(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) Int. Cl.[6]
G06F 11/28

(45) 공고일자 2000년03월15일
(11) 등록번호 10-0248376
(24) 등록일자 1999년12월17일

(21) 출원번호 10-1997-0055643
(22) 출원일자 1997년10월28일
(65) 공개번호 특1999-0034145
(43) 공개일자 1999년05월15일

(73) 특허권자 한국전자통신연구원 정선종
대전광역시 유성구 가정동 161번지
(72) 발명자 온기원
대전광역시 유성구 전민동 나래아파트 107동 903호
지동해
대전광역시 유성구 어은동 99번지 한빛아파트 122동 1503호
이범식
대전광역시 서구 탄방동 산호아파트 101동 1106호
박치항
대전광역시 유성구 어은동 99번지 한빛아파트 131동 1002호
(74) 대리인 김명섭, 이화익

심사관 : 민혜정

(54) 동적-비주얼 통합 병렬 디버깅 장치 및 디버깅 방법

요약

본 발명은 고속병렬 컴퓨터(High Speed Parallel Computer, 이하 SPAX라고 약칭함) 상에서 수행되는 병렬 프로그램을 디버깅하는 동적-비주얼 통합 병렬 디버깅 장치 및 디버깅 방법에 관한 것이다. 그 목적은 동적 디버깅과 비주얼 디버깅이 동시에 가능한 '동적-비주얼 통합 디버깅 환경'을 구축하는 데에 있다. 그 특징은 프로그램을 입력으로 받아서 참조실행을 통하여 프로그램 심볼 테이블 정보와 실행 로그파일을 생성하는 재실행 구동수단과, 상기 재실행 구동수단으로부터 상기 프로그램 심볼 테이블 정보와 상기 실행 로그파일을 입력받아 뷰와 사건을 관리하는 병렬 디버거 코어 및 상기 병렬 디버거 코어와 사용자를 인터페이스하는 그래픽 사용자 인터페이스 수단으로 구성되는 데에 있다.

대표도

도3

명세서

도면의 간단한 설명

도 1은 본 발명이 적용되는 고속병렬 컴퓨터의 하드웨어 구성도.

도 2는 고속병렬 컴퓨터, 병렬 프로그램, 통합 병렬 디버거 사이의 인터페이스를 나타낸 도면.

도 3은 동적-비주얼 통합 병렬 디버거의 구성도.

도 4는 동적-비주얼 통합 병렬 디버깅의 흐름도.

도 5a와 도 5b는 동적-비주얼 통합 병렬 디버거 및 텍스트/그래픽 뷰 매퍼의 제어 흐름도.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술분야 및 그 분야의 종래기술

본 발명은 고속병렬 컴퓨터(High Speed Parallel Computer, 이하 SPAX라고 약칭함) 상에서 수행되는 병렬 프로그램을 디버깅하는 동적-비주얼 통합 병렬 디버깅 장치 및 디버깅 방법에 관한 것이다.

일반적으로, 프로그램 개발환경 중에서 특히 병렬 프로그램 디버깅 분야에서 종래의 프로그램 디버깅 기술로 정적 디버깅과 동적 디버깅 방법이 있었고, 특히 동적 디버깅 방법 중에서 순환 디버깅 방법이 대표적으로 이용되어 왔다. 그러나, 순환 디버깅 방법은 프로그램의 실행경로가 단일 경로로써 보장되는 순차 프로그램의 디버깅에 적합한 것으로서, 단일 실행경로가 보장되지 않는 병렬 프로그램의 디버깅에 적용하

기에는 한계가 있다는 문제점이 있었다.

발명이 이루고자 하는 기술적 과제

상기 문제점을 해소하기 위한 본 발명은 SPAX 상에서 수행되는 병렬 프로그램을 디버깅하는 과정에서, 실행경로 보장을 통한 동적 순환 디버깅 기능을 제공하고, 디버깅 결과를 텍스트 형태로 보여주며, 디버깅되는 프로그램의 실행 흐름을 가시적 뷰로 보여주고, 병렬 프로그램에서 오류발생이 예상되는 실행 지점에 대한 뷰 매핑을 자동적으로 해줌으로써, 오류발생이 예상되는 스레드 사이의 공유 메모리 접근이나 프로세스 사이의 MPI 통신부분에 대한 집중적인 오류 디버깅 기능을 제공하는 통합된 동적-비주얼 디버깅 방법과 이를 기반으로 하는 병렬 디버거를 제공하는 데에 목적이 있다.

병렬 프로그램을 최소의 시간에 정확하게 디버깅하기 위해서는 오류가 발생했던 실행경로의 재실행이 보장되어야 하고, 동시에 오류의 원인이 될 만한 지점 즉, 프로세스(스레드) 사이의 통신 또는 공유 메모리 접근 등에 대한 디버깅 결과를 다양한 형태의 가시적 뷰로 분석하고 볼 수 있어야 한다. 이는 기술적인 측면에서 볼 때, 디버깅 하고자 하는 프로그램의 실행을 로그파일 형태로 기록할 수 있어야 하고, 이 로그파일을 기반으로 프로그램 전반에 걸친 재실행과 각 디버깅 지점에 대한 그래픽 뷰 디스플레이가 동시에 이루어져야 한다. 이를 위해서는, 동적 디버깅을 하는 중에 텍스트 값으로 보여지는 각각의 프로그램 변수 및 통신 함수들이, 가시적으로 디스플레이되고 있는 그래픽 뷰에서, 어느 지점에 해당하는지를 디버거가 자동적으로 매핑해 주어야 한다.

따라서, 본 발명은 텍스트 뷰와 그래픽 뷰로 디스플레이되는 디버깅 결과를 실행 로그파일을 기반으로 정확하게 찾아서 텍스트/그래픽 뷰 매핑을 시킴으로써, 동적 디버깅과 비주얼 디버깅이 동시에 가능한 '동적-비주얼 통합 디버깅 환경'을 구축하는 데에 그 목적이 있는 것이다.

