



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년11월15일
(11) 등록번호 10-1918051
(24) 등록일자 2018년11월07일

(51) 국제특허분류(Int. Cl.)

G06F 17/50 (2006.01)

(52) CPC특허분류

G06F 17/5009 (2013.01)

(21) 출원번호 10-2017-0061141

(22) 출원일자 2017년05월17일

심사청구일자 2017년05월17일

(56) 선행기술조사문헌

KR1020060005286 A*

JP2015515673 A*

KR1020170045437 A

JP2012221254 A

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

에스케이하이닉스 주식회사

경기도 이천시 부발읍 경충대로 2091

고려대학교 산학협력단

서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)

(72) 발명자

김선욱

경기도 남양주시 도농로 34, 214동 2003호(도농동, 부영아파트)

김민성

서울특별시 성북구 개운사길 22-6, 402호

(뒷면에 계속)

(74) 대리인

특허법인 이노

전체 청구항 수 : 총 8 항

심사관 : 박승철

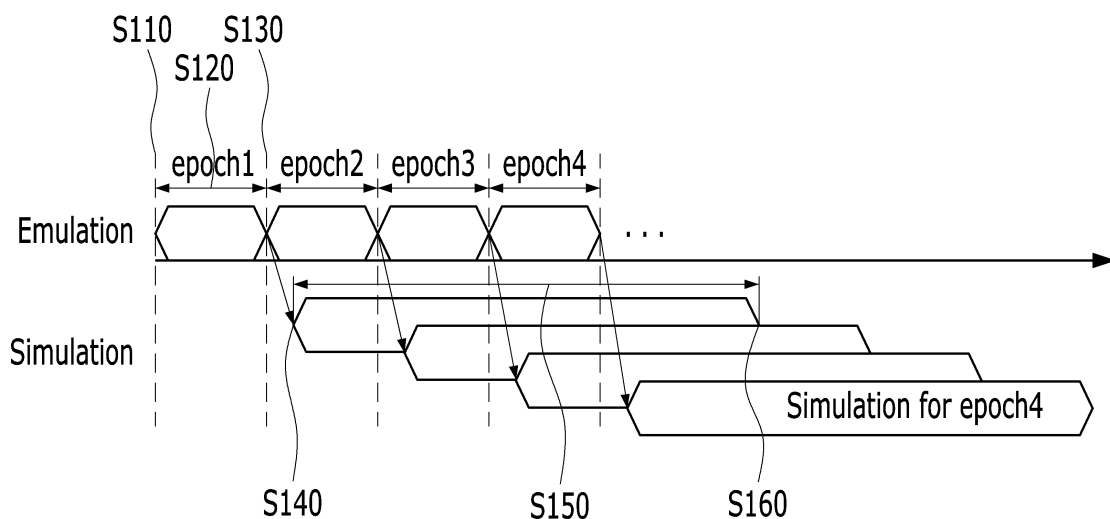
(54) 발명의 명칭 Epoch 기반의 시뮬레이션 방법

(57) 요약

본 발명은 전체 시뮬레이션 구간에 대하여 시뮬레이션 엔진이 에뮬레이션 엔진과 상호작용 없이 독립적으로 실행할 수 있는 단위 구간을 epoch로 구분하고, 각각의 epoch에 대하여 시뮬레이션 엔진을 병렬적으로 실행하는 epoch 기반의 시뮬레이션 방법을 제공한다.

(뒷면에 계속)

대표도 - 도1



본 발명의 일 태양에 따른 Epoch 기반의 시뮬레이션 방법은, 소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 제1 epoch - 상기 제1 epoch는 상기 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 의 시작점을 검출하는 단계; 중앙처리부는 상기 에뮬레이션 엔진의 동작 중에 live-in 데이터를 기록하는 단계; 상기 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면, 상기 중앙처리부가 상기 제1 epoch의 종료점에 도달한 것으로 판단하는 단계; 상기 제1 epoch의 종료점에 도달하면, 상기 에뮬레이션 엔진은 상기 live-in 데이터의 기록을 종료하는 단계; 상기 에뮬레이션 엔진은 시뮬레이션 엔진에 상기 live-in 데이터를 전달하는 단계; 및 상기 live-in 데이터에 기반하여 상기 제1 epoch에 대한 시뮬레이션을 실행하는 단계를 포함한다.

