은 프로그램 저장부로부터 명령을 인출하고, 명령을 디코딩하며, 명령을 실행하고, 메모리 기록 및 신호 전송을 실행하는 회로를 포함한다. 본 발명에 의해 사용될 수 있는 특정 실행 파이프라인의 일 예는 에릭슨 AXE 디지털 스위칭 시스템의 파이프라인이다.

작업 큐우(20)는 적절한 제어 소프트웨어 또는 하드웨어와 결합하여 실행 파이프라인(30-1 내지 30-4)으로의 작업 신호를 할당하고, 완료까지 추론적으로 실행된 작업의 트랙을 유지하며, 작업 신호에 인도 우선 순위를 순차적으로 할당하고, 인도된 작업 신호를 제거하는데 필요한 프로토콜을 관리한다.

작업 큐우(20)는 통상적으로 다수의 저장 위치를 갖는 보통의 큐우이다. 일 예로서, 작업 큐우(20)는 메모리의 소정 부분을 자체의 개별 메모리 어드레스를 각각 갖는 다수의 저장 위치로 논리적으로 분리함으로써 공통 메모리에서 수행될 수 있다. 일반적으로, 작업 큐우(20)내의 각 저장 위치는 예컨대, 아래의 표 1에 설명되어 있는 바와 같이 다수의 필드로 분할된다.

[표 1]

필드명	설명
유효	설정되는 경우, 저장 위치가 유효 작업 신호를 포함한다
점유	설정되는 경우, 작업 신호가 실행 파이프라인에 할당된다
종료	설정되는 경우, 대응하는 작업이 완료 시까지 실행되지만 인도된다
파이프 id 및 작업 id	작업 신호를 처리하는 파이프의 식별자 및 파이프에 제공되는 작업 신호의 식별자를 나타낸다
신호	신호 헤더 및 데이터

필드 '유효'는 저장 위치가 유효 작업 신호를 포함하는지의 여부를 나타내는데 사용된다. 작업 신호가 작업 신호 스케쥴러(10)로부터 수신될 때, 작업 신호는 작업 큐우(20)의 제 1 프리 위치, 즉 유효 플래그가 설정되지 않은 위치에 위치된다. 그 후, 유효 플래그는 '신호' 필드가 유효 작업 신호를 포함하는 것 및 그 위치가 점유되어 있는 것을 나타내도록 설정된다.

