(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0002608
(43) 공개일자 2025년01월07일

(51) 국제특허분류(Int. Cl.)
G06F 12/14 (2006.01) *G06F 12/1009* (2016.01)
G06F 9/30 (2018.01) *G06F 9/38* (2024.01)
G06F 9/46 (2006.01)
(52) CPC특허분류
G06F 12/1441 (2013.01)
G06F 12/1009 (2013.01)
(21) 출원번호 10-2024-7038784
(22) 출원일자(국제) 2023년03월17일
심사청구일자 없음
(85) 번역문제출일자 2024년11월21일
(86) 국제출원번호 PCT/GB2023/050653
(87) 국제공개번호 WO 2023/209323
국제공개일자 2023년11월02일
(30) 우선권주장
2206164.2 2022년04월28일 영국(GB)

(71) 출원인
에이알엠 리미티드
영국 캠브리지 씨비1 9엔제이 체리힌톤 풀번로드 110
(72) 발명자
크라스키, 사이먼 존
영국 씨비1 9엔제이 캠브리지쉬어 캠브리지 풀번로드 110 에이알엠 리미티드 씨/오
호레이, 존 미첼
영국 씨비1 9엔제이 캠브리지쉬어 캠브리지 풀번로드 110 에이알엠 리미티드 씨/오
(74) 대리인
이화익

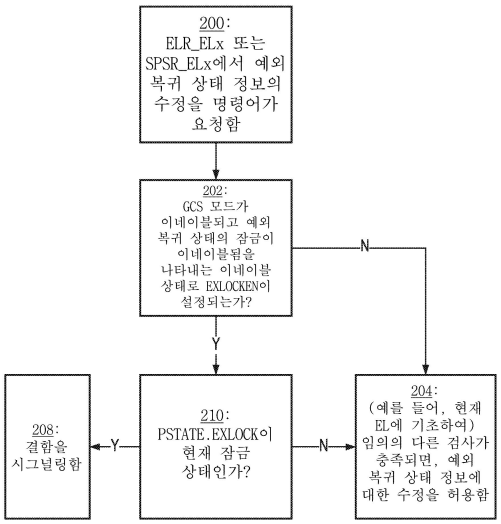
전체 청구항 수 : 총 18 항

(54) 발명의 명칭 **예외 복귀 상태 잠금 파라미터**

(57) 요 약

장치는 예외 복귀 상태 레지스터 저장소와 프로세싱 회로부를 포함한다. 보호형 제어 스택(guarded control stack; GCS) 예외 복귀 상태 푸시(push) 명령어에 응답하여, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소로부터 예외 복귀 상태 정보를 획득하고 상태 정보를 GCS 데이터 구조로 푸시한다. GCS 예외 복귀 상태 팝 명령어에 응답하여, 프로세싱 회로부는 GCS 데이터 구조로부터 GCS-보호된 예외 복귀 상태 정보를 획득한다. 적어도 하나의 동작 상태에서, 프로세싱 회로부는, 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있을 때 결함을 시그널링한다.

대 표 도 - 도13



(52) CPC특허분류
G06F 12/1491 (2013.01)
G06F 9/3004 (2013.01)
G06F 9/30101 (2013.01)
G06F 9/3861 (2013.01)
G06F 9/461 (2013.01)

명 세 서

청구범위

청구항 1

장치로서,

예외 복귀 상태 정보를 저장하기 위한 예외 복귀 상태 레지스터 저장소; 및

명령어들에 응답하여 프로세싱 동작들을 수행하기 위한 프로세싱 회로부를 포함하며; 여기서,

보호형 제어 스택(guarded control stack; GCS) 예외 복귀 상태 푸시(push) 명령어에 응답하여, 상기 프로세싱 회로부는 상기 예외 복귀 상태 레지스터 저장소로부터 상기 예외 복귀 상태 정보를 획득하고 상기 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하도록 구성되고;

GCS 예외 복귀 상태 팝 명령어에 응답하여, 상기 프로세싱 회로부는 상기 GCS 데이터 구조로부터 GCS-보호된 예외 복귀 상태 정보를 획득하도록 구성되고;

상기 장치는, 타겟 어드레스에 대응하는 메모리 영역이 상기 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 상기 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 상기 타겟 어드레스를 명시하는 기록 요청을 거부하기 위한 메모리 관리 회로부를 포함하며, 상기 GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 상기 GCS 예외 복귀 상태 푸시 명령어를 포함하고;

상기 프로세싱 회로부의 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는 상기 예외 복귀 상태 레지스터 저장소에 저장된 상기 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하도록 구성되는, 장치.

청구항 2

제1항에 있어서,

상기 프로세싱 회로부는 상이한 레벨들의 권한과 연관된 복수의 예외 레벨들 중 하나에서 상기 프로세싱 동작들을 수행하도록 구성되고;

상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는 예외가 취해지는 것에 응답하여 상기 예외 복귀 상태 잠금 파라미터를 상기 잠금 상태로 설정하도록 구성되고, 상기 예외는 상기 예외가 취해지기 전의 현재 예외 레벨과 동일한 예외 레벨에서 취해지는, 장치.

청구항 3

제1항 또는 제2항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는 상기 GCS 예외 복귀 상태 푸시 명령어에 응답하여 상기 예외 복귀 상태 잠금 파라미터를 상기 잠금 해제 상태로 설정하도록 구성되는, 장치.

청구항 4

제1항 내지 제3항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는 상기 GCS 예외 복귀 상태 팝 명령어에 응답하여 상기 예외 복귀 상태 잠금 파라미터를 상기 잠금 상태로 설정하도록 구성되는, 장치.

청구항 5

제1항 내지 제4항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는, 상기 예외 복귀 상태 레지스터 저장소에 저장된 상기 예외 복귀 상태 정보에 기초하여 예외 복귀를 트리거하기 위한 예외 복귀 명령어에 응답하여 - 상기 예외 복귀는 상기 예외 복귀 전의 현재 예외 레벨

과 동일한 예외 레벨에서의 프로세싱으로의 복귀임 -, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태로 설정되는지 여부를 검사하고, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 해제 상태로 설정된다고 검출하는 것에 응답하여 결함을 시그널링하도록 구성되는, 장치.