(72) 발명자

한미선

서울특별시 동대문구 무학로 193, 502호(삼영오피스텔)

박찬현

서울특별시 성북구 보문로22길 58, 406호

명세서

청구범위

청구항 1

소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 제1 epoch - 상기 제1 epoch는 상기 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 의 시작점을 검출하는 단계;

중앙처리부는 상기 에뮬레이션 엔진의 동작 중에 live-in 데이터를 기록하는 단계;

상기 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면, 상기 중앙처리부가 상기 제1 epoch의 종료점에 도달한 것으로 판단하는 단계;

상기 제1 epoch의 종료점에 도달하면, 상기 에뮬레이션 엔진에서 live-in 데이터의 기록을 중지하는 단계;

상기 에뮬레이션 엔진은 상기 시뮬레이션 엔진에 상기 live-in 데이터를 전달하는 단계; 및

상기 live-in 데이터에 기반하여 상기 제1 epoch에 대한 시뮬레이션을 실행하는 단계

를 포함하는 Epoch 기반의 시뮬레이션 방법.

청구항 2

제1항에 있어서,

상기 에뮬레이션 엔진은 상기 live-in 데이터 중 메모리 영역에 포함된 명령 및 데이터 자체를 상기 시뮬레이션 엔진에 전달하는 Epoch 기반의 시뮬레이션 방법.

청구항 3

제1항에 있어서,

상기 에뮬레이션 엔진은 상기 live-in 데이터 중 메모리 영역에 포함된 명령 및 데이터의 주소를 상기 시뮬레이션 엔진에 전달하는 Epoch 기반의 시뮬레이션 방법.

청구항 4

제1항에 있어서,

상기 epoch 탐색용 프로그램 코드는 interrupt 명령, exception 명령, syscall 명령 또는 이에 상응하는 명령인 것을 특징으로 하는 Epoch 기반의 시뮬레이션 방법.

청구항 5

소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 제1 epoch - 상기 제1 epoch는 상기 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 의 시작점을 검출하는 제1 단계;

중앙처리부가 상기 에뮬레이션 엔진을 실행하여 상기 제1 epoch에 대하여 live-in 데이터를 기록하는 제2 단계;

상기 에뮬레이션 엔진이 상기 live-in 데이터 중 상기 제1 epoch에 대하여 시뮬레이션 엔진의 기동에 필수적인 live-in 데이터를 상기 시뮬레이션 엔진에 전달하면 상기 시뮬레이션 엔진을 실행하는 제3 단계;

상기 중앙처리부는 상기 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면 상기 제1 epoch의 종료점에 도달한 것으로 판단하고, 상기 에뮬레이션 엔진에서 상기 live-in 데이터의 기록을 중지하는 제4

단계;

상기 제2 단계에서 기록되지 않은 상기 제1 epoch의 종료점 정보를 포함한 live-in 데이터를 상기 시뮬레이션 엔진에 전달하는 제5 단계; 및

상기 시뮬레이션 엔진은 전달된 상기 제1 epoch의 종료점 정보에 기반하여 상기 제1 epoch의 종료점까지 상기 시뮬레이션을 진행하는 제6 단계

를 포함하는 Epoch 기반의 시뮬레이션 방법.

청구항 6

제5항에 있어서,

상기 에뮬레이션 엔진은 상기 live-in 데이터 중 메모리 영역 정보 자체를 상기 시뮬레이션 엔진에 전달하는 Epoch 기반의 시뮬레이션 방법.

청구항 7

제5항에 있어서,

상기 에뮬레이션 엔진은 상기 live-in 데이터 중 메모리 영역 정보의 주소를 상기 시뮬레이션 엔진에 전달하는 Epoch 기반의 시뮬레이션 방법.

청구항 8

제5항에 있어서,

상기 제2 단계에서 기록되지 않은 상기 제1 epoch 종료점 정보를 포함한 상기 live-in 데이터는 실행된 프로그램 명령의 개수를 포함하는 Epoch 기반의 시뮬레이션 방법.

청구항 9

삭제

발명의 설명

기술 분야

[0001] 본 발명은 컴퓨터 시스템의 시뮬레이션 방법에 관한 것으로, 더욱 상세하게는 컴퓨터 시스템의 시뮬레이션을 Epoch에 기반하여 진행하는 방법에 관한 것이다.