상기 목적을 달성하기 위한 본 발명의 특징은 프로그램을 입력으로 받아서 참조실행을 통하여 프로그램 심볼 테이블 정보와 실행 로그파일을 생성하는 재실행 구동수단과, 상기 재실행 구동수단으로부터 상기 프로그램 심볼 테이블 정보와 상기 실행 로그파일을 입력받아 뷰와 사건을 관리하는 병렬 디버거 코어 및 상기 병렬 디버거 코어와 사용자를 인터페이스하는 그래픽 사용자 인터페이스 수단으로 구성되는 데에 있다. 상기 재실행 구동수단은 다중 스레드형 프로그램 또는 MPI 형 프로그램을 입력으로 받고, 상기 추적 대상 사건을 정의하기 위하여 사건추적 라이브러리를 이용하며, 상기 입력 프로그램에 따라 스레드 재실행 모듈이나 MPI 재실행 모듈을 호출하여 수행한다. 또한, 상기 재실행 구동수단은, 원래의 프로그램이 참조실행 시에 생성된 실행 로그파일을 참조함으로써 참조실행 시와 동일한 실행순서를 따르도록 하기 위하여 재실행 라이브러리를 이용한다. 상기 병렬 디버거 코어는 디버깅 결과가 디스플레이되는 윈도우를 관리하는 뷰 관리기와, 디버깅되는 프로그램의 실행을 제어하는 제어기와, 디버깅 사건들을 관리하는 사건 관리기를 포함한다. 상기 그래픽 사용자 인터페이스 수단은, 디버깅 결과를 텍스트로 디스플레이하고, 디버깅 사건 및 제어 기능을 지원하기 위한 메뉴와 다이얼로그, 그리고 이들에 해당하는 함수 콜백 기능을 제공하는 소스 텍스트 뷰 브라우저와, 디버깅되는 프로그램의 실행에 대한 그래픽 뷰를 지원하고, 뷰 에니메이션에 필요한 제어기능을 제공하는 그래픽 뷰 브라우저 및 텍스트 뷰와 그래픽 뷰로 디스플레이되는 임의의 사건에 대한 매핑 기능을 제공하는 텍스트/그래픽 뷰 매퍼를 포함한다.

상기 목적을 달성하기 위한 본 발명의 다른 특징은 텍스트형 뷰 윈도우에서 동적 순환 디버깅을 수행하면서 비주얼 디버깅을 하기 위한 정보를 입력하고, 텍스트/그래픽 매핑을 의뢰하는 단계와, 매핑을 의뢰한 다이얼로그와 그래픽 뷰 브라우저에게 실행 로그파일에서 매핑대상 사건을 탐색한 결과를 통보하는 단계와, 디버깅되는 프로그램의 실행 흐름을 디스플레이 하는 단계와, 현재 디스플레이 되고 있는 프로그램의 실행 흐름을 반복적으로 에니매이션 하는 비주얼 디버깅 단계와, 비주얼 디버깅의 진행 여부에 관계없이, 디버깅 사건으로 인하여 프로그램 실행이 멈춘 지점에서의 실행 상태를 여러 텍스트형 뷰 윈도우에서 검사하는 단계 및 상기 검사단계와 동시에 텍스트/그래픽 뷰 매핑을 의뢰하는 단계로 이루어지는 데에 있다. 사용자가 병렬 디버거의 수행을 종료하거나 또는 오류를 찾기 전까지, 상기 단계들을 처음부터 끝까지 반복적으로 수행하면서 디버깅을 수행한다. 상기 텍스트/그래픽 매핑의뢰 단계에서는, 텍스트/그래픽 뷰 매핑 다이얼로그에서의 매핑대상 사건과, 상기 매핑대상 사건을 수행하는 프로세스 또는 스레드의 고유번호와, 상기 매핑대상 사건이 매핑 목적 지점까지 수행된 총 횟수 및 원하는 그래픽 뷰를 입력한다. 상기 디스플레이 단계는, 상기 매핑대상 사건이 실행 로그파일에 존재하여 그 결과를 통보받은 그래픽 뷰 브라우저가 초기화 되는 과정과, 찾은 매핑 사건이 위치한 지점까지 실행 로그파일을 읽는 과정 및 선택된 그래픽 뷰 윈도우 상에 디버깅되는 프로그램의 실행 흐름을 디스플레이 하는 과정으로 이루어진다. 상기 비주얼 디버깅 단계는 동적 디버깅과는 독립적으로 수행되며, 프로그램의 실행 흐름을 매핑 사건이 위치한 지점부터 실행 로그파일을 계속적으로 읽어서 에니메이션하고, 또한, 뷰 제어 기능을 이용한다. 상기 검사단계에서는 이미 화면에 나타난 텍스트 뷰 윈도우 상에서 동적 순환 디버깅을 반복한다.

발명의 구성 및 작용

이하, 첨부된 도면을 참조하여 본 발명에 따른 바람직한 실시예들 중의 하나를 상세하게 설명한다.

도 1은 본 발명이 적용되는 SPAX의 하드웨어 구성도이다. 도 1을 참조하여, 본 발명의 병렬 디버거와 디버깅 방법이 적용되는 SPAX의 하드웨어의 구성을 설명하면 다음과 같다.

SPAX는 SMP(Symmetric Multi-Processor) 구조를 갖는 한 프로세싱 노드(1)에 4개의 처리기(5)를 실장하고, UMA(Uniform Memory Access) 공유 메모리(4) 모델을 제공하는 클러스터형 병렬 컴퓨터이다. 한 클러스터는 4개의 프로세싱 노드로 구성되고, 노드간과 클러스터간에는 NORMA(NORemote Memory Access) 메모리 모델을 제공한다. 노드간과 클러스터간 통신을 위하여 계층 크로스바 연결망(2,6)을 제공하며, 여기에 입출력 노드(3)를 연결한다.

도 2는 SPAX, 병렬 프로그램, 통합 병렬 디버거 사이의 인터페이스를 나타낸 도면이다. 도 2를 참조하여, SPAX, 병렬 프로그램, 통합 병렬 디버거 사이의 인터페이스를 설명하면 다음과 같다.

고속병렬 컴퓨터는 분산형 메모리(11)를 갖고 노드(10) 사이에는 고속의 네트워크(9)로 연결되어 있으며,

메시지 전송 라이브러리(Message-Passing Library)(8)를 제공함으로써, 프로세스(14) 또는 스레드(13) 사이의 통신을 지원한다. 병렬 프로그램(15)은 시스템에서 제공되는 프로세스 및 스레드 관리자와 스케줄러에 의하여 여러 개의 노드에 분산되어 병렬로 실행된다. 각각의 노드에는 하나 이상의 서브 프로그램(12)이 실행되며, 노드의 처리기(14)들은 스케줄러를 통하여 자신에게 할당된 서브 프로그램을 프로세스 및 스레드 단위로 수행한다. 하나의 서브 프로그램은 하나 이상의 프로세스로 구성되는데, 다중 스레드 형태의 프로그램인 경우, 하나의 동일한 프로세스 안에 여러 개의 스레드가 존재하고 이들은 들은 프로세스의 메모리(11)를 공유하는 방식으로 서로 간에 통신한다. 병렬 디버거(16)는 상기와 같이 분산되어 실행되는 병렬 프로그램의 제어권(17)을 가지고, 이 프로그램의 실행에 포함되는 모든 프로세스와 스레드들에 대한 오류 디버깅 작업을 수행한다. 병렬 디버거의 세부적인 디버깅 과정은 도 4에 상세히 나타나 있다.

도 3은 동적-비주얼 통합 병렬 디버거의 구성도이다. 도 3을 참조하여, 동적-비주얼 통합 병렬 디버거의 구성을 설명하면 다음과 같다.