청구항 6

제1항 내지 제5항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 프로세싱 회로부는, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태로 설정될 때, 상기 예외 복귀 상태 레지스터 저장소에 기록될 새로운 값이 상기 예외 복귀 상태 레지스터 저장소에 미리 저장된 오래된 값과 동일하거나 상이한지와 무관하게, 상기 예외 복귀 상태 레지스터 저장소 내의 정보에 대한 수정을 요청하는 명령어를 실행하려는 시도에 응답하여 결함을 시그널링하도록 구성되는, 장치.

청구항 7

제1항 내지 제5항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태에서, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태로 설정될 때, 상기 예외 복귀 상태 레지스터 저장소 내의 정보에 대한 수정을 요청하는 명령어를 실행하려는 시도에 응답하여, 상기 프로세싱 회로부는 상기 예외 복귀 상태 레지스터 저장소에 기록될 새로운 값이 상기 예외 복귀 상태 레지스터 저장소 내에 이전에 저장된 오래된 값과 동일하거나 상이한지 여부를 검사하고, 상기 새로운 값이 상기 오래된 값과 상이한 것으로 결정될 때 상기 결함을 시그널링하도록 구성되는, 장치.

청구항 8

제1항 내지 제7항 중 어느 한 항에 있어서, 상기 GCS 예외 복귀 상태 팝 명령어에 응답하여, 상기 프로세싱 회로부는, 상기 예외 복귀 상태 레지스터 저장소로부터 획득된 의도된 예외 복귀 상태 정보와 상기 GCS 데이터 구조로부터 획득된 GCS-보호된 예외 복귀 상태 정보 사이의 미스매치가 검출될 때 결함을 시그널링하도록 구성되는, 장치.

청구항 9

제1항 내지 제8항 중 어느 한 항에 있어서, 상기 예외 복귀 상태 레지스터 저장소는, 예외로부터 복귀한 후에 실행될 명령어의 예외 복귀 어드레스를 저장하기 위한 예외 복귀 어드레스 레지스터를 포함하는, 장치.

청구항 10

제1항 내지 제9항 중 어느 한 항에 있어서, 상기 예외 복귀 상태 레지스터 저장소는 예외로부터 복귀할 때 복원될 프로세서 상태를 저장하기 위해 저장된 프로세서 상태 레지스터를 포함하는, 장치.

청구항 11

제10항에 있어서, 예외를 취하는 것에 응답하여, 상기 프로세싱 회로부는 프로세서 상태를 프로세서 상태 레지스터로부터 상기 저장된 프로세서 상태 레지스터로 복사하도록 구성되고;

상기 예외 복귀 상태 잠금 파라미터는 상기 프로세서 상태 레지스터에 명시되는 파라미터인, 장치.

청구항 12

제11항에 있어서, 예외 복귀 명령어에 응답하여, 상기 프로세싱 회로부는, 저장된 예외 복귀 상태 잠금 파라미터를 포함하는 저장된 프로세서 상태를, 상기 저장된 프로세서 상태 레지스터로부터 상기 프로세서 상태 레지스터로 복사하여, 상기 예외 복귀 상태 잠금 파라미터를 상기 저장된 예외 복귀 상태 잠금 파라미터에 의해 표시된 값으로 복원하도록 구성되는, 장치.

청구항 13

제1항 내지 제12항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태는, 예외-복귀-상태-잠금 이네이블 제어 파라미터가 상기 예외 복귀 상태 잠금 파라미터를 사용하는 예외 복귀 상태 정보의 잠금이 이네이블됨을 나타내는 이네이블 상태로 설정되는, 동작 상태를 포함하는, 장치.

청구항 14

제1항 내지 제12항 중 어느 한 항에 있어서, 상기 프로세싱 회로부의 상기 적어도 하나의 동작 상태는:

예외-복귀-상태-잠금 이네이블 제어 파라미터가, 상기 예외 복귀 상태

잠금 파라미터를 사용하여 예외 복귀 상태 정보의 잠금이 이네이블됨을 나타

내는 이네이블 상태로 설정되고;

GCS-이네이블 파라미터가, 상기 GCS 데이터 구조의 사용을 지원하기

위한 GCS 모드가 이네이블되어 있음을 나타내는 이네이블 상태로 설정되는,

둘 모두의 동작 상태를 포함하는, 장치.

청구항 15

제1항 내지 제14항 중 어느 한 항에 있어서, 상기 메모리 관리 회로부는, 주어진 타겟 어드레스에 대응하는 메모리 영역이 상기 GCS 영역으로서 명시되지 않는다는 결정에 응답하여, 상기 주어진 타겟 어드레스를 명시하는 GCS 메모리 액세스 요청을 거부하도록 구성되는, 장치.

청구항 16

방법으로서,

예외 복귀 상태 정보를 예외 복귀 상태 레지스터 저장소에 저장하는 단계; 및

프로세싱 회로부를 사용하여, 명령어들에 응답하여 프로세싱 동작들을 수행하는 단계;

보호형 제어 스택(GCS) 예외 복귀 상태 푸시 명령어에 응답하여, 상기 예외 복귀 상태 레지스터 저장소로부터 상기 예외 복귀 상태 정보를 획득하고 상기 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하는 단계;

GCS 예외 복귀 상태 팝 명령어에 응답하여, 상기 GCS 데이터 구조로부터 GCS-보호된 예외 복귀 상태 정보를 획득하는 단계;

타겟 어드레스에 대응하는 메모리 영역이 상기 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 상기 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 상기 타겟 어드레스를 명시하는 기록 요청을 거부하는 단계 - 상기 GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 상기 GCS 예외 복귀 상태 푸시 명령어를 포함함 -; 및

상기 프로세싱 회로부의 적어도 하나의 동작 상태에서, 상기 예외 복귀 상태 레지스터 저장소에 저장된 상기 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하는 단계를 포함하는, 방법.