배경 기술

[0003] 일반적으로 어플리케이션을 하드웨어에서 직접 실행하는 시간보다 시뮬레이션으로 실행하는 시간이 수십만배 이상 소요된다. 그렇기 때문에 시뮬레이션 시간이 긴 단점을 극복하기 위해 실행되는 코드의 유사성을 이용하는 방법, 샘플링 기술을 이용하는 방법 등이 제시되었다.

[0004] 예컨대, 논문 'Automatically Characterizing Large Scale Program Behavior'에서는 시뮬레이션 시간을 감소시키기 위해 코드의 유사성을 이용하는 방법을 사용한다. 즉, 시뮬레이션을 수행하기 위한 어플리케이션의 basic block의 instruction 및 control flow 정보를 이용하여 유사한 control flow를 가지는 일련의 basic block들을 분석하고, 이에 대하여 시뮬레이션을 진행하는 과정과 방법을 제안한다. 즉, 해당 논문에 따르면, 시뮬레이션을 진행하기 전에 어플리케이션에 대한 분석이 필수적이므로 여전히 시뮬레이션을 진행하기 위한 시간이 많이 소요된다.

[0005] 그리고, 논문 'Accelerating Microarchitecture Simulation via Rigorous Statistical Sampling'에서는 통계적 샘플링 기반 기술을 이용하여 시뮬레이션 결과를 짧은 시간 내에 얻는 방법이 개시된다. 즉, 통계적 샘플링 기반 기술을 통해 시뮬레이션 할 구간을 임의로 선정하고 임의 구간들에 대해 시뮬레이션을 진행한 후, 해당 시뮬레이션 결과를 취합하여 전체 시뮬레이션 결과를 얻는 방법을 제안한다. 이 또한, 시뮬레이션을 진행하기 전에 전체 시뮬레이션 구간 중 샘플링을 통해 시뮬레이션을 진행할 특정 구간을 정하여야 하므로, 여전히 시뮬레이션을 진행하는 시간이 많이 소요된다.

선행기술문헌

비특허문헌

[0007] (비특허문헌 0001) Automatically Characterizing Large Scale Program Behavior, Timothy Sherwood 외 3
(비특허문헌 0002) Accelerating Microarchitecture Simulation via Rigorous Statistical Sampling, Roland E. Wunderlich 외 3

발명의 내용

해결하려는 과제

[0008] 본 발명은 전체 시뮬레이션 구간에 대하여 시뮬레이션 엔진이 에뮬레이션 엔진과 상호작용 없이 독립적으로 실행할 수 있는 단위 구간을 epoch로 구분하고, 각각의 epoch에 대하여 시뮬레이션 엔진을 병렬적으로 실행하는 epoch 기반의 시뮬레이션 방법을 제공함에 목적이 있다.

[0009] 또한, 본 발명은 각각의 epoch에 대하여 에뮬레이션 엔진의 실행 없이 시뮬레이션 엔진을 병렬적 또는 직렬적으로 실행할 수 있는 epoch 기반의 시뮬레이션 방법을 제공함에 다른 목적이 있다.

과제의 해결 수단

[0011] 본 발명의 일 태양에 따른 Epoch 기반의 시뮬레이션 방법은, 소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 제1 epoch - 상기 제1 epoch는 상기 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 의 시작점을 검출하는 단계; 중앙처리부는 상기 에뮬레이션 엔진의 동작 중에 live-in 데이터를 기록하는 단계; 상기 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면, 상기 중앙처리부가 상기 제1 epoch의 종료점에 도달한 것으로 판단하는 단계; 상기 제1 epoch의 종료점에 도달하면, 상기 에뮬레이션 엔진은 상기 live-in 데이터의 기록을 종료하는 단계; 상기 에뮬레이션 엔진은 시뮬레이션 엔진에 상기 live-in 데이터를 전달하는 단계; 및 상기 live-in 데이터에 기반하여 상기 제1 epoch에 대한 시뮬레이션을 실행하는 단계를 포함한다.

[0012] 또한, 상기 에뮬레이션 엔진은 상기 live-in 데이터 중 상기 메모리 영역에 포함된 명령 및 데이터 자체를 상기 시뮬레이션 엔진에 전달한다.

[0013] 또한, 상기 에뮬레이션 엔진은 상기 live-in 데이터 중 상기 메모리 영역에 포함된 명령 및 데이터의 주소를 상기 시뮬레이션 엔진에 전달한다.

[0014] 또한, 상기 epoch 탐색용 프로그램 코드는 interrupt 명령, exception 명령, syscall 명령 또는 이에 상응하는 명령이다.