필드 '점유'는 작업 신호가 파이프라인에 할당되었는지의 여부를 나타내는데 사용된다. 작업 큐우(20)의 저장 위치내의 작업 신호가 실행 파이프라인에 할당될 때, 그 위치에 대한 점유 플래그는 작업 신호가 할당된 것을 나타내도록 설정된다.
필드 '종료'는 작업이 추론적으로 완료 시까지 실행된 것을 나타내는데 사용되고, 필드 '파이프 id 및 작업 id'는 파이프의 아이덴터티 및 작업 신호의 아이덴터티 및/또는 대응하는 작업이 그 파이프에 제공된 것을 나타내는데 사용된다. 파이프라인이 제 1 작업에 대한 종료 작업 루틴을 실행할 때, 작업이 완료 시까지 실행되어 인도될 준비가 되었다는 종료 플래그가 신호에 설정된다. 원하는 경우, 파이프라인은 제 2의 추론적인 작업 신호를 인출할 준비가 된다. 제 1 작업에는 당해의 파이프의 작업 id '1'이 할당되고, 파이프에서 실행될 제 2 작업에는 작업 id '2'가 할당된다. 파이프가 인도 우선 순위를 수신하기 전에 제 2 작업을 종료하는 경우, 제 3의 추론적인 작업이 실행될 수 있으며, 이하 동일하게 실행될 수 있다. 일반적으로, 파이프당 작업 수 및 복잡성 사이의 트레이드오프(trade-off)가 행해지고, 실제로 파이프당 작업의 수는 제한된다. 그러나, 이하에서는, 각 파이프라인이 한번에 하나의 작업만을 처리하는 것으로 가정한다.
추론적인 실행에 적합한 처리 시스템에서, 작업은 통상적으로 아래의 상태: 즉 비개시, 개시 및 종료될 수 있다. 이들 상태는 상이한 방법으로 작업 큐우(20)에서 인코딩될 수 있고, 표 1의 필드 유효, 점유 및 종료는 이들 작업 상태를 인코딩하는 하나의 방법의 예일 뿐이다.
바람직하게는, 작업 큐우(20)는 작업 큐우(20)내의 저장 위치가 인도 위치에 있는 것을 지시하는 포인터와 관련된다. 인도 위치에서 작업 신호는 인도 우선 순위를 갖고, 이 작업 신호를 처리하는 실행 파이프라인은 메모리 기록 및 인도 신호 전송을 실행할 수 있게 된다. 작업 신호가 인도 위치로 이동될 때, 대응하는 파이프라인은 모든 기록 동작 및 작업에 대한 신호 전송을 인도하기 시작한다. 작업이 완료 시까지 실행되고 모든 기록 및 신호 전송이 인도될 때, 위치는 유효 플래그를 클리어시킴으로써 해제되고, 작업 큐우(20)는 새로운 위치를 인도 위치로 이동시키도록 종래의 수단에 의해 진행되므로, 인도 우선 순위를 작업 큐우내의 작업 신호에 순차적으로 할당한다.
정보를 작업 큐우 필드에 기록하고 인도 우선 순위를 작업 신호에 순차적으로 할당하는 것과 같은 작업 큐우(20)를 처리하는 기능은 바람직하게는 소프트웨어 예컨대, 마이크로 코드 명령 프로그램 또는 어셈블러 프로그램의 형태에 의해 제어된다. 폐기를 처리하고 기록 큐우 장치를 제어하며 스케쥴링 메커니즘을 제어하는 것과 같은 처리 시스템의 다른 기능도 또한 소프트웨어로 수행될 수 있다.
종속성 검사기(35)는 통상적으로, 실행 파이프라인(30-1 내지 30-4)과 결합된 하나 이상의 판독 버퍼(RB)를 사용함으로써 수행된다. 파이프라인이 데이터 저장부(44)로부터 데이터를 인출할 때, 판독 어드레스(들)가 대응하는 판독 버퍼에서 버퍼링된다. 인도 우선 순위를 갖는 실행 파이프라인이 데이터 저장부(44)로의 가변 데이터(D)의 기록을 실행할 때, 작업간에 어떠한 데이터 종속이 있는지를 찾기 위해 데이터 저장부에 대한 기록 어드레스가 판독 버퍼(RB) 내의 판독 어드레스와 비교된다. 이러한 비교는 어드레스 비교기에서 통상 실행된다. 추론적으로 실행되는 작업에 의한 데이터 판독이 인도 작업에 의해 결국 변경되는 경우, 데이터 종속성이 존재하고 추론적으로 실행되거나 실행하는 작업은 폐기 및 재개되어야 한다.
추론적으로 실행되는 작업에 의해 제안되는 데이터 저장 변경은 기록 큐우 장치(35)에서 로깅(logged)되지만, 작업이 인도 우선 순위를 얻을 때까지 데이터 저장부(44)에 기록된다. 바람직하게는, 기록 큐우 장치(35)는 다수의 분리된 기록 큐우를 포함하고, 하나의 기록 큐우(WQ)는 실행 파이프라인(30-1 내지 30-4)의 각각의 하나에 전용된다. 높은 성능을 위해, 기록 큐우는 고속이지만 비교적 소형의 메모리인 캐시 메모리로 통상 수행된다. 일반적으로, 추론적인 작업이 인도 우선 순위를 얻을 때, 당해의 작업에 속하는 기록 큐우 엔트리는 데이터 저장부(44)에 즉시 기록된다. 그러나, 실행 시의 인도 작업은 기록 큐우에 자신의 데이터를 저장하지 않지만, 종속성 검사가 가능하도록 대응하는 메모리 어드레스를 어드레스 비교기에 간단히 전송한다.
대안적으로, 기록 큐우 장치(35)는 예를 들어, 결합된 종속성 검사기 및 기록 큐우를 개시하고 있는 국제 특허 출원 WO 88/02513호에 기재되어 있는 바와 같이 모든 실행 파이프라인에 공통인 단일 중심 기록 큐우의 형태이다.
작업 신호는 라운드 로빈 방식(round-robin fashion)으로 실행 파이프라인에 할당된다. 라운드 로빈에서, 파이프라인은 단계적으로 이동되는 원형 루프내에 배열되는 것으로 간주될 수 있다. 예를 들어, 이러한 배열은 파이프라인 및 기록 큐우측에서 인도 위치의 관리를 매우 간략화시킨다. 작업 큐우(20)에서 후속 작업 신호가 인도 위치로 이동될 때, 대응하는 실행 파이프라인이 이전의 인도 파이프라인과 비교하여 루프내의 후속 파이프라인으로 직선 방향으로 식별되고, 식별된 파이프라인에 전용되는 기록 큐우(WQ)에 저장된 결과가 인도된다.
본 발명에 따르면, 추론적으로 발생된 작업 신호 및 통신 메시지는 FPU 유닛(16) 및 메모리 인터페이스(14)를 통해 작업 신호 메모리(12)에 직접 기록된다. 대응하는 작업 신호 및 메시지가 발견될 수 있는 작업 신호 메모리(12)내의 메모리 영역에 대한 포인터만이 기록 큐우(WQ)에 일시적으로 저장되고, 인도 시에 FPU 유닛(16) 및 메모리 인터페이스(14)를 통해 포인터 메모리(13)로 전송된다.
실행되는 과다 병렬 처리 이외에, 고가의 기록 큐우 캐시내의 메모리 영역이 절감된다. 더욱이, 실행 파이프라인은 발생된 작업 신호 및 통신 메시지를 추론적인 모드로 전송할 수 있다. 이것은 작업 신호 데이터의 전송이 제때에 확산되어 처리 유닛(50) 및 작업 신호 스케쥴러(10) 사이의 인터페이스에 대해 더 낮은 대역폭 요건을 초래한다. 또한, 기록 큐우는 포인터만 전송되기 때문에 인도 시에 빨리 오프로드된다.
작업 신호 스케쥴러(10)는 프리 포인터의 풀을 생성하기 위해 작업 신호 메모리(12)내의 프리 메모리 소자에 대한 포인터를 처리 유닛(50)에 전송한다. 실행 파이프라인(30-1 내지 30-4)의 각각의 하나에는 다수의 그러한 프리 포인터(FP)가 통상 할당된다. 각 실행 파이프라인은 할당된 포인터에 의해 표시되는 메모리 소자 중 하나로 추론적으로 실행되는 작업에 의해 발생되는 새로운 작업 신호/통신 메시지(S)를 기록하고, 작업 신호/통신 메시지(S)가 기록된 메모리 소자에 관한 포인터(P)를 실행 파이프라인에 접속되는 기록 큐우(WQ)에 저장한다. 실행 파이프라인에 신호 전송