청구항 17

호스트 데이터 프로세싱 장치에 의해 실행될 때, 타겟 코드를 실행시키기 위한 명령어 실행 환경을 제공하기 위해 상기 호스트 데이터 프로세싱 장치를 제어하는 명령어들을 포함하는 컴퓨터 프로그램으로서, 상기 컴퓨터 프로그램은,

예외 복귀 상태 정보를 저장하기 위한 예외 복귀 상태 레지스터 저장소를 시뮬레이팅하도록 상기 호스트 데이터 프로세싱 장치를 제어하기 위한 레지스터 에뮬레이팅 프로그램 논리; 및

상기 타겟 코드의 명령어들의 실행을 시뮬레이팅하도록 상기 호스트 데이터 프로세싱 장치를 제어하기 위한 프로세싱 프로그램 논리를 포함하며; 여기서,

상기 타켓 코드의 보호형 제어 스택(GCS) 예외 복귀 상태 푸시 명령어에 응답하여, 상기 프로세싱 프로그램 논리는 상기 시뮬레이팅된 예외 복귀 상태 레지스터 저장소로부터 상기 예외 복귀 상태 정보를 획득하고 상기 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하기 위해 상기 호스트 데이터 프로세싱 장치를 제어하도록 구성되

고;

상기 타겟 코드의 GCS 예외 복귀 상태 팝 명령어에 응답하여, 상기 프로세싱 프로그램 논리는 상기 GCS 데이터 구조로부터 획득되는 GCS-보호된 예외 복귀 상태 정보를 획득하도록 구성되고;

상기 컴퓨터 프로그램은, 타겟 어드레스에 대응하는 메모리 영역이 상기 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 상기 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 상기 타겟 어드레스를 명시하는 기록 요청을 거부하기 위한 메모리 관리 프로그램 논리를 포함하며, 상기 GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 상기 GCS 예외 복귀 상태 푸시 명령어를 포함하고;

상기 프로세싱 프로그램 논리의 적어도 하나의 시뮬레이팅된 동작 상태에서, 상기 프로세싱 프로그램 논리는 상기 시뮬레이팅된 예외 복귀 상태 레지스터 저장소에 저장된 상기 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 상기 예외 복귀 상태 잠금 파라미터가 상기 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하기 위해 상기 호스트 데이터 프로세싱 장치를 제어하도록 구성되는, 컴퓨터 프로그램.

청구항 18

제17항의 컴퓨터 프로그램을 저장하는 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 기법은 데이터 프로세싱의 분야에 관한 것이다.

[0002] 데이터 프로세싱 장치는 예외를 취하는 것에 반응하여 예외 복귀 상태 정보를 저장할 수 있다. 예외 복귀 상태 정보는 예외를 취할 시의 프로세서 상태를 나타내므로 예외 핸들링이 완료되면, 프로세서는, 예외가 취해지기 전에 수행되고 있는 프로세싱으로 복귀할 수 있다.

[0003] 적어도 일부 예들은 예외 복귀 상태 정보를 저장하기 위한 예외 복귀 상태 레지스터 저장소; 및 명령어들에 응답하여 프로세싱 동작들을 수행하기 위한 프로세싱 회로부를 포함하는 장치를 제공하며; 여기서 보호형 제어 스택(guarded control stack; GCS) 예외 복귀 상태 푸시(push) 명령어에 응답하여, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소로부터 예외 복귀 상태 정보를 획득하고 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하도록 구성되고; GCS 예외 복귀 상태 팝 명령어에 응답하여, 프로세싱 회로부는 GCS 데이터 구조로부터 GCS-보호된(GCS-protected) 예외 복귀 상태 정보를 획득하도록 구성되고; 본 장치는, 타겟 어드레스에 대응하는 메모리 영역이 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 기록 요청이 GCS-액세스 타입들(GCS-accessing type)의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 타겟 어드레스를 명시하는 기록 요청을 거부하기 위한 메모리 관리 회로부를 포함하며, GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 GCS 예외 복귀 상태 푸시 명령어를 포함하고; 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하도록 구성된다.

[0004] 적어도 일부 예들은, 예외 복귀 상태 정보를 예외 복귀 상태 레지스터 저장소에 저장하는 단계; 및 프로세싱 회로부를 사용하여, 명령어들에 응답하여 프로세싱 동작들을 수행하는 단계; 보호형 제어 스택(GCS) 예외 복귀 상태 푸시 명령어에 응답하여, 예외 복귀 상태 레지스터 저장소로부터 예외 복귀 상태 정보를 획득하고 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하는 단계; GCS 예외 복귀 상태 팝 명령어에 응답하여, GCS 데이터 구조로부터 GCS-보호된 예외 복귀 상태 정보를 획득하는 단계; 타겟 어드레스에 대응하는 메모리 영역이 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 타겟 어드레스를 명시하는 기록 요청을 거부하는 단계 - GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 GCS 예외 복귀 상태 푸시 명령어를 포함함 -; 및 프로세싱 회로부의 적어도 하나의 동작 상태에서, 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상

태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하는 단계를 포함하는 방법을 제공한다.

[0005] 적어도 일부 예들은, 호스트 데이터 프로세싱 장치에 의해 실행될 때, 타겟 코드를 실행시키기 위한 명령어 실행 환경을 제공하기 위해 호스트 데이터 프로세싱 장치를 제어하는 명령어들을 포함하는 컴퓨터 프로그램을 제공하고, 컴퓨터 프로그램은, 예외 복귀 상태 정보를 저장하기 위한 예외 복귀 상태 레지스터 저장소를 시뮬레이팅하도록 호스트 데이터 프로세싱 장치를 제어하기 위한 레지스터 에뮬레이팅 프로그램 논리; 및 타겟 코드의 명령어들의 실행을 시뮬레이팅하도록 호스트 데이터 프로세싱 장치를 제어하기 위한 프로세싱 프로그램 논리를 포함하며; 여기서 타켓 코드의 보호형 제어 스택(GCS) 예외 복귀 상태 푸시 명령어에 응답하여, 프로세싱 프로그램 논리는 시뮬레이팅된 예외 복귀 상태 레지스터 저장소로부터 예외 복귀 상태 정보를 획득하고 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시하기 위해 호스트 데이터 프로세싱 장치를 제어하도록 구성되고; 타겟 코드의 GCS 예외 복귀 상태 팝 명령어에 응답하여, 프로세싱 프로그램 논리는 GCS 데이터 구조로부터 획득되는 GCS-보호된 예외 복귀 상태 정보를 획득하도록 구성되고; 컴퓨터 프로그램은, 타겟 어드레스에 대응하는 메모리 영역이 GCS 데이터 구조를 저장하기 위한 GCS 영역으로서 명시되고 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여, 타겟 어드레스를 명시하는 기록 요청을 거부하기 위한 메모리 관리 프로그램 논리를 포함하며, GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 GCS 예외 복귀 상태 푸시 명령어를 포함하며; 프로세싱 프로그램 논리의 적어도 하나의 시뮬레이팅된 동작 상태에서, 프로세싱 프로그램 논리는 시뮬레이팅된 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링하기 위해 호스트 데이터 프로세싱 장치를 제어하도록 구성된다.