동적-비주얼 통합 병렬 디버거는 크게 재실행 구동기(18), 디버거 코어(19), 그리고 그래픽 사용자 인터페이스(20)로 구성된다. 재실행 구동기(18)는 다중 스레드형 프로그램, 또는 MPI 형 프로그램을 입력으로 받아서, 참조실행을 통하여 프로그램 심볼 테이블 정보(21)와 실행 로그파일(22)을 생성한다. 참조실행은 프로그램의 실행경로를 단일화하기 위하여 필요한 것으로서, 이를 위하여 즉, 추적 대상 사건을 정의하기 위하여 사건 추적 라이브러리(23)를 이용하며, 입력 프로그램에 따라서, 스레드 재실행 모듈(24)이나 또는 MPI 재실행 모듈(25)이 호출되어 수행된다. 프로그램의 재실행은 원래의 프로그램이 참조실행 시에 생성된 실행 로그파일을 참조함으로써, 참조실행 시와 동일한 실행순서를 따르도록 하는 것이며, 이를 위하여 재실행 라이브러리(26)를 이용한다. 병렬 디버거 코어는 뷰 관리기(27)와 제어기(28), 그리고 사건 관리기(29) 등을 가진다. 뷰 관리기는 디버깅 결과가 디스플레이되는 윈도우에 대한 관리기능(윈도우 프레임, 메뉴, 서브 윈도우 구성, 메뉴버튼 등록 등)을 제공한다. 제어기는 디버깅되는 프로그램의 실행을 제어하는 기능(run, stop, step 등)을 제공한다. 사건 관리기는 디버깅 사건(breakpoint, watchpoint, stop on function 등)들을 관리하는 기능을 제공한다. 그래픽 사용자 인터페이스는 소스 텍스트 뷰 브라우저(32)와 그래픽 뷰 브라우저(30), 그리고 텍스트/그래픽 뷰 매퍼(31)로 구성된다. 소스 뷰 브라우저는 디버깅 결과를 텍스트로 디스플레이하고, 디버깅 사건 및 제어 기능을 지원하기 위한 메뉴와 다이얼로그, 그리고 이들에 해당하는 함수 콜백 기능을 제공한다. 그래픽 뷰 브라우저는 디버깅되는 프로그램의 실행에 대한 그래픽 뷰(스레드 실행 뷰, MPI 통신 뷰)를 지원하고, 뷰 에니메이션에 필요한 제어기능(stop, pause, run, reset)을 제공한다. 한편, 텍스트/그래픽 뷰 매퍼는 텍스트 뷰와 그래픽 뷰로 디스플레이 되는 임의의 사건(프로그램 문: 스레드 생성, 스레드 종료, 함수 호출, MPI 송/수신 등)에 대한 매핑 기능을 제공한다.

도 4는 동적-비주얼 통합 병렬 디버깅의 흐름도이다. 도 4를 참조하여 다중 스레드 형태나 또는 MPI 라이브러리를 이용하여 작성된 병렬 프로그램을 디버깅 하는 과정과 각각의 과정에 참여하는 디버거의 요소들을 설명하면 다음과 같다.

동적-비주얼 통합 병렬 디버거를 이용하는 프로그램 디버깅은 두 단계로 이루어진다. 제 1 단계는 참조실행 단계(33)로써, 먼저 디버깅 대상 소스 프로그램(34)에서 추적 대상이 되는 사건에 추적에 필요한 루틴을 씌운다. 이 루틴은 사건 추적 라이브러리(35)에서 제공된다. 수정된 프로그램은 컴파일러(36)와 라이브러리 링커(37)를 통하여 실행추적 파일기록용 실행파일을 가지고, 이 파일을 SPAX(38) 상에서 수행하면 실행 로그파일(39)이 생성된다. 제 2단계는 재실행 단계(40)로서, 디버깅 대상 소스 프로그램에서 추적대상이 되었던 사건마다 실행 로그파일을 참조하는 루틴을 씌우고, 이를 재실행 구동기(41)의 제어 하에서 컴파일하고, 재실행 라이브러리(42)가 링크되도록 라이브러리 링커를 수행하면, 재실행용 실행파일(43)이 생성된다. 병렬 디버거에서 동적 디버깅은, 재실행용 실행파일을 병렬 디버거 코어(44)의 제어 하에 SPAX 상에서 수행하면서, 프로세스 상태, 변수값, 레지스터 값 등을 심볼 테이블(45)에서 검사함으로써 이루어진다. 물론, 이 결과는 소스 뷰 브라우저(46)에 의하여 텍스트형 뷰 윈도우(47)에 디스플레이되며, 이 윈도우 상에서 정지점(Breakpoint) 또는 검사점(Watchpoint) 설정, 프로그램 실행, 심볼 값 검사 과정 등을 순환적으로 이용하면서, 동적 순환 디버깅을 수행한다. 한편, 동적 디버깅을 수행하면서, 오류발생 예상 지점 즉, 프로그램 문을 대상으로 텍스트/그래픽 뷰 매퍼(48)를 수행한다. 이를 위하여 텍스트형 뷰 윈도우의 하나인 소스 뷰 윈도우 상에서, 텍스트/그래픽 뷰 매핑 다이얼로그를 호출하고, 매핑 대상 사건과 이 사건을 실행하고 있는 프로세스 번호(스레드형 프로그램일 경우에는 스레드 번호) 그리고, 이 사건이 현재 호출된 횟수, 원하는 그래픽 뷰 등을 입력한 후에, 매핑시작 버튼을 누르면 매핑이 수행된다. 한편, 매핑대상 사건이 실행 로그파일에 존재하는 것이 확인되면, 그래픽 뷰 브라우저(49)는 선택된 그래픽 뷰(50) 상에서 현재의 매핑사건이 있는 지점까지 수행된 프로그램을 그래픽하게 디스플레이된다. 사용자는 이제 그래픽 뷰 상에서 프로그램의 실행을 에니메이션하거나 또는 텍스트형 뷰 윈도우에서 동적 디버깅을 계속 할 수 있다.

도 5a와 도 5b는 동적-비주얼 통합 병렬 디버거 및 텍스트/그래픽 뷰 매퍼의 제어 흐름도이다. 도 5a와 도 5b를 참조하여 동적-비주얼 통합 병렬 디버거 및 텍스트/그래픽 뷰 매퍼의 제어흐름을 설명하면 다음과 같다.