을 인도하도록 우선 순위가 할당될 때마다, 포인터(P)는 기록 큐우(WQ)로부터 포인터 메모리(13)내의 적절한 메모리 큐우로 또는 외부 유닛으로 전송하기 위한 I/O 유닛(11)으로 전송된다. 새로운 작업 신호의 경우에, 포인터 메모리(13)내의 적절한 메모리 큐우가 대응하는 작업 신호(S)의 우선 순위 레벨 정보에 기초하여 식별되고, 포인터(P)가 작업 신호와 동일한 우선 순위 레벨의 메모리 큐우로 기록된다. 포인터 메모리(13)내의 메모리 큐우는 FIFO 쿠우로 구성되고, 메모리 큐우는 우선 순위의 순서로 서비스되기 때문에, 스케쥴링 순서는 포인터 메모리에서 얻어진다.
작업 신호 스케쥴러(10)는 이 스케쥴러가 포인터를 더 이상 요구하지 않으면 즉시 TPU 유닛(15)을 통해 포인터(FP)를 처리 유닛(50)으로 전송함으로써 프리 포인터의 풀을 충전된 상태로 유지시킨다.
완전한 작업 신호 또는 완전한 통신 메시지를 한번에 기록할 필요는 없다. 큰 신호가 예컨대, 처리 유닛(50) 및 작업 신호 스케쥴러(10) 사이의 버스 인터페이스의 차단을 방지하기 위해 여러 조각으로 기록될 수 있다. 그 후, 처리 유닛(50)은 버스트 사이의 어드레스를 증가시켜야 하고, 각 버스트의 개시 시에 증가된 어드레스를 전송한다.
이상에서, 실행 파이프가 분리된 엔티티로서 설명되어 있으며, 이들은 각각 명령 실행을 위한 자체의 전용 자원을 갖는 것으로 설명되어 있다. 그러나, 일반적으로, 실행 파이프라인의 다수의 자원은 명령 실행에 전적으로 이용되지는 않는다. 그 후, 자원은 파이프 사이에서 공유될 수 있으므로, 요구 시에 또는 시간 다중화 방식으로 자원으로의 실행 파이프 액세스를 제공한다. 고도의 자원 공유를 수행하는 기술은 동시에 멀티스레딩된다. 동시 멀티스레딩에서, 거의 모든 자원이 공유되고, 각 파이프라인에 전용인 자원은 전용 프로그램 카운터와 물리적인 레지스터 및 명령내의 논리 레지스터 번호 사이의 전용 맵핑으로 제한된다.
포인터는 작업 신호 메모리내의 메모리 소자와 직접 관계가 없지만, 하나 이상의 순람표를 통해 작업 신호 메모리 영역과 관계가 있을 수 있다.
또한, 종속성 검사는 어드레스 비교 방법을 참조하여 설명되어 있다. 그러나, 다른 방법으로는, 종속성 검사가 또한 마커 방법에 의해 달성될 수 있으며, 여기에서 각 처리 유닛은 메인 메모리에서 변수의 사용을 마크하고, 종속성 충돌이 이러한 마킹에 기초하여 검출된다.
이상에 설명되어 있는 실시예는 단지 예로서 제공되어 있을 뿐, 본 발명이 이 실시예로 제한되는 것은 아님을 이해해야 한다. 본 명세서에 개시 및 청구되고 있는 기본적인 기초 원리를 갖는 추가의 변형, 변경 및 개량은 본 발명의 범위내에 있다.