[0006] 컴퓨터 프로그램은 저장 매체에 저장될 수 있다. 저장 매체는 비일시적 저장 매체일 수 있다.

[0007] 본 기법의 추가의 양태들, 특징들 및 이점들은 첨부 도면들과 관련하여 읽혀질 예들의 하기의 설명으로부터 명백해질 것이다.

[0008] 도 1은 프로세싱 회로부 및 메모리 관리 회로부를 갖는 데이터 프로세싱 장치의 예를 예시한다;

[0009] 도 2는 프로세싱 회로부의 실행 상태들의 예를 예시한다;

[0010] 도 3은 2-스테이지 어드레스 변환을 예시한다;

[0011] 도 4는 프로세싱 회로부의 레지스터들의 예를 예시한다.

[0012] 도 5는 취해진 예외로의 엔트리 및 예외로부터의 복귀 시 실행되는 명령어들의 예를 예시한다.

[0013] 도 6은 예외 엔트리에 응답하여 캡처(capture)된 복귀 상태 정보를 보호하기 위한 보호형 제어 스택(GCS) 구조의 사용을 예시한다.

[0014] 도 7은 GCS 구조를 사용할 때, 취해진 예외로의 엔트리 및 예외로부터의 복귀 시 실행되는 명령어들의 예를 예시한다.

[0015] 도 8은 메모리 액세스 요청이 허용되는지 여부를 검사(check)하는 방법을 예시하는 흐름도이다.

[0016] 도 9는 예외 엔트리 시에 수행되는 단계들을 예시하는 흐름도이다.

[0017] 도 10은 GCS 예외 복귀 상태 푸시 명령어에 응답하여 수행되는 단계들을 도시하는 흐름도이다.

[0018] 도 11은 GCS 예외 복귀 상태 팝 명령어에 응답하여 수행되는 단계들을 도시하는 흐름도이다.

[0019] 도 12는 예외 복귀 명령어에 응답하여 수행되는 단계들을 도시하는 흐름도이다.

[0020] 도 13 및 도 14는 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보의 수정을 요청하는 명령어에 응답하여 수행되는 단계들의 2개의 대안적인 예들을 도시하는 흐름도들이다.

[0021] 도 15는 시뮬레이터 구현을 예시한다.

[0022] 장치는 예외 복귀 상태 정보를 저장하기 위한 예외 복귀 상태 레지스터 저장소, 및 명령어들에 응답하여 프로세싱 동작들을 수행하기 위한 프로세싱 회로부를 갖는다.

[0023] 복귀 배향 프로그래밍(return-oriented-programming; ROP) 기반 공격은 데이터 프로세싱 시스템들에 대한 일반

적인 종류의 공격이다. ROP 공격은 함수 호출 또는 예외로부터 복귀하는 데 사용되는 복귀 상태 정보를 손상시켜 프로그램이 예기치 않은 방식으로 거동하게 하는 공격이다. 종종 소프트웨어는 예를 들어, 함수 호출 또는 예외의 내포(nesting)를 용이하게 하기 위해 복귀 상태 정보를 메모리에 저장할 것이며, (함수 호출 또는 예외의 내포된 세트의) 외부 함수 호출 또는 예외에 대한 복귀 상태 정보는 내부 함수 호출 또는 예외에 대한 복귀 상태 정보를 사용해 레지스터들에서 해당 정보가 겹쳐 쓰여 질 수 있기 전에 해당 정보를 보전하기 위해 메모리에 저장된다. 성공적인 ROP 공격은 함수 복귀 또는 예외 복귀로 하여금 함수가 호출되었거나 또는 예외가 취해진 지점 이후에 다음 명령어 이외의 명령어로 프로그램 흐름을 복귀하게 할 수 있으며, 이는 공격자가 프로그래머에 의해 의도된 동작 시퀀스 이외의 임의의 동작들을 수행하도록 프로세싱 회로부를 제어하게 할 수 있다.

[0024] 예외 복귀 상태 정보를 템퍼링하는 ROP 기반 공격에 대항해 보호하기 위해, 예외로부터 복귀하기 위한 복귀 상태 정보를 보호하는 데 사용되는 보호형 제어 스택(GCS) 데이터 구조를 제공하는 것이 유용할 수 있다. 보호형 제어 스택(GCS) 예외 복귀 상태 푸시 명령어에 응답하여, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소로부터 예외 복귀 상태 정보를 획득하고 예외 복귀 상태 정보를 GCS 데이터 구조로 푸시한다. GCS 예외 복귀 상태 팝 명령어에 응답하여, 프로세싱 회로부는 GCS 데이터 구조로부터 GCS-보호된 예외 복귀 상태를 획득한다. GCS 데이터 구조에 대한 보호는 GCS 데이터 구조를 저장하도록 의도되는 GCS 영역인 것으로서 메모리 영역을 명시하는 것을 지원하는 메모리 관리 회로부에 의해 제공될 수 있다. 메모리 관리 회로부는, 기록 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트(GCS-액세스 타입들의 명령어의 제한된 서브세트는 적어도 GCS 예외 복귀 상태 푸시 명령어를 포함함) 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청인 경우, 타겟 어드레스에 대응하는 메모리 영역이 GCS 영역으로서 명시된다고 결정하는 것에 응답하여 타겟 어드레스를 명시하는 기록 요청들을 거부한다. 따라서, 모든 타입들의 기록 요청(예를 들어, 범용(GCS-특유가 아닌) 저장 명령어들은 GCS-액세스 타입들의 명령어의 제한된 서브세트로부터 배제될 수 있음)에 의해 기록될 수 없는 메모리의 보호되는 영역이 정의될 수 있다. 이렇게 하면 공격자가 GCS 데이터 구조에 저장된 GCS-보호된 예외 복귀 상태를 손상시키려고 시도하는 데 사용할 수 있는 공격 표면(attack surface)이 제한된다. 따라서, 표준 판독 가능/기록 가능 메모리 영역에서 소프트웨어에 의해 유지되는 범용 스택을 사용하여 가능한 것보다 예외 복귀 상태에 대해 더 강한 보호가 제공될 수 있다.