명령어 프롬프트에서 동적-비주얼 통합 병렬 디버거(ParaDebug)를 호출하면, S1에서 텍스트형 소스 뷰 윈도우가 수행되면서 화면에 나타난다. S2에서는 S1의 파일(file) 메뉴에서 소스 프로그램 생성 다이얼로그를 통하여, 디버깅 대상 프로그램을 호출한다. S3에서는 프로그램 디버깅에 필요한 디버깅 사건(예: 정지점, 검사점 등)을 설정한다. S4에서는 디버깅 사건이 설정된 지점까지 프로그램을 실행한다. S5에서는 디버깅 사건이 설정된 지점에서, 프로그램의 실행정보(예: 프로세스 및 스레드 실행 상태, 변수의 심볼 값 및 레지스터 값 등)를 검사한다. S3에서 S5 과정은 동적 디버깅을 의미한다. S6에서는 동적 디버깅을 종료할지를 결정한다. 종료하게 되면, 병렬 디버거의 실행이 모두 완료됨을 의미한다. 한편, S7에서는 디버깅을 종료하지 않고, 동적 디버깅을 계속하거나 또는 텍스트/그래픽 뷰 매핑을 이용할지를 결정한다. S7에서 S3으로의 흐름은 동적 디버깅의 반복을 의미한다. 텍스트/그래픽 뷰 매핑의 이용은 S8과 같이 매핑 다이얼로그의 호출로부터 시작된다. S9에서는 매핑 다이얼로그 상에서 매핑대상 사건, 이 사건을 수행하는 프로세스(또는 다중 스레드형 프로그램의 경우, 스레드) 고유번호, 이 사건이 현재까지 실행된 총 횟수, 원하는 그래픽 뷰 윈도우 (MPI/시간 뷰 또는 스레드/시간 뷰) 등을 입력한다. S10에서는 병렬 디버거

호출 전에 이미 재실행 과정에서 생성된 실행 로그파일 안에 매핑대상 사건이 존재하는지 탐색한다. S11에서는 사건이 로그파일 안에 존재하지 않는 경우에는, 매핑대상 사건을 새로이 선택할 수 있도록 S9로 돌아간다. 매핑 대상 사건이 실행 로그파일 안에 존재하는 경우에는, S12에서 그래픽 뷰 브라우저가 호출되고, S13에서는 매핑대상 사건의 종류를 판단하고 그래픽 뷰를 선택한다. 상기 S13에서 MPI/시간 뷰가 선택되면, S14에서는 MPI/시간 뷰가 디스플레이된다. 뷰 디스플레이는 매핑대상 사건이 뷰의 중앙에 위치할 때까지 이루어진다. S15에서는 디버깅 모드를 선택한다. 동적 디버깅을 계속할 경우, S3으로 간다. MPI/시간 뷰 에니매이션을 이용할 경우, S16에서는 MPI/시간 뷰 에니매이션을 실행한다. MPI/시간 뷰 에니매이션이란, 실행 로그파일을 바탕으로, S14로부터 뷰 디스플레이를 계속적으로 실행함을 의미한다. 또한, S17에서는 재생(play), 일시멈춤(pause), 단계별 실행(step), 정지(stop) 등의 뷰 제어 기능을 적용할 수 있다. S18에서는 그래픽 뷰 에니매이션의 종료를 결정한다. 종료를 결정하면, 그래픽 뷰 브라우저의 실행이 종료된다. 그러나, 사용자는 S16~S18를 반복적으로 수행하면서 비주얼 디버깅을 계속 할 수 있다. 만일, 상기 S13에서 스레드/시간 뷰가 선택되면, S19에서는 쓰레드/시간 뷰가 디스플레이된다. 뷰 디스플레이는 매핑대상 사건이 뷰의 중앙에 위치할 때까지 이루어진다. S20에서는 디버깅 모드를 선택한다. 동적 디버깅을 계속할 경우, S3으로 간다. 쓰레드/시간 뷰 에니매이션을 이용할 경우, S21에서는 뷰 에니매이션을 실행한다. 쓰레드/시간 뷰 에니매이션이란, 실행 로그파일을 바탕으로, S19로부터 뷰 디스플레이를 계속적으로 실행함을 의미한다. 또한, S21에서는 재생(play), 일시멈춤(pause), 단계별 실행(step), 정지(stop) 등의 뷰 제어 기능을 적용할 수 있다. S22에서는 그래픽 뷰 에니매이션의 종료를 결정한다. 종료를 결정하면, 그래픽 뷰 브라우저의 실행이 종료된다. 그러나, 사용자는 S21~S23을 반복적으로 수행하면서 비주얼 디버깅을 계속 할 수 있다.

발명의 효과

본 디버깅 방법의 효과는 다음의 3가지로 요약 할 수 있다. 첫째는 병렬 컴퓨터 상에서 실행되는 임의의 병렬 프로그램에 대하여, 프로그램의 참조실행을 통하여 실행 경로를 로그파일에 저장하고, 이를 가시적인 뷰로 보여줌으로써, 병렬 프로그램에 내포된 비결정적 실행 특성, 즉 실행 경로의 다중성을 해결하고 결정적 실행 경로를 보장하는 것이다. 둘째는, 결정적 실행 경로를 가지는 프로그램 즉, 순차 프로그램의 디버깅에 대표적으로 이용되어온 동적 디버깅 방법을 그대로 제공함으로써, 새로운 디버깅 방법을 공부하지 않고서도 병렬 프로그램을 디버깅 할 수 있다는 것이다. 셋째로, 텍스트 형태로만 제공되어오던 동적 순환 디버깅 결과를 그래픽 뷰로 가시화시켜서 보여줌으로써, 병렬 프로그램의 실행에 대한 이해를 돕고, 오류 추적에 소요되는 시간을 최소화시킨다는 것이다. 마지막으로, 이러한 '동적-비주얼 통합 디버깅 환경'은 특히 대규모의 복잡한 병렬 프로그램 개발에 적합한 새로운 개념의 프로그램 개발 환경을 제공하는 데에 그 효과가 있다.

(57) 청구의 범위

청구항 1

프로그램을 입력으로 받아서 참조실행을 통하여 프로그램 심볼 테이블 정보와 실행 로그파일을 생성하는 재실행 구동수단;

상기 재실행 구동수단으로부터 상기 프로그램 심볼 테이블 정보와 상기 실행 로그파일을 입력받아 뷰와 사건을 관리하는 병렬 디버거 코어; 및

상기 병렬 디버거 코어와 사용자를 인터페이스하는 그래픽 사용자 인터페이스 수단으로 구성되는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 2

제 1 항에 있어서,

상기 재실행 구동수단이 다중 스레드형 프로그램 또는 MPI 형 프로그램을 입력으로 받는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 3

제 1 항에 있어서,

상기 재실행 구동수단이 상기 추적대상 사건을 정의하기 위하여 사건추적 라이브러리를 이용하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 4

제 1 항에 있어서,

상기 재실행 구동수단이 상기 입력 프로그램에 따라 스레드 재실행 모듈이나 MPI 재실행 모듈을 호출하여 수행하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 5

제 1 항에 있어서,

상기 재실행 구동수단이, 원래의 프로그램이 참조실행 시에 생성된 실행 로그파일을 참조함으로써 참조실행 시와 동일한 실행순서를 따르도록 하기 위하여 재실행 라이브러리를 이용하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 6