(57) 청구의 범위

청구항 1.
처리 시스템에 있어서:
작업을 추론적으로 실행하는 다중 프로세서(30)와;
작업에 인도 우선 순위를 한번에 한 작업에 할당하는 수단과;
추론적으로 실행되는 작업의 결과를 유지하는 전용 메모리 버퍼를 추론적으로 할당하는 수단과;
인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 할당된 전용 메모리 버퍼에 관한 포인터를 입/출력 장치(10)에 전달하는 수단을 포함하는 것을 특징으로 하는 처리 시스템.

청구항 2.
제 1 항에 있어서,
추론적으로 실행되는 작업의 결과는 통신 메시지, 운영 시스템 호출 또는 새로운 작업을 나타내는 작업 신호의 형태인 것을 특징으로 하는 처리 시스템.

청구항 3.
제 1 항에 있어서,
상기 입/출력 장치(10)는 입/출력 프로세서, 네트워크 제어기, 이더넷 회로 또는 작업 신호 스케쥴러인 것을 특징으로 하는 처리 시스템.

청구항 4.
제 1 항에 있어서,
상기 전용 메모리 버퍼는 처리 시스템의 메인 메모리(40) 또는 프로세서(30) 및 입/출력 장치(10)에 의해 공유되는 또 다른 메모리에 상주함으로써, 상기 입/출력 장치(10)는 전달된 포인터에 의해 상기 결과에 액세스할 수 있는 것을 특징으로 하는 처리 시스템.

청구항 5.
제 1 항에 있어서,
각 프로세서는 메모리 버퍼를 할당 및 할당 취소하는 자체의 메모리 버퍼의 전용 풀(45)을 갖는 것을 특징으로 하는 처리 시스템.

청구항 6.
제 5 항에 있어서,
버퍼의 공통 풀로부터 보다 많은 버퍼를 갖는 버퍼의 전용 풀(45)을 충전시키고, 전용 풀(45)내의 고/저 워트마크에서 공통 풀로 전용 풀(45)로부터의 버퍼를 해제시키는 수단을 더 포함하는 것을 특징으로 하는 처리 시스템.

청구항 7.
제 1 항에 있어서,

각 프로세서는 메모리 버퍼의 풀(45)내에 프리 메모리 버퍼에 대한 자체의 포인터 리스트(32)를 갖고, 상기 버퍼 할당 수단은 대응하는 프로세서의 리스트의 헤더로부터 포인터를 인출함으로써 추론적으로 실행되는 작업의 결과를 유지하는 메모리 버퍼를 할당하는 것을 특징으로 하는 처리 시스템.

청구항 8.

제 1 항에 있어서,

추론적으로 실행되는 작업 및 인도 우선 순위를 갖는 작업 사이의 종속성을 검출하는 수단(35)과;

검출된 종속성에 응답하여 추론적으로 할당된 메모리 버퍼를 할당 취소시키는 수단을 더 포함하는 것을 특징으로 하는 처리 시스템.

청구항 9.

제 1 항에 있어서,

상기 전달 수단은 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 I/O 장치(10)에 의해 포인터에 대해 검사되는 메모리 수단(46)에 포인터를 기록하는 수단을 포함하는 것을 특징으로 하는 처리 시스템.