[0015] 또한, 본 발명의 다른 태양에 따른 Epoch 기반의 시뮬레이션 방법은 소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 제1 epoch - 상기 제1 epoch는 상기 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 의 시작점을 검출하는 제1 단계; 중앙처리부가 상기 에뮬레이션 엔진을 실행하여 상기 제1 epoch에 대하여 live-in 데이터를 기록하는 제2 단계; 상기 에뮬레이션 엔진이 상기 live-in 데이터 중 상기 제1 epoch에 대하여 시뮬레이션 엔진의 기동에 필수적인 live-in 데이터를 상기 시뮬레이션 엔진에 전달하면 상기 시뮬레이션 엔진을 실행하는 제3 단계; 상기 중앙처리부는 상기 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면 상기 제1 epoch 종료점에 도달한 것으로 판단하고, 상기 에뮬레이션 엔진에서 상기 live-in 데이터의 기록을 중지하는 제4 단계; 상기 제2 단계에서 기록되지 않은 상기 제1 epoch 종료점 정보를 포함한 live-in 데이터를 상기 시뮬레이션 엔진에 전달하는 제5 단계; 및 상기 시뮬레이션 엔진은 전달된 상기

제1 epoch 종료점 정보에 기반하여 상기 제1 epoch 종료점까지 상기 시뮬레이션을 진행하는 제6 단계를 포함한다.

[0016] 또한, 상기 에뮬레이션 엔진은 상기 live-in 데이터 중 상기 메모리 영역 정보 자체를 상기 시뮬레이션 엔진에 전달한다.

[0017] 또한, 상기 에뮬레이션 엔진은 상기 live-in 데이터 중 상기 메모리 영역 정보의 주소를 상기 시뮬레이션 엔진에 전달한다.

[0018] 또한, 본 발명의 또 다른 태양에 따른 Epoch 기반의 시뮬레이션 방법은 적어도 하나 이상의 epoch - 상기 epoch는 에뮬레이션 엔진과 상호 작용 없이 시뮬레이션 엔진의 동작만으로 시뮬레이션 됨 - 에 관한 live-in 데이터가 메모리에 저장되는 단계; 및 상기 적어도 하나 이상의 epoch에 관한 live-in 데이터에 기반하여 에뮬레이션 엔진의 기동 없이 시뮬레이션을 진행하는 단계를 포함한다.

발명의 효과

[0020] 본 발명의 epoch 기반의 시뮬레이션 방법에 따르면, 전체 시뮬레이션 구간에 대하여 시뮬레이션 엔진이 에뮬레이션 엔진과 상호작용 없이 독립적으로 실행할 수 있는 단위 구간을 epoch로 구분하고, 각각의 epoch에 대하여 시뮬레이터를 병렬적으로 실행하므로 시뮬레이션 진행 시간을 대폭 단축할 수 있다.

[0021] 본 발명은 메인메모리 또는 보조메모리에 저장된 epoch에 관한 명령 및 데이터를 포함하는 정보를 재사용함으로써 각각의 epoch에 대하여 에뮬레이션 엔진의 실행 없이 시뮬레이션 엔진을 병렬적 또는 직렬적으로 실행할 수 있어 시뮬레이션 진행 시간을 단축하는 효과가 있다.

도면의 간단한 설명

[0023] 도 1은 본 발명의 제1 실시예에 따른 epoch 기반의 시뮬레이션 개념도,

도 2는 본 발명의 제2 실시예에 따른 epoch 기반의 시뮬레이션 개념도, 및

도 3은 본 발명의 제3 실시예에 따른 epoch 기반의 시뮬레이션 개념도이다.

발명을 실시하기 위한 구체적인 내용

[0024] 이하, 본 발명의 바람직한 실시예(들)에 대하여 첨부도면을 참조하여 상세히 설명한다. 우선 각 도면의 구성요소들에 참조부호를 부가함에 있어서, 동일한 구성요소들에 한해서는 비록 다른 도면상에 표시되더라도 가능한 한 동일한 부호로 표기되었음에 유의하여야 한다. 또한, 하기의 설명에서는 구체적인 회로의 구성소자 등과 같은 많은 특정사항들이 도시되어 있는데, 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐 이러한 특정 사항들 없이도 본 발명이 실시될 수 있음은 이 기술분야에서 통상의 지식을 가진 자에게는 자명하다 할 것이다. 그리고 본 발명을 설명함에 있어서, 관련된 공지 기능 혹은 구성에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우, 그 상세한 설명을 생략한다.