제 1 항에 있어서,

상기 병렬 디버거 코어가,

디버깅 결과가 디스플레이되는 윈도우를 관리하는 뷰 관리기;

디버깅되는 프로그램의 실행을 제어하는 제어기; 및

디버깅 사건들을 관리하는 사건 관리기를 포함하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 7

제 1 항에 있어서,

상기 그래픽 사용자 인터페이스 수단이,

디버깅 결과를 텍스트로 디스플레이하고, 디버깅 사건 및 제어 기능을 지원하기 위한 메뉴와 다이얼로그, 그리고 이들에 해당하는 함수 콜백 기능을 제공하는 소스 텍스트 뷰 브라우저;

디버깅되는 프로그램의 실행에 대한 그래픽 뷰를 지원하고, 뷰 에니메이션에 필요한 제어기능을 제공하는 그래픽 뷰 브라우저; 및

텍스트 뷰와 그래픽 뷰로 디스플레이 되는 임의의 사건에 대한 매핑 기능을 제공하는 텍스트/그래픽 뷰 매퍼를 포함하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 장치.

청구항 8

텍스트형 뷰 윈도우에서 동적 순환 디버깅을 수행하면서 비주얼 디버깅을 하기 위한 정보를 입력하고, 텍스트/그래픽 매핑을 의뢰하는 단계;

매핑을 의뢰한 다이얼로그와 그래픽 뷰 브라우저에게 실행 로그파일에서 매핑대상 사건을 탐색한 결과를 통보하는 단계;

디버깅되는 프로그램의 실행 흐름을 디스플레이 하는 단계;

현재 디스플레이 되고 있는 프로그램의 실행 흐름을 반복적으로 에니매이션 하는 비주얼 디버깅 단계;

비주얼 디버깅의 진행 여부에 관계없이, 디버깅 사건으로 인하여 프로그램 실행이 멈춘 지점에서의 실행 상태를 여러 텍스트형 뷰 윈도우에서 검사하는 단계; 및

상기 검사단계와 동시에 텍스트/그래픽 뷰 매핑을 의뢰하는 단계로 이루어지는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 9

제 8 항에 있어서,

사용자가 병렬 디버거의 수행을 종료하거나 또는 오류를 찾기 전까지, 상기 단계들을 처음부터 끝까지 반복적으로 수행하면서 디버깅을 수행하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 10

제 8 항에 있어서,

상기 텍스트/그래픽 매핑의뢰 단계에서,

텍스트/그래픽 뷰 매핑 다이얼로그에서의 매핑대상 사건;

상기 매핑대상 사건을 수행하는 프로세스 또는 스레드의 고유번호;

상기 매핑대상 사건이 매핑 목적 지점까지 수행된 총 횟수; 및

원하는 그래픽 뷰를 입력하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 11

제 8 항에 있어서,

상기 디스플레이 단계가,

상기 매핑대상 사건이 실행 로그파일에 존재하여 그 결과를 통보받은 그래픽 뷰 브라우저가 초기화 되는 과정;

찾은 매핑 사건이 위치한 지점까지 실행 로그파일을 읽는 과정; 및

선택된 그래픽 뷰 윈도우 상에 디버깅되는 프로그램의 실행 흐름을 디스플레이 하는 과정으로 이루어지는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 12

제 8 항에 있어서,

상기 비주얼 디버깅 단계가 동적 디버깅과는 독립적으로 수행되는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 13

제 8 항에 있어서,

상기 비주얼 디버깅 단계에서, 상기 프로그램의 실행 흐름을 매핑 사건이 위치한 지점부터 실행 로그파일을 계속적으로 읽어서 에니메이션하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