[0025] 그러나, 발명자는, (예외 복귀 상태가 예외 복귀 상태 레지스터 저장소에 캡처되는 경우) 예외 엔트리와 GCS 예외 복귀 상태 푸시 명령어의 실행 사이에 시간의 윈도우(window)가 있을 수 있고, 예외 복귀 상태 정보가 레지스터들에서의 템퍼링에 취약할 수 있는 경우, GCS 예외 복귀 상태 팝 명령어의 실행과 예외 복귀 사이에 또 다른 시간의 윈도우가 있을 수 있다는 것을 인식하였는데, 이 템퍼링이 성공하면 GCS 데이터 구조에 저장된 보호된 예외 복귀 상태 정보에 기초하여 검사를 우회할 수 있다. 공격자가 예외 엔트리와 GCS 예외 복귀 상태 푸시 명령어의 실행 사이에서 예외 복귀 상태 레지스터 저장소 내의 예외 복귀 상태를 수정할 수 있다면, 잘못된 예외 복귀 상태가 GCS 데이터 구조로 푸시될 수 있고, 따라서 GCS 데이터 구조에서 신뢰되는 것으로 잘못 간주될 수 있다. 유사하게, 어떤 공격은 GCS 예외 복귀 상태 팝 명령어의 실행 후이지만 예외 복귀의 실행 전에 예외 복귀 상태 레지스터 저장소에서의 예외 복귀 상태를 수정하는 것에 기초할 수 있다.

[0026] 이러한 문제를 해결하기 위한 한 가지 접근법은, 이들 시간 윈도우들을 제거하기 위해, GCS 예외 복귀 상태 푸시 명령어의 동작들이 그 대신에 예외 엔트리 자체에 응답하여 하드웨어에서 수행되고, GCS 예외 복귀 상태 팝 명령어의 동작들이 예외 복귀를 트리거하는 동일한 명령어에 응답하여 수행되는 것을 요구하는 것일 수 있다. 그러나 이 접근법은 단일 이벤트(예외 엔트리 및 복귀)에 의해 트리거되어야 하는 동작들의 수를 증가시킬 수 있다. 단일 트리거 이벤트에 응답하여 요구되는 액션들의 이러한 과부하는 프로세서 마이크로아키텍처 설계자가 회로 타이밍 요건들을 충족시키면서 이들 동작들을 지원하기 위한 회로 하드웨어를 구현하는 것을 훨씬 더 복잡하게 만들 수 있다. 또한, GCS 예외 복귀 상태 푸시 명령어를 예외 엔트리로부터 분리하고 GCS 예외 복귀 상태 팝 명령어를 예외 복귀로부터 분리하는 것은 또한 소프트웨어가 어느 예외 엔트리/복귀 이벤트들을 보호할지 또는 보호하지 않을지를 결정하는 데 더 많은 융통성이 있는 이점을 가져서, 이들 동작들을 조합하는 것이 바람직하지 않을 수 있다.

[0027] 아래에서 논의되는 예들에서, 잠금 상태 또는 잠금 해제 상태로 설정될 수 있는 예외 복귀 상태 잠금 파라미터가 제공된다. 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도에 응답하여, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지를 검출하고, 예외 복귀 상태 잠금 파라미터가 잠금 상태에 있는 것으로 검출될 때 결함을 시그널링한다. 예외 복귀 상태 정보가 수정되는 것을 방지하기 위해 이 정보를 잠그기 위한 아키텍처 메커니즘을 제공함으로써, 이는 예외 엔트리와 GCS 데이터 구조로의 예외 복귀 상태의 푸시 사이

그리고 GCS-보호된 예외 복귀 상태 정보의 팝과 후속 예외 복귀 사이의 윈도우들에서 예외 복귀 상태 정보를 템퍼링하는 것을 방지하여, 예외 복귀 상태 정보의 GCS 기반 검사에 의해 제공되는 보안을 개선할 수 있다.

[0028] 프로세싱 회로부는 상이한 권한 레벨들과 연관된 다수의 예외 레벨들 중 하나에서 명령어들의 실행을 지원할 수 있다. 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는 예외가 취해지는 것에 응답하여 예외 복귀 상태 잠금 파라미터를 잠금 상태로 설정할 수 있고, 예외는 예외가 취해지기 전의 현재 예외 레벨과 동일한 예외 레벨에서 취해진다. 따라서, 예외가 취해지기 전에 실행되는 프로세싱과 동일한 예외 레벨에서 예외가 취해질 수 있도록, 예외 엔트리는, 캡처된 예외 복귀 상태 정보가 GCS 데이터 구조로 푸시될 수 있기 전에, 예외 엔트리에 응답하여 캡처된 예외 복귀 상태 정보의 템퍼링으로부터 보호하기 위해, 프로세싱 회로부로 하여금 수정에 대비하여 예외 복귀 상태 정보를 잠그게 할 수 있다. 예외 복귀 상태 잠금 파라미터를 잠금 상태로 설정하는 것은 예외 엔트리 후에 명시적 소프트웨어 명령어가 실행되어 예외 복귀 상태 잠금 파라미터를 잠금 상태로 설정할 필요 없이, 예외가 취해지는 것에 응답하여 자동으로 발생할 수 있다.