[0026] 본 발명에서 epoch라 함은 시뮬레이션 엔진이 독립적으로 실행 가능한 단위 구간을 의미하는바, 예컨대, interrupt 명령, exception 명령, syscall 명령 또는 이에 상응하는 명령을 포함하는 epoch 탐색용 프로그램 코드가 존재함으로 인해 시뮬레이션 엔진이 단독으로 처리할 수 없는 명령을 기준으로 검출할 수 있다. Epoch의 시작점은 바로 직전 Epoch 탐색용 프로그램 코드의 다음 프로그램 코드로부터 시작되거나, 프로그래머가 임의로 지정한 소정 코드 혹은 소정 시점부터 시작될 수 있으며, 시뮬레이션 엔진이 단독으로 처리할 수 없는 명령 직 전까지의 명령 집합을 하나의 Epoch라 한다.

[0027] 본 발명에서 '에뮬레이션'이라 함은 프로그램의 기능적 동작을 파악할 수 있도록 프로그램을 실행하는 방법을 의미하고, '시뮬레이션'이라 함은 아키텍처의 타이밍 동작 및 프로그램의 기능적 동작을 파악할 수 있도록 타겟이 모델링된 아키텍처를 cycle-accurate 방법으로 프로그램을 실행하는 방법을 의미한다.

[0028] 도 1은 본 발명의 제1 실시예에 따른 epoch 기반의 시뮬레이션 개념도이다.

[0029] 본 발명의 제1 실시예에 따른 epoch 기반의 시뮬레이션 방법에 따르면, 소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 상기 epoch의 시작점에 해당하는 코드 혹은 지점을 검출함으로써 시작된다. 이후 에뮬레이션 모드에서 시뮬레이션 엔진의 실행을 위해 필요한 PC, 레지스터 등의 CPU 상태, 메모리 상태 및 데이터 등의 데이터(이하, 'live-in 데이터'라 함)를 기록한다(S110).