청구항 14

제 8 항에 있어서,

상기 비주얼 디버깅 단계가 뷰 제어 기능을 이용하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

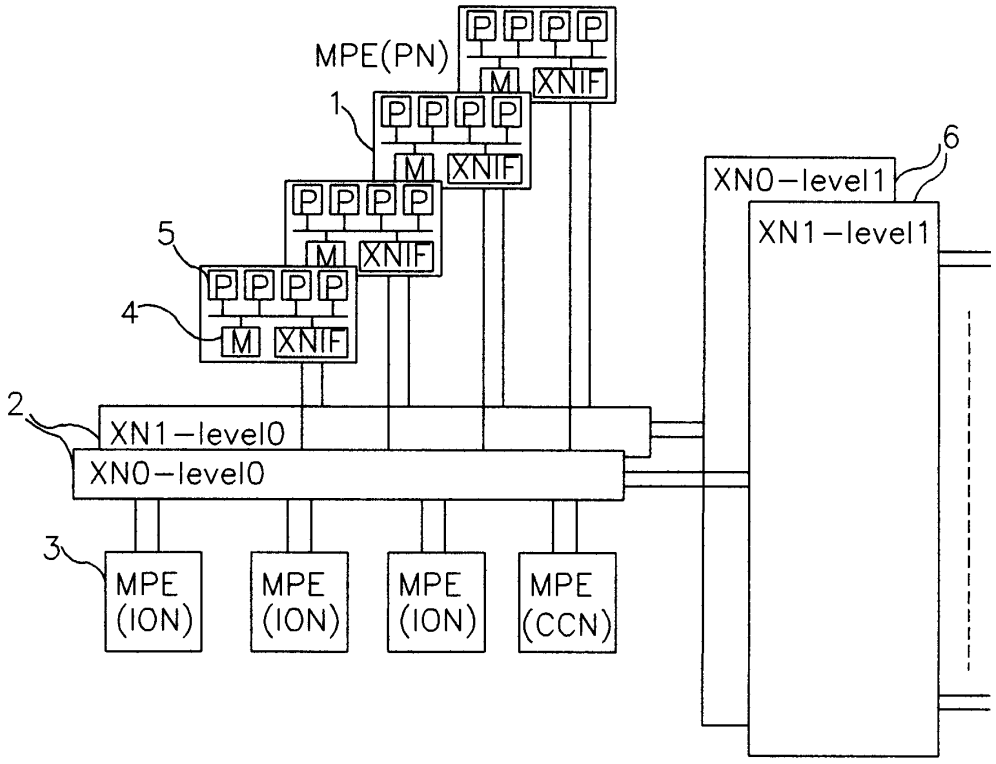
청구항 15

제 8 항에 있어서,

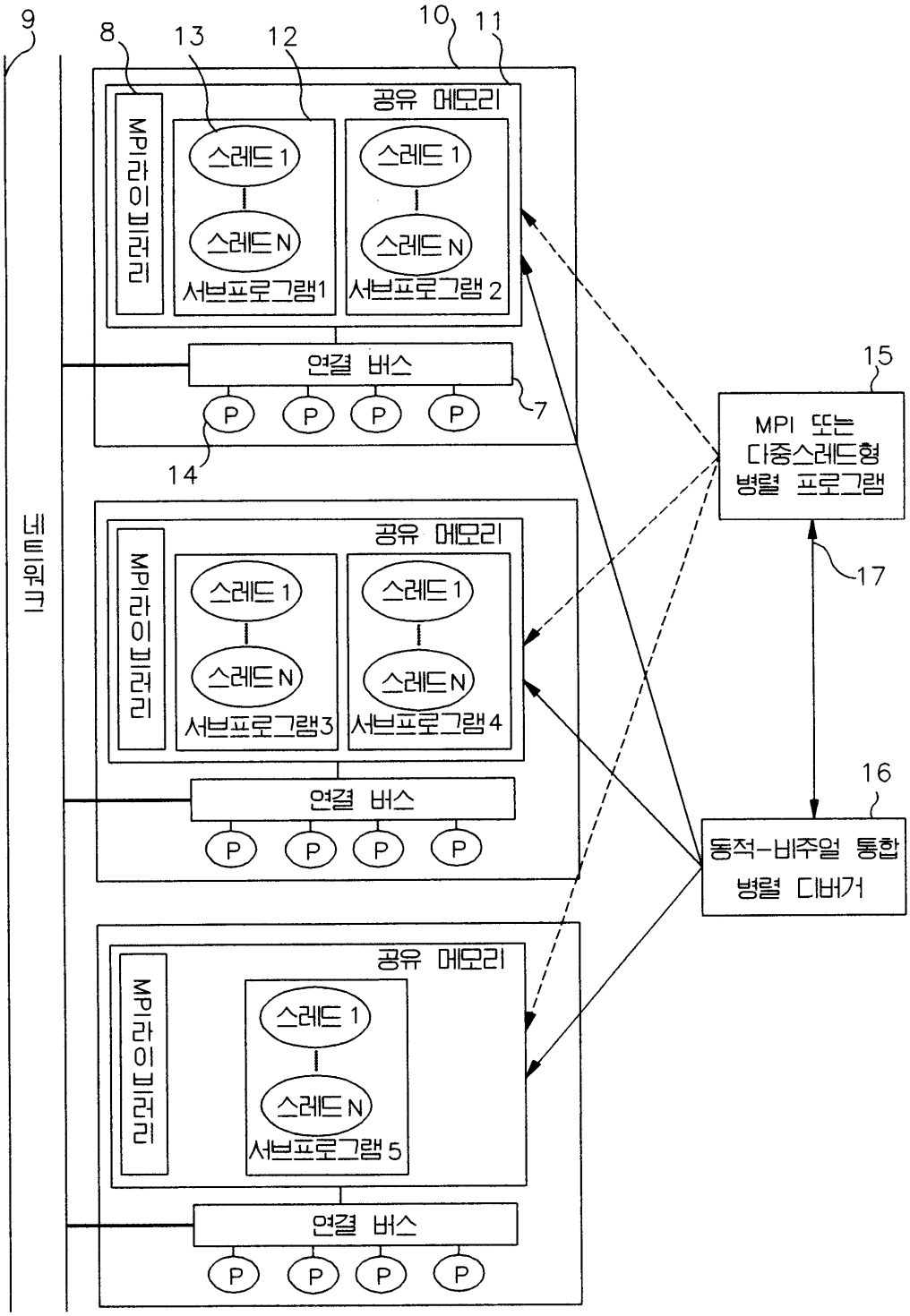
상기 검사단계에서 이미 화면에 나타난 텍스트 뷰 윈도우 상에서 동적 순환 디버깅을 반복하는 것을 특징으로 하는 동적-비주얼 통합 병렬 디버깅 방법.