[0029] 예외가 취해지기 전의 현재 예외 레벨보다 더 많은 권한을 가진 예외 레벨에서 예외가 취해질 예외 엔트리에 대한 응답으로 예외 복귀 상태 잠금 파라미터를 잠금 상태로 설정하는 것은 필수가 아니다. 더 많은 권한을 가진 레벨로 예외가 취해지는 경우, 더 많은 권한을 가진 예외 핸들링 코드를 신뢰하여 예외 복귀를 적절하게 관리할 수 있다. 동일한 예외 레벨로 예외가 취해지는 경우 공격이 더 자주 발생한다. 그럼에도 불구하고, 다른 구현들은 예외를 더 많은 권한을 가진 예외 레벨로 취할 때 예외 복귀 상태 정보를 또한 잠그도록 선택할 수 있다.

[0030] 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는 GCS 예외 복귀 상태 푸시 명령어에 응답하여 예외 복귀 상태 잠금 파라미터를 잠금 해제 상태로 설정할 수 있다. 따라서, 일단 예외 복귀 상태 정보가 GCS 데이터 구조로 푸시되고 나면, 예외 복귀 상태 레지스터 저장소에서의 예외 복귀 상태 정보의 수정을 방지할 필요가 더 이상 없으며, 따라서 잠금 파라미터를 잠금 해제 상태로 스위칭함으로써, 이는 예외 복귀 상태 레지스터 저장소에서의 정보를 변경할 필요가 유효하게(validly) 있을 수도 있는 후속 동작들을 계속 진행할 수 있게 한다. GCS 예외 복귀 상태 푸시 명령어에 응답하여 예외 복귀 상태 잠금 파라미터를 잠금 해제하는 것은 또한 예외 복귀 상태 수정이 잠겨야 되는 시간에서 실행되는 명령어(예를 들어, 예외 복귀 명령어)가, 예외 복귀 상태 수정이 현재 잠금 해제되어 있는 시간에서 잘못된 프로그램 흐름의 부정확한 시퀀스로 실행되는 경우를 검출하게 함으로써 보안을 개선하는 데 도움이 된다.

[0031] 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는 GCS 예외 복귀 상태 팝 명령어에 응답하여 예외 복귀 상태 잠금 파라미터를 잠금 상태로 설정할 수 있다. GCS 예외 복귀 상태 팝 명령어는 GCS 데이터 구조로부터 팝된 예외 복귀 상태가 템퍼링에 취약한 윈도우의 시작을 표시할 수 있으므로, GCS 예외 복귀 상태 팝 명령어에 응답하여 예외 복귀 상태를 잠그는 것은 후속 예외 복귀 시에 예외 복귀에 사용되는 예외 복귀 상태 정보가 GCS 데이터 구조로부터 획득된 것과 여전히 동일하다는 확신을 주는 데 도움이 된다.

[0032] 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는, 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보에 기초하여 예외 복귀를 트리거하기 위한 예외 복귀 명령어에 응답하여 - 예외 복귀는 예외 복귀 전의 현재 예외 레벨과 동일한 예외 레벨에서의 프로세싱으로의 복귀임 -, 예외 복귀 상태 잠금 파라미터가 잠금 상태로 설정되는지 여부를 검사하고, 예외 복귀 상태 잠금 파라미터가 잠금 해제 상태로 설정된다고 검출하는 것에 응답하여 결함을 시그널링할 수 있다. 이는 GCS 예외 복귀 상태 팝 명령어를 사용하여 GCS-보호된 예외 복귀 상태 정보를 먼저 획득하지 않고도 예외 복귀 명령어가 실행되게 하도록 프로그램 흐름을 제어하려고 시도할 수 있는 공격으로부터 보호하는 데 도움이 될 수 있다.

[0033] 예외 복귀 상태 잠금 파라미터가 잠금 상태로 설정될 때 예외 복귀 상태 레지스터 저장소의 정보의 수정을 야기할 수 있는 명령어가 실행될 때 결함이 시그널링되어야 하는지 여부를 결정하기 위해 상이한 접근법들이 취해질 수 있다.

[0034] 하나의 예에서, 프로세싱 회로부의 적어도 하나의 동작 상태에서, 프로세싱 회로부는, 예외 복귀 상태 잠금 파라미터가 잠금 상태로 설정될 때, 예외 복귀 상태 레지스터 저장소에 기록될 새로운 값이 예외 복귀 상태 레지스터 저장소에 미리 저장된 오래된 값과 동일하거나 상이한지와 무관하게, 예외 복귀 상태 레지스터 저장소 내의 정보에 대한 수정을 요청하는 명령어를 실행하려는 시도에 응답하여 결함을 시그널링할 수 있다. 따라서, 이러한 경우에, 명령어가 이러한 수정이 잠겨 있을 때 예외 복귀 상태 레지스터 저장소에서의 정보를 수정하려고 시도하면, 명령어에 의해 기록될 새로운 값이 예외 복귀 상태 레지스터 저장소에 이전에 저장된 오래된 값과 동일했을 경우라도, 결함이 시그널링될 것이다. 새로운 값이 오래된 값과 동일한 경우 예외 복귀 상태 정보의 수정은 없을지라도, 결함을 트리거할지 여부를 결정하기 위해 예외 복귀 상태 레지스터 저장소에 현재 저장된

값을 읽을 필요가 없으므로, 결함의 시그널링을 오래된 값과 새로운 값 간의 임의의 비교와 독립적으로 만드는 것이 (예외 복귀 상태 잠금 파라미터에 따라 결함의 시그널링을 제어하기 위한 제어 회로 논리를 구현하고자 하는 하드웨어 설계자의 관점에서) 더 간단할 수 있다.