청구항 10.

제 9 항에 있어서,

상기 입/출력 장치(10)는 포인터용 메모리 수단(46)을 주기적으로 폴링하는 네트워크 제어기이고, 상기 네트워크 제어기는 포인터가 발견될 때 포인터와 관련된 전용 메모리 버퍼로부터 검색된 통신 메시지를 전송하는 것을 특징으로 하는 처리 시스템.

청구항 11.

제 9 항에 있어서,

상기 메모리 수단(46)은 처리 시스템의 메인 메모리(40)에 할당되는 링 구조이고, 상기 포인터는 링 구조(46)의 하나의 위치에 기록되는 것을 특징으로 하는 처리 시스템.

청구항 12.

제 1 항에 있어서,

상기 포인터를 메모리 수단(WQ)에 추론적으로 기록하는 수단을 더 포함하고, 상기 전달 수단은 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 메모리 수단(WQ)으로부터 입/출력 장치(10)로 포인터를 전달하는 것을 특징으로 하는 처리 시스템.

청구항 13.

제 12 항에 있어서,

상기 메모리 수단(WQ)은 하드웨어 큐우로서 수행되거나 처리 시스템의 메인 메모리(40)에 할당되는 기록 큐우인 것을 특징으로 하는 처리 시스템.

청구항 14.

제 1 항에 있어서,

상기 입/출력 장치(10)는 다중 프로세서에 의해 처리하는 작업 신호를 스케쥴링하는 작업 신호 스케쥴러이고, 상기 작업 신호 스케쥴러는 작업 신호를 저장하는 작업 신호 메모리(12)를 가지며;

상기 할당 수단은 추론적으로 실행되는 작업에 의해 발생되는 새로운 작업 신호를 유지하는 작업 신호 메모리(12)에 전용 메모리 버퍼를 추론적으로 할당하고, 상기 대응하는 포인터가 메모리 수단(WQ)에 추론적으로 기록되며;

상기 전송 수단은 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 새로운 작업 신호를 스케쥴링할 수 있도록 메모리 수단(WQ)으로부터 작업 신호 스케쥴러(10)로 포인터를 전달하는 것을 특징으로 하는 처리 시스템.

청구항 15.

제 14 항에 있어서,

각 작업 신호 및 이에 대응하는 포인터는 우선 순위 레벨 정보와 관련되고, 상기 작업 스케쥴러(10)는:

스케쥴링 순서를 얻기 위해 우선 순위 레벨 정보에 따라 그리고 도달 순서로 포인터를 저장하는 포인터 메모리(13)와;

포인터 메모리(13)내의 스케쥴링 순서에 따라 다중 프로세서(30)에 의해 처리하는 작업 신호 메모리(12)로부터 작업 신호를 전달하는 수단을 포함하는 것을 특징으로 하는 처리 시스템.

청구항 16.

제 15 항에 있어서,

상기 전달 수단은 포인터를 포인터 메모리(13)로 전달하여, 포인터를 스케쥴링 순서로 배치하는 것을 특징으로 하는 처리 시스템.

청구항 17.

처리 시스템에서 결과를 처리하는 방법에 있어서:

병렬로 작업을 추론적으로 실행하는 단계(101)와;

작업에 인도 우선 순위를 한번에 한 작업에 할당하는 단계와;

추론적으로 실행되는 작업의 결과를 유지하는 전용 메모리 버퍼를 추론적으로 할당하는 단계(102)와;

인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때, 대응하는 할당 메모리 버퍼에 관한 포인터를 전송된 포인터에 의해 대응하는 할당 메모리 버퍼에 액세스하는 입/출력 장치(10)로 전송하는 단계(105)를 포함하는 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 18.

제 17 항에 있어서,
추론적으로 실행되는 작업 및 인도 우선 순위를 갖는 작업 사이의 종속성을 검출하는 단계(104)와;
종속성이 추론적으로 실행되는 작업에 대해 검출되는 경우, 대응하는 추론적으로 할당된 메모리 버퍼를 할당 취소하는 단계(106)를 더 포함하는 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 19.
제 17 항에 있어서,
추론적으로 실행되는 작업의 결과는 통신 메시지, 운영 시스템 호출 또는 새로운 작업을 나타내는 작업 신호의 형태인 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 20.
제 17 항에 있어서,
상기 입/출력 장치(10)는 입/출력 프로세서, 네트워크 제어기, 이더넷 회로 또는 작업 신호 스케쥴러인 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 21.
제 17 항에 있어서,
상기 전송 단계는 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 I/O 장치에 의해 포인터에 대해 검사되는 메모리 수단으로 포인터를 기록하는 단계를 포함하는 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 22.
제 17 항에 있어서,
상기 포인터를 메모리 수단(WQ)에 추론적으로 기록하는 단계를 더 포함하고, 상기 전송 단계(105)는 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 메모리 수단(WQ)으로부터 입/출력 장치(10)로 포인터를 전달하는 단계를 포함하는 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