도면

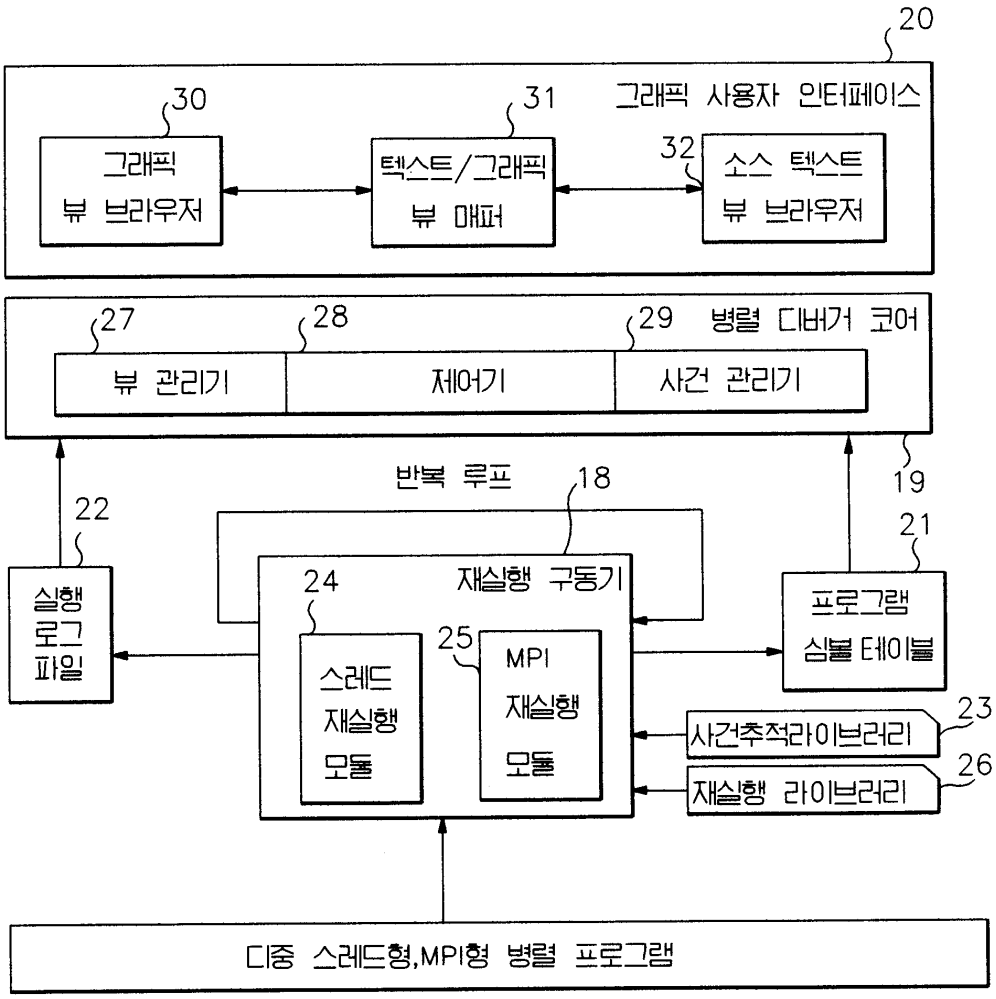
도면1



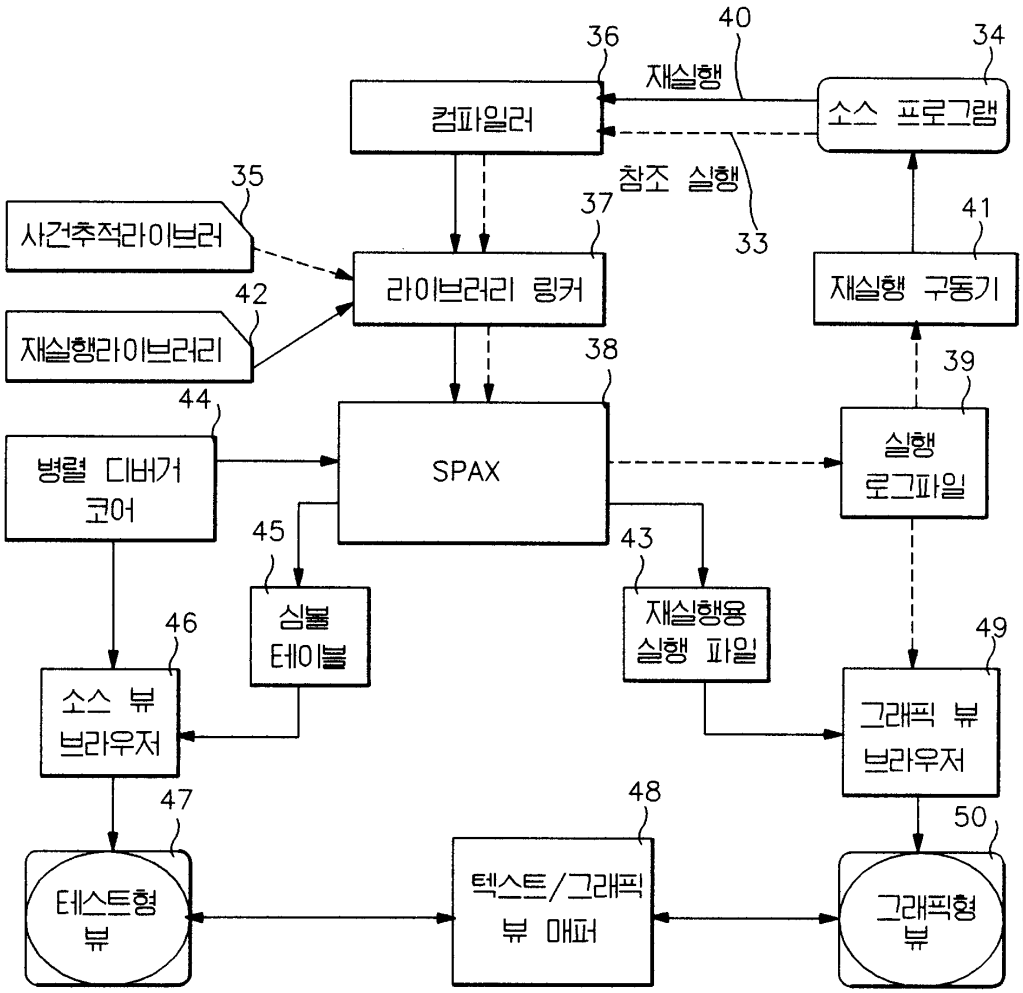
도면2


9
8
13
12
10
11
공유 메모리
MPI라이브러리
스레드 1
스레드 N
서브프로그램1
서브프로그램 2
연결 버스
P
7
14
15
MPI 또는
다중스레드형
병렬 프로그램
17
16
동적-비주얼 통합
병렬 디버거
네트워크
서브프로그램3
서브프로그램 4
서브프로그램 5