[0035] 또 다른 예에서, 프로세싱 회로부의 적어도 하나의 동작 상태에서, 예외 복귀 상태 잠금 파라미터가 잠금 상태로 설정될 때 예외 복귀 상태 레지스터 저장소 내의 정보에 대한 수정을 요청하는 명령어를 실행하려는 시도에 응답하여, 프로세싱 회로부는 예외 복귀 상태 레지스터 저장소에 기록될 새로운 값이 예외 복귀 상태 레지스터 저장소 내에 이전에 저장된 오래된 값과 동일하거나 상이한지 여부를 검사하고, 새로운 값이 오래된 값과 상이한 것으로 결정될 때 결함을 시그널링할 수 있다. 이 접근법은 예외 핸들러(exception handler)가 너무 짧아 메모리의 스택에 예외 복귀 상태를 저장하거나 메모리의 스택으로부터 예외 복귀 상태를 복원할 필요가 없는 어떤 빠른 예외 코드 경로들의 프로세싱 속도를 높이는 데 유용할 수 있다. 이 경우, 원래 캡처된 예외 복귀 상태가 예외 핸들러의 실행 전반에 걸쳐 예외 복귀 상태 레지스터 저장소에 유지될 수 있기 때문에, 예외 핸들러는 임의의 GCS 예외 복귀 상태 푸시/팝 명령어들을 실행하지 않는 것을 선호할 수 있다. 이는 예외 복귀 상태 잠금 파라미터가 예외 핸들러 전체에 걸쳐 잠금 상태로 유지될 수 있음을 의미할 수 있다. 예외 핸들러가 그럼에도 불구하고 예외 복귀 상태 레지스터 저장소를 수정하려고 시도하는 어떤 정당한 이유가 있을 수 있다. 따라서 잠금 파라미터가 잠금 상태에 있는 동안 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보를 수정하려는 시도가 있으면 결함이 트리거되어야 하지만, 때때로 시도된 수정은 실제로 예외 복귀 상태 레지스터 저장소에 이미 저장된 동일한 값을 기록할 수 있으며, 이 경우 예외 복귀 상태가 손상될 위험이 없다. 예를 들어, 위에서 논의된 바와 같이 고속 예외 코드 경로에 있기 때문에 예외 복귀 상태 정보를 변경하려고 의도하지 않았지만, 예외 핸들러 코드는 예외 복귀 상태 정보가 수정되어야 하는 다른 사용 사례와 공유되는 공통 코드 경로를 포함하고, 예외 복귀 상태 레지스터에 기록하는 명령어를 여하튼 실행하는 것(예를 들어, 조건 분기를 피함으로써 성능을 가속화하는 것)이 바람직하지만, 명령어가 고속 예외 코드 경로와 함께 사용되는 경우, 예외 복귀 상태 레지스터에 이전에 저장되었던 것과 동일한 값을 기록하도록 해당 명령어에 대한 피연산자를 설정하는 것이 바람직하다. 따라서, 수정 명령어가 예외 복귀 상태 레지스터 저장소에서의 값을 변경하지 않을 경우, 이 경우에 결함을 시그널링하지 않음으로써, 이는 결함을 시그널링함으로써 야기되는 추가된 레이턴시를 제거함으로써 성능을 향상시킬 수 있다(그렇지 않으면 현재의 프로세싱을 방해하고 또 다른 예외 핸들러가 실행되게 할 것임).

[0036] GCS 예외 복귀 상태 팝 명령어는 상이한 방식으로 구현될 수 있다.

[0037] 일부 예에서, GCS 예외 복귀 상태 팝 명령어는 GCS-보호된 예외 복귀 상태 정보를 레지스터 저장소에 간단히 팝할 수 있다(예외 복귀 상태 레지스터 저장소에 직접 또는 범용 레지스터에 팝할 수 있으며, 범용 레지스터로부터 후속 명령어가 GCS-보호된 예외 복귀 상태 정보를 예외 복귀 상태 레지스터 저장소로 이동시킬 수 있음). 그런 다음, GCS-보호된 예외 복귀 상태 정보는 후속 예외 복귀 명령어가 실행될 때 예외 복귀를 제어하는 데 직접 사용될 수 있다. 이 경우, GCS 예외 복귀 상태 푸시/팝 명령어들의 실행은, 내포된 예외로 인해 예외 복귀 상태 레지스터 저장소에서의 예외 복귀 상태가 겹쳐 쓰여지더라도 GCS 데이터 구조가 예외 복귀 상태를 추적하기에 충분하다는 것을 의미하기 때문에, 소프트웨어가 일반 메모리에서의 소프트웨어-관리(software-managed) 스택 구조로의 예외 복귀 상태 정보의 저장/복원을 관리할 필요가 없다.

[0038] 다른 예들에서, GCS-보호된 예외 복귀 상태는 예외 복귀를 직접 제어하는 데 사용되기 보다는, 예외 복귀 상태 레지스터 저장소에 저장된 의도된 예외 복귀 상태 정보가 후속 예외 복귀에 사용하기에 안전한지 여부를 검증하는 데 사용될 수 있다. 이 경우, 소프트웨어는 (예를 들어, 예외가 내포되어 있는 경우, 소프트웨어는 예외 복귀 상태를 비-GCS 메모리에 유지되는 스택 데이터 구조로 저장/복원하면서) 자신의 정상 메커니즘들을 사용하여 예외 복귀 상태 레지스터 저장소에서 예외 복귀 상태 정보를 관리할 수 있다. GCS 예외 복귀 상태 푸시/팝 명령어들은 소프트웨어에 의해 획득된 의도된 예외 복귀 상태 정보가 예외 복귀 제어에 사용하기 전에 이 정보가 검사될 수 있도록 예외 복귀 상태를 GCS 데이터 구조에 추가로 저장/복원한다.

[0039] 따라서, 이러한 예에서, GCS 데이터 구조로부터의 GCS-보호된 예외 복귀 상태 정보의 팝을 야기하는 것에 더하여, GCS 예외 복귀 상태 팝 명령어에 응답하여, 프로세싱 회로부는 또한 예외 복귀 상태 레지스터 저장소로부터 획득된 의도된 예외 복귀 상태 정보와 GCS 데이터 구조로부터 획득된 GCS-보호된 예외 복귀 상태 정보 사이에서 미스매치가 검출될 때 결함을 시그널링한다. 따라서, GCS-보호된 예외 복귀 상태 정보는 예외 복귀를 직접 제어하는 데 사용되는 것이 아니라 의도된 예외 복귀 상태 정보를 검증하는 데 사용된다.

[0040] 또 다른 예에서, 의도된 예외 복귀 상태 정보가 정확한지 여부를 검증하기 위해 GCS-보호된 예외 복귀 상태 정

보가 사용될 수 있지만, 이러한 검증은 GCS 예외 복귀 상태 팝 명령어 자체에 응답하여 발생하지 않을 수 있지만, 대신 후속 GCS 예외 복귀 상태 검증 명령어는 예외 복귀 상태 레지스터 저장소 내의 의도된 예외 복귀 상태 정보가 GCS 예외 복귀 상태 팝 명령어에 의해 획득된 이전에 팝된 GCS 예외 복귀 상태 정보와 일치하는지 여부를 검증할 수 있다.