- [0030] 여기서, live-in 데이터라 함은, 예시적으로, 시뮬레이션 엔진이 시뮬레이션을 진행하기 위한 메모리 영역(예: 메인 메모리, 룬, 바이오스, 이더넷 메모리, 그래픽 메모리 등), 메모리 관리를 위한 virtual to physical 메모리 매핑 정보 및 각 페이지의 상태 기록(예: unassigned, valid, clean, dirty 등), epoch 시작 지점에서 시뮬레이션 엔진을 실행하기 위한 CPU 상태 값 기록(예: CPU 레지스터 값, flag 값 등), 실행된 프로그램 명령의 개수, 타겟 아키텍처의 메모리 크기, 및 CPU 개수를 포함할 수 있다.
- [0031] S110에서 시뮬레이션에서 필요한 모든 live-in 데이터를 기록할 수 있지만, 가상의 메인 메모리용 데이터의 경우, 그 크기가 상당하므로 저장 시간과 저장 공간에 대한 오버헤드가 발생할 수 있다. 따라서 저장 시간과 저장 공간의 최적화를 위하여, S110에서 가상의 메인 메모리용 데이터를 모두 기록하지 않고, S120에서 시뮬레이션 구간에서 사용되는 메모리 영역(예: 메인 메모리 중 시뮬레이션 구간에서 접근된 메모리 영역)의 데이터만을 기록할 수 있다. 그리고 이 데이터는 제1 epoch을 실행하면서 사용된 명령과 데이터를 포함한다. 여기서, 제1 epoch(epoch 1)은 에뮬레이션과 상호 작용없이 독립적으로 시뮬레이션 된다.
- [0032] 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면, 중앙처리부(미도시)는 제1 epoch의 종료점에 도달한 것으로 판단하고, 제1 epoch에 관한 정보(예, 실행된 프로그램 명령의 개수, 타겟 아키텍처의 메모리 크기, CPU 개수 등)를 메인 메모리 또는 보조 메모리에 저장하고, live-in 데이터의 기록을 중지하며, 에뮬레이션 엔진은 시뮬레이션 엔진에 live-in 데이터를 전달한다(S130). 여기서, 제1 epoch에 관한 정보 중 실행된 프로그램 명령의 개수는 S130에서 기록되는 정보이지만, 타겟 아키텍처의 메모리 크기나 CPU 개수는 단계 S110 내지 S130 중 어느 하나의 단계에서 기록될 수 있다.
- [0033] 이때, 본 발명의 일실시예에 따르면, 에뮬레이션 엔진은 기록된 live-in 데이터 모두를 시뮬레이션 엔진에 전달할 수 있다. 또한, 본 발명의 다른 실시예에 따르면, 전달 오버헤드를 줄이기 위해 에뮬레이션 엔진을 이용하여 단계 S120을 실행하면서 사용된 영역의 live-in 데이터만을 전달할 수 있다.
- [0035] live-in 데이터의 전달이 완료되면, 제1 epoch에 대한 시뮬레이션을 실행한다(S140). 본 발명의 일실시예에 따르면, 에뮬레이션 엔진은 메모리 영역에 포함된 live-in 데이터 자체를 시뮬레이션 엔진에 전달할 수 있다. 또한, 본 발명의 다른 실시예에 따르면, 에뮬레이션 엔진은 메모리 영역에 포함된 live-in 데이터의 주소를 시뮬레이션 엔진에 전달할 수 있다.
- [0037] 이후, 에뮬레이션 엔진을 이용하여 에뮬레이션을 직렬적으로 수행함과 아울러 epoch 탐색용 프로그램 코드가 검출되면, 제2 epoch, 제3 epoch, 제4 epoch에 대하여 시뮬레이션을 병렬적으로 진행할 수 있다.
- [0038] 즉, 본 발명에 따르면, 해당 프로그램에 대하여 에뮬레이션을 진행해 가면서 epoch로 확인되면, 즉시 해당 epoch에 대하여 시뮬레이션을 진행하는바, 제1 epoch에 대한 시뮬레이션이 종료되는 것과 무관하게 제2 epoch에 대한 시뮬레이션을 실행함으로써 시뮬레이션 시간을 대폭 단축할 수 있다.
- [0040] 한편, 에뮬레이션 엔진에서의 명령 시퀀스 및 메모리 상태와 시뮬레이션 엔진에서의 명령 시퀀스 및 메모리 상태를 비교함으로써 시뮬레이션 방법의 정확성을 검증할 수 있다.
- [0041] 즉, 단계 S120의 제1 epoch 구간 동안 실행된 명령을 기록하고, 단계 S150의 시뮬레이션 구간 동안 실행된 명령을 기록하여 비교함으로써 에뮬레이션 엔진 및 시뮬레이션 엔진에서 실행한 명령 시퀀스가 일치하는 것을 확인한다.
- [0042] 또한, 단계 S130에서 제1 epoch 종료점에서 메모리 상태를 기록하고, 단계 S160에서 제1 epoch에 대한 시뮬레이션 종료점에서의 메모리 상태를 기록하여 상호 비교함으로써 에뮬레이션 엔진 및 시뮬레이션 엔진에서의 메모리 상태가 일치하는 것을 확인한다.
- [0044] 도 2는 본 발명의 제2 실시예에 따른 epoch 기반의 시뮬레이션 개념도이다.
- [0045] 본 발명의 제2 실시예에 따른 epoch 기반의 시뮬레이션 방법에 따르면, 소정 프로그램을 에뮬레이션 엔진에서 실행하면서, 상기 epoch의 시작점에 해당하는 코드 혹은 지점을 검출함으로써 시작된다. 이후 에뮬레이션 엔진에서 live-in 데이터를 기록하고(S210), 에뮬레이션 모드를 계속한다(S220). 이때, 실시예 1과 마찬가지로 저장 시간과 저장 공간의 최적화를 위해 단계 S210에서 가상의 메인 메모리의 데이터를 모두 기록하지 않고, 에뮬레이션 구간에서 사용되는 메모리 영역(예: 메인 메모리 중 에뮬레이션 구간에서 접근된 메모리 영역)을 기록할 수 있으며, 기록된 메인 메모리 데이터를 시뮬레이션 엔진으로 전달한다.
- [0046] 일반적으로 시뮬레이션 실행 시간은 에뮬레이션 실행 시간보다 매우 길어, 상기 epoch의 시작점에 해당하는 코드가 검출되고, 상기 epoch의 시뮬레이션 시작에 필수적인 live-in 데이터(예: CPU 상태 값, 타겟 아키텍처의

메모리 크기, CPU 개수 등)만 전달되면 시뮬레이션을 실행한다(S230). 이후 시뮬레이션에서 사용할 live-in 데이터들은 에뮬레이션 엔진의 실행 동안 기록되어 시뮬레이션 엔진으로 전달되는데, 시뮬레이션 엔진의 실행 시간이 에뮬레이션 엔진의 실행 시간에 비해 매우 느리기 때문에 가능하다(S220).