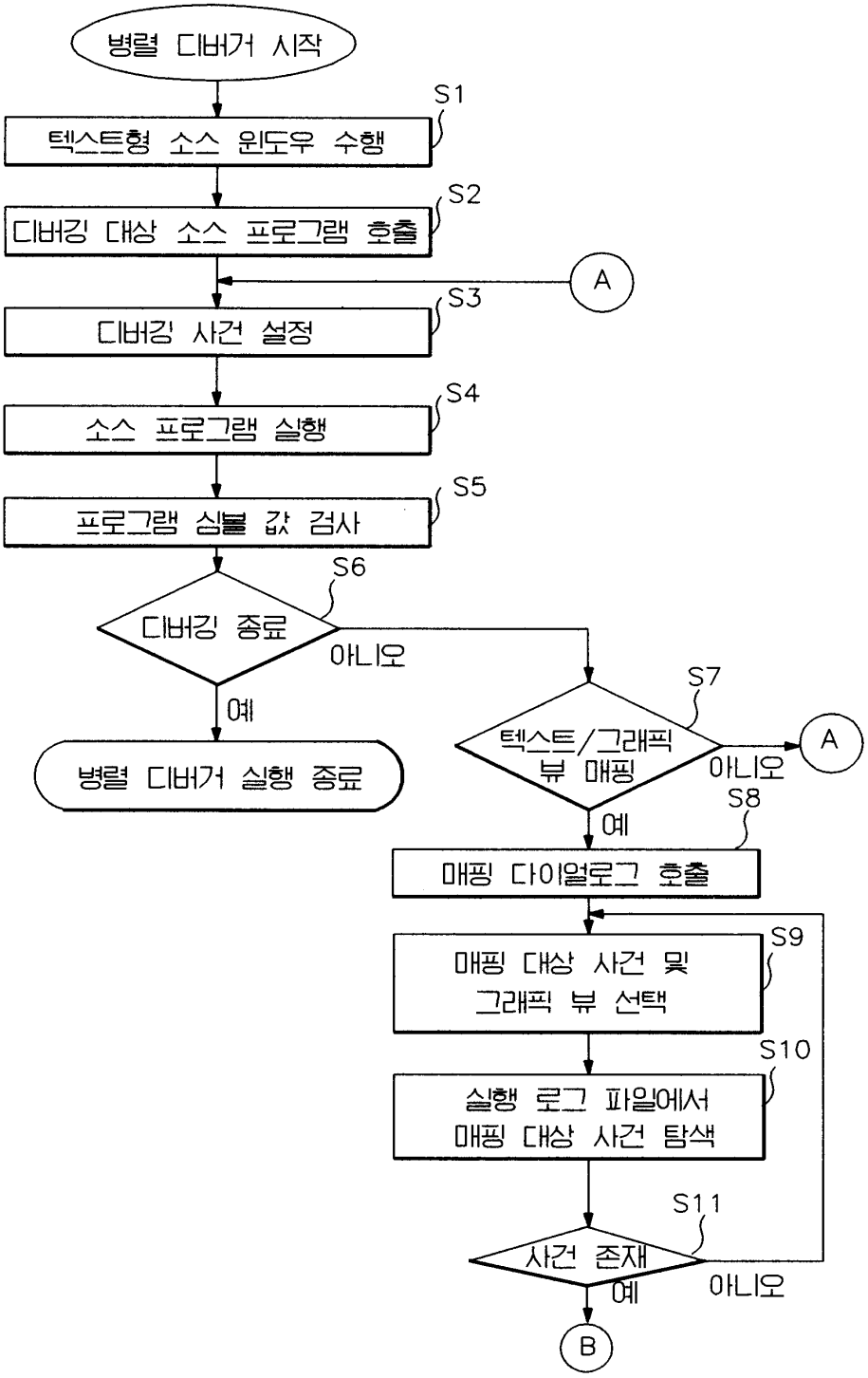
도면3


20
그래픽 사용자 인터페이스
30
그래픽
뷰 브라우저
31
텍스트/그래픽
뷰 매퍼
32
소스 텍스트
뷰 브라우저
27
28
29
병렬 디버거 코어
뷰 관리기
제어기
사건 관리기
반복 루프
18
19
22
21
실행
로그
파일
24
재실행 구동기
25
스레드
재실행
모듈
MPI
재실행
모듈
프로그램
심볼 테이블
23
사건추적라이브러리
26
재실행 라이브러리
디중 스레드형,MPI형 병렬 프로그램

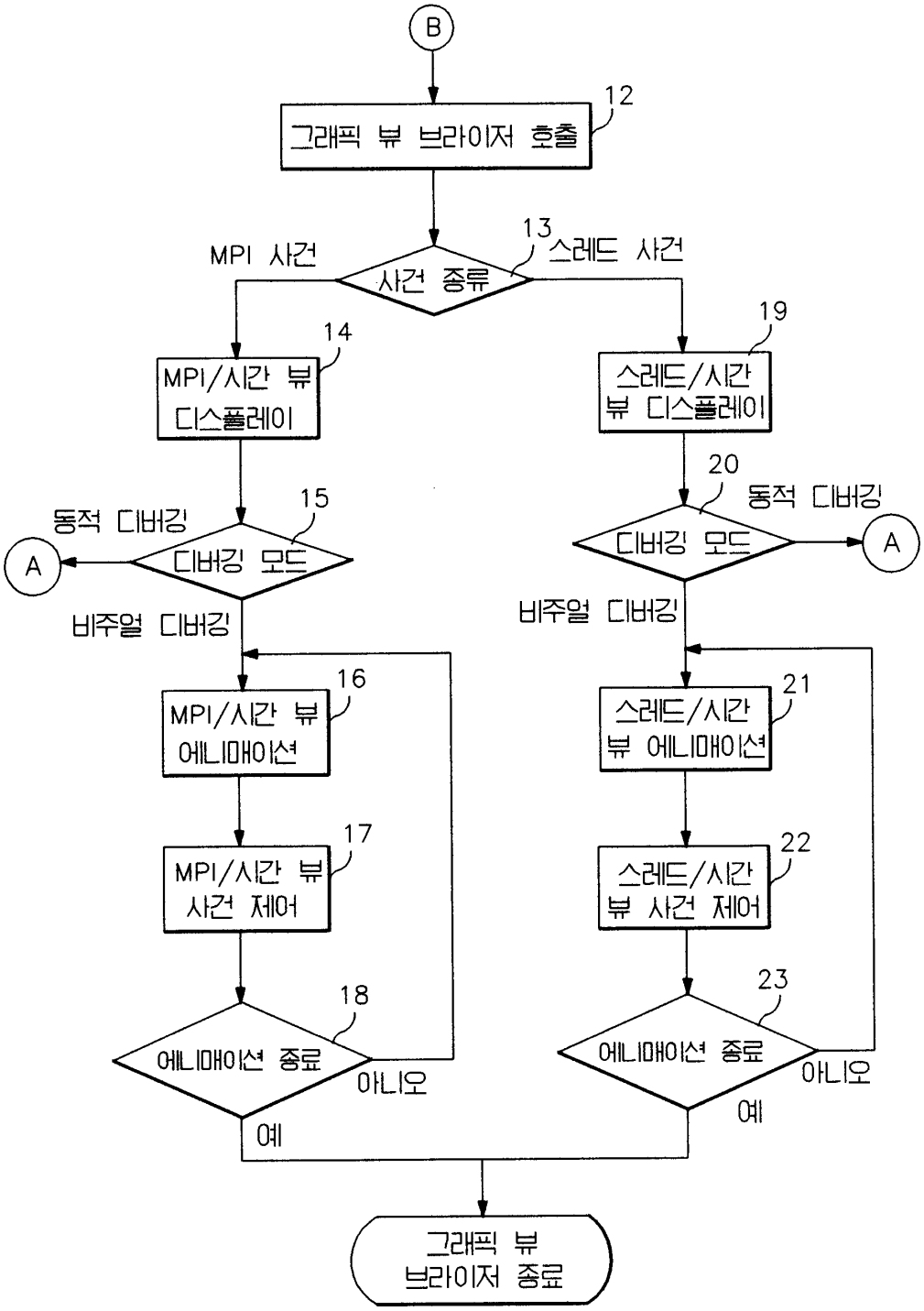
도면4


컴파일러
36
재실행
40
소스 프로그램
34
참조 실행
33
사건추적라이브러
35
라이브러리 링커
37
재실행 구동기
41
재실행라이브러리
42
SPAX
38
실행
로그파일
39
병렬 디버거
코어
44
심볼
테이블
45
재실행용
실행 파일
43
소스 뷰
브라우저
46
그래픽 뷰
브라우저
49
테스트형
뷰
47
텍스트/그래픽
뷰 매퍼
48
그래픽형
뷰
50

도면5a


병렬 디버거 시작
S1
텍스트형 소스 윈도우 수행
S2
디버깅 대상 소스 프로그램 호출
A
S3
디버깅 사건 설정
S4
소스 프로그램 실행
S5
프로그램 심볼 값 검사
S6
디버깅 종료
아니오
예
병렬 디버거 실행 종료
S7
텍스트/그래픽 뷰 매핑
아니오
A
예
S8
매핑 다이얼로그 호출
S9
매핑 대상 사건 및 그래픽 뷰 선택
S10
실행 로그 파일에서 매핑 대상 사건 탐색
S11
사건 존재
아니오
예
B

도면5b


B
12
그래픽 뷰 브라이저 호출
13
MPI 사건
사건 종류
스레드 사건
14
MPI/시간 뷰 디스플레이
19
스레드/시간 뷰 디스플레이
15
동적 디버깅
A
디버깅 모드
20
동적 디버깅
디버깅 모드
A
비주얼 디버깅
비주얼 디버깅
16
MPI/시간 뷰 에니메이션
21
스레드/시간 뷰 에니메이션
17
MPI/시간 뷰 사건 제어
22
스레드/시간 뷰 사건 제어
18
에니메이션 종료
아니오
예
23
에니메이션 종료
아니오
예
그래픽 뷰 브라이저 종료