청구항 23.
제 17 항에 있어서,
상기 작업을 추론적으로 실행하는 단계는 다중 프로세서(30)에 의해 실행되고, 상기 방법은 프로세서가 메모리 버퍼를 할당 및 할당 취소시키는 메모리 버퍼의 자신의 전용 풀(45)과 각 프로세서를 결합시키는 단계를 더 포함하는 것을 특징으로 하는 처리 시스템에서의 결과 처리 방법.

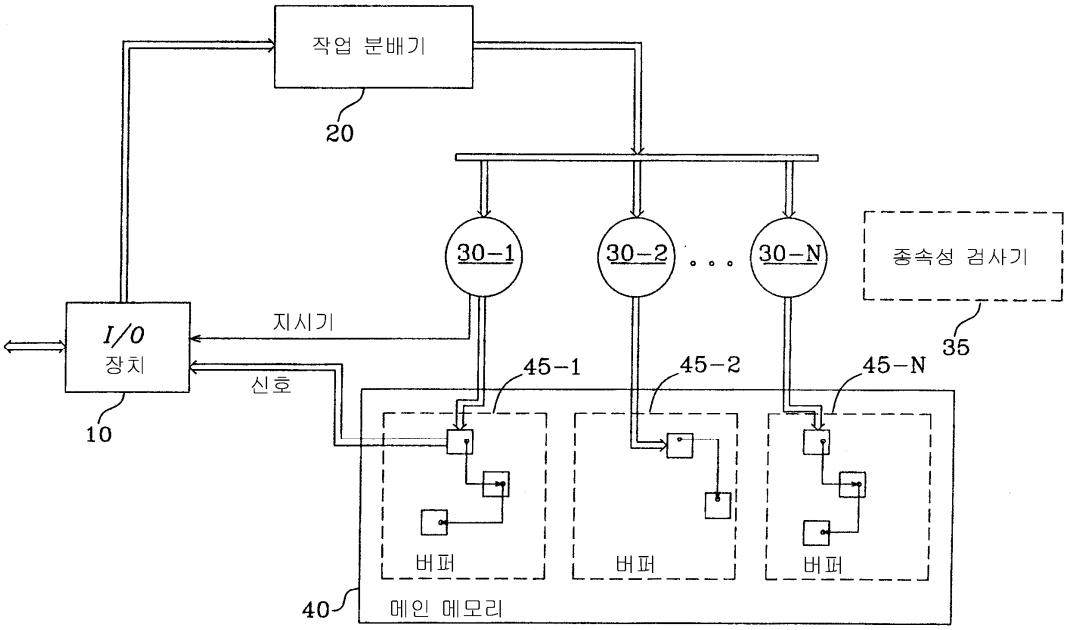
청구항 24.
처리 시스템에 있어서:
작업을 추론적으로 실행하는 다중 프로세서(30)와;
작업에 인도 우선 순위를 한번에 한 작업에 할당하는 수단과;
추론적으로 실행되는 작업에 의해 발생된 새로운 기록 또는 객체를 유지하는 처리 시스템의 메인 메모리(40)내에 전용 메모리 버퍼를 추론적으로 할당하는 수단을 포함하며, 새로운 기록 또는 객체는 인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 메인 메모리(40)내의 적소에 있는 것을 특징으로 하는 처리 시스템.

청구항 25.
제 24 항에 있어서,
인도 우선 순위가 추론적으로 실행되는 작업에 할당될 때 할당된 전용 메모리 버퍼에 관한 포인터를 공유 메모리로 핸드오버시키는 수단을 더 포함하는 것을 특징으로 하는 처리 시스템.