[0047] 중앙처리부(미도시)는 에뮬레이션 엔진의 동작 중에 epoch 탐색용 프로그램 코드가 검출되면 제1 epoch 종료점에 도달한 것으로 판단하며, 단계 S210에서 기록되지 않은 live-in 데이터(예: 실행된 프로그램 명령의 개수)를 시뮬레이션 엔진에 전달한다(S240).

[0048] 시뮬레이션 엔진은 전달된 epoch 종료점까지 시뮬레이션을 진행한다(S250).

[0050] 도 3은 본 발명의 제3 실시예에 따른 epoch 기반의 시뮬레이션 개념도이다.

[0051] 본 발명의 제3 실시예에 따르면, 에뮬레이션 엔진을 재실행하지 않고, 기록된 데이터를 재사용하여 병렬로 시뮬레이션을 진행하는 방법이 제안된다.

[0052] 즉, 메인메모리 또는 보조메모리에 저장된 epoch 에 관한 명령 및 데이터를 포함하는 정보를 재사용하여 시뮬레이션 엔진의 configuration 만을 변화시켜 병렬적 또는 직렬적으로 시뮬레이션을 진행할 수 있다.

[0053] 단계 S310에서 동시에 4개의 epoch에 대한 시뮬레이션이 진행되는 한편, 단계 S320에서는 각기 epoch 종료점이 달라 상이한 시점에서 연이은 4개의 epoch에 대하여 시뮬레이션이 진행된다.

[0055] 위에서 개시된 발명은 기본적인 사상을 훼손하지 않는 범위 내에서 다양한 변형예가 가능하다. 즉, 위의 실시예들은 모두 예시적으로 해석되어야 하며, 한정적으로 해석되지 않는다. 따라서 본 발명의 보호범위는 상술한 실시예가 아니라 첨부된 청구항에 따라 정해져야 하며, 첨부된 청구항에 한정된 구성요소를 균등물로 치환한 경우 이는 본 발명의 보호범위에 속하는 것으로 보아야 한다.

부호의 설명

[0057] S110: 초기 데이터 기록 단계

S120: 에뮬레이션 모드시 메모리 영역 기록 단계

S130: 에뮬레이션 모드시 메모리 상태 기록 및 초기 데이터 전달 단계

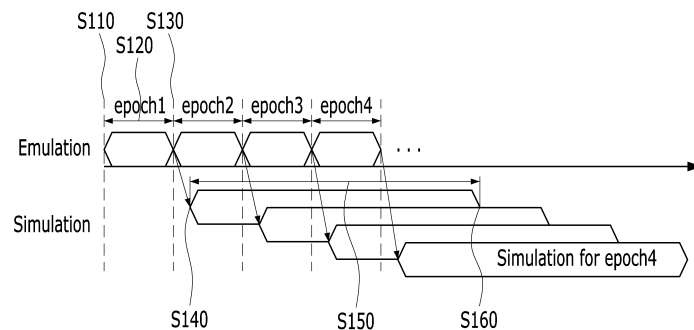
S140: 시뮬레이션 실행 단계

S150: 시뮬레이션 모드시 메모리 영역 기록 단계

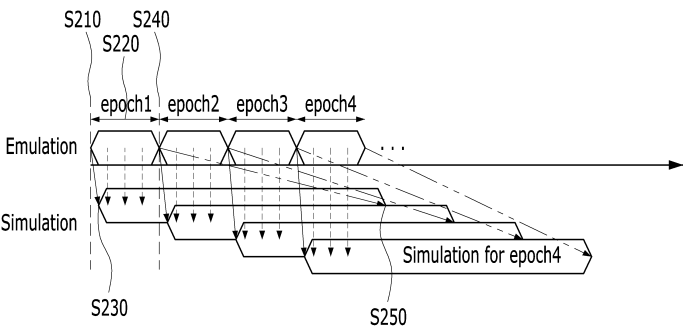
S160: 시뮬레이션 모드시 메모리 상태 기록 단계

도면

도면1



도면2



도면3