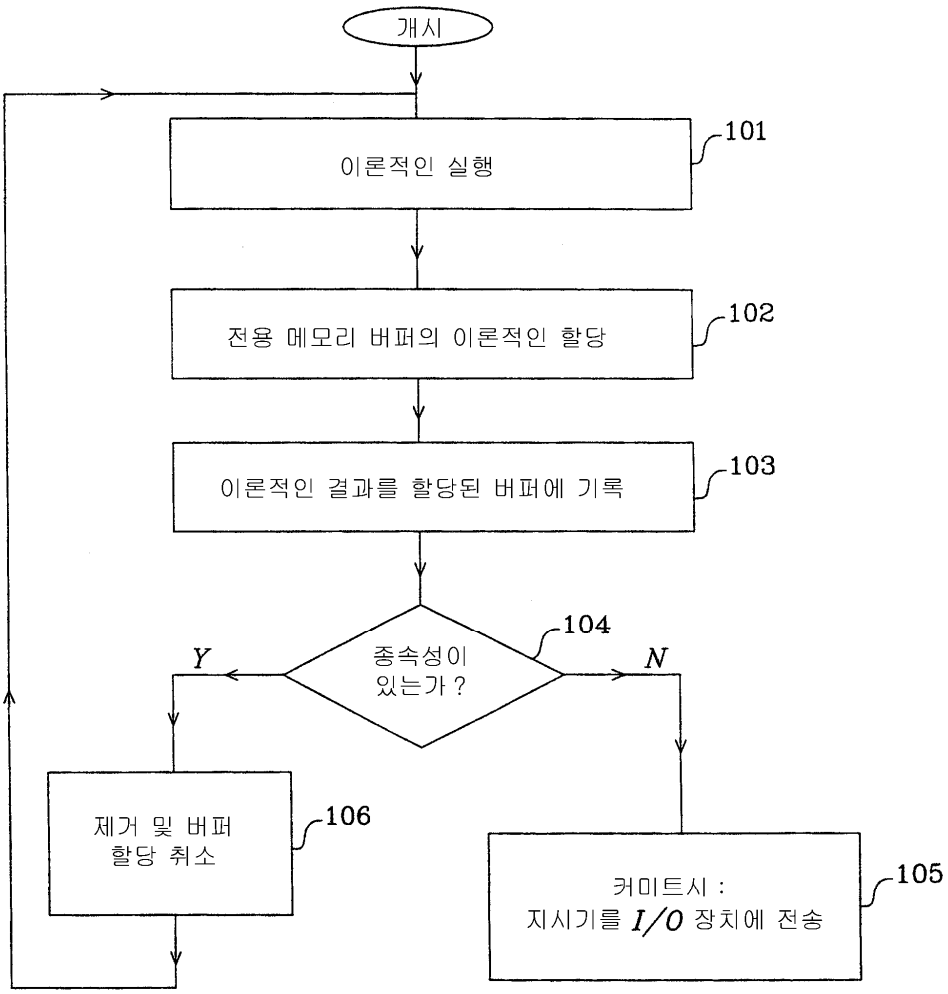
청구항 26.
제 24 항에 있어서,
추론적으로 실행되는 작업 및 인도 우선 순위를 갖는 작업 사이의 종속성을 검출하는 수단(35)과;
검출된 종속성에 응답하여 추론적으로 할당된 메모리 버퍼를 할당 취소시키는 수단을 더 포함하는 것을 특징으로 하는 처리 시스템.

도면

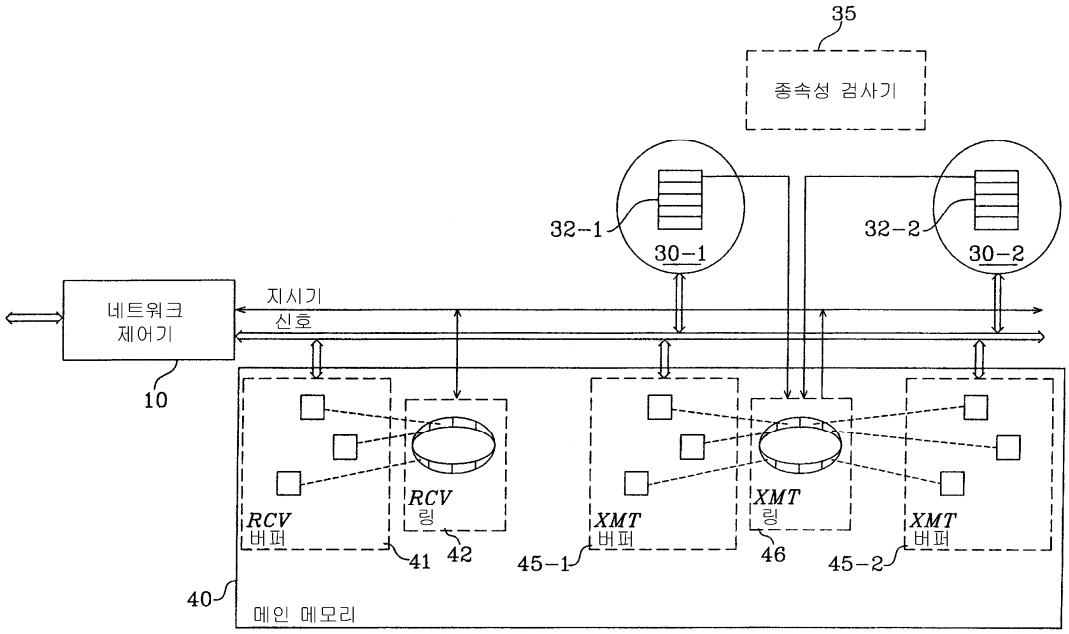
도면1


작업 분배기
20
30-1
30-2
30-N
종속성 검사기
35
I/O
장치
10
지시기
신호
45-1
45-2
45-N
버퍼
버퍼
버퍼
40
메인 메모리

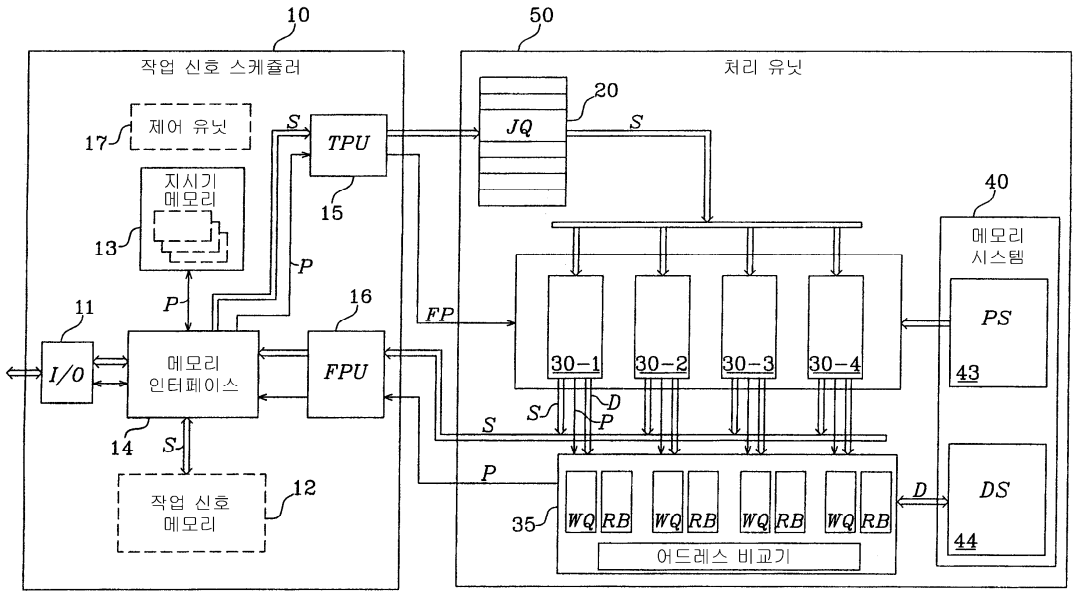
도면2


개시
이론적인 실행
101
전용 메모리 버퍼의 이론적인 할당
102
이론적인 결과를 할당된 버퍼에 기록
103
종속성이
있는가?
104
Y
N
제거 및 버퍼
할당 취소
106
커미트시 :
지시기를 I/O 장치에 전송
105

도면3


35
종속성 검사기
32-1
30-1
32-2
30-2
네트워크
제어기
지시기
신호
10
RCV
버퍼
RCV
링
41
42
XMT
버퍼
45-1
XMT
링
46
XMT
버퍼
45-2
40
메인 메모리

도면4


10
작업 신호 스케쥴러
50
처리 유닛
17
제어 유닛
S
TPU
15
JQ
20
S
지시기
메모리
13
P
P
11
16
FP
40
메모리
시스템
I/O
메모리
인터페이스
FPU
30-1
30-2
30-3
30-4
PS
43
14
S
S
D
P
P
35
WQ
RB
WQ
RB
WQ
RB
WQ
RB
D
DS
44
어드레스 비교기
작업 신호
메모리
12