[0041] 예외 복귀 상태 레지스터 저장소에 저장된 예외 복귀 상태 정보는 다양한 형태들을 취할 수 있다. 일반적으로, 예외 복귀 상태 레지스터 저장소는 예외 복귀 상태 정보를 저장하는 하나 이상의 레지스터들을 포함할 수 있다. 예를 들어, 예외 복귀 상태 레지스터 저장소는 예외로부터 복귀한 후에 실행될 명령어의 예외 복귀 어드레스를 저장하기 위한 예외 복귀 어드레스 레지스터 및/또는 예외로부터 복귀 시 복원될 프로세서 상태를 저장하기 위한 저장된 프로세서 상태 레지스터를 포함할 수 있다. 이러한 레지스터들을 중 하나 또는 그 둘 모두에 저장된 정보는 GCS 예외 복귀 상태 푸시 명령어에 응답하여 GCS 구조에 저장될 수 있다.

[0042] 저장된 프로세서 상태 레지스터는 예외를 취하는 것에 응답하여, 프로세싱 회로부의 현재 프로세서 상태가 프로세서 상태 레지스터로부터 저장된 프로세서 상태 레지스터로 복사되는 레지스터일 수 있다. 예외를 취하는 것이 현재 프로세서 상태를 변경하게 할 수 있기 때문에, 현재 프로세서 상태는 저장된 프로세서 상태 레지스터에 복사함으로써 캡처될 수 있고, 그 후 프로세싱 회로부는 예외가 취해질 상태에 따라 현재 프로세서 상태를 자유롭게 업데이트할 수 있다. 그런 다음, 소프트웨어에서 실행되는 예외 핸들러는 저장된 프로세서 상태 레지스터를 판독하고 캡처된 상태를 메모리에 저장하여, 이전의 예외의 핸들링을 선점하는(pre-empting) 추가 예외가 발생하는 경우에도 저장된 프로세서 상태가 유지되도록 보장할 수 있다. GCS 예외 상태 푸시 명령어를 사용하여, 그 저장된 프로세서 상태는 또한 GCS 데이터 구조로 푸시되어 대응하는 예외 복귀에서 예외 복귀 상태 정보의 검증을 가능하게 할 수 있다.

[0043] 예외 복귀 상태 잠금 파라미터가 프로세서 상태 레지스터에 명시된 파라미터인 것이 유용할 수 있다. 이것은, 예외가 취해지는 경우, 프로세서 상태 레지스터의 다른 콘텐츠와 함께, 예외를 취하는 것에 응답하여 프로세서 상태가 저장된 프로세서 상태 레지스터에 복사될 수 있기 때문에 예외 복귀 상태 잠금 파라미터 자체가 자동으로 저장될 것이다. 유사하게, 예외 복귀 명령어에 응답하여, 저장된 프로세서 상태 레지스터로부터 프로세서 상태 레지스터로 저장된 프로세서 상태를 복사하면 예외 복귀 상태 잠금 파라미터의 이전 값이 복원된다. 따라서, 이 접근법은, 예외 엔트리와 GCS 예외 복귀 상태 푸시 명령어 사이의 윈도우 동안 또는 GCS 예외 복귀 상태 팝 명령어와 예외 복귀 사이의 윈도우에서 새로운 예외가 취해지는 경우, 예외 복귀 상태 잠금 파라미터의 현재 상태(이들 윈도우들 동안 잠금 상태에 있어야 함)가 캡처되고, 따라서 저장된 프로세서 상태의 정상 복원의 일부로서 새로운 예외로부터 복귀할 때 복원되어, 새로운 예외가 취해진 시점에서 예외 복귀 상태 잠금 파라미터의 이전 상태로 복귀할 수 있다는 것을 의미한다. 따라서 공격자는 예외 복귀 상태 잠금 파라미터에 의해 제공되는 보호를 우회하려고 시도하는 메커니즘으로 새로운 예외를 트리거하는 것을 사용할 수 없다.

[0044] 예외 복귀 상태 잠금 파라미터에 기초한 예외 복귀 상태 레지스터 저장소에 대한 수정의 감시(policing)가 프로세싱 회로부의 모든 동작 상태들에서 사용되는 것이 필수는 아니다. 예를 들어, 하나 이상의 제어 레지스터들에 저장된 어떤 제어 상태 정보는 전술한 예외 복귀 상태 잠금 파라미터의 다양한 검사가 수행되어야 하는지 여부를 결정하는 데 사용될 수 있다. 이러한 검사를 디스에이블하면 예외 복귀 상태 잠금 파라미터를 지원하지 않은 레거시 시스템들에 대해 작성된 소프트웨어와의 역호환 가능성에 유용할 수 있다. 이러한 레거시 소프트웨어는 공격자가 ROP 공격을 시작하기 위해 악용할 수 있는 허점이 없도록 보장함에 있어 소프트웨어 개발자에 의한 더 큰 경계(vigilance)를 필요로 할 수 있다.

[0045] 따라서, 일례에서, 프로세싱 회로부의 적어도 하나의 동작 상태는, 예외-복귀-상태-잠금 이네이블 제어 파라미터가 예외 복귀 상태 잠금 파라미터를 사용하는 예외 복귀 상태 정보의 잠금이 이네이블됨을 표시하는 이네이블 상태로 설정되는 동작 상태를 포함한다. 예외-복귀-상태-잠금 이네이블 제어 파라미터가 디스에이블 상태로 설정되어 있으면 위에서 논의된 대로 예외 복귀 상태 잠금 파라미터를 업데이트하거나 검사할 필요가 없다.

[0046] 일부 예들에서, (GCS 예외 복귀 상태 푸시/팝 명령어들, GCS 데이터 구조와의 상호작용하는 다른 명령어들, 및 비인가된 액세스로부터 GCS 데이터 구조를 보호하기 위해 트리거되는 다양한 결함들과 같은) GCS 데이터 구조의 사용을 지원하는 아키텍처 메커니즘들은 GCS 이네이블 파라미터의 상태에 따라 완전히 디스에이블될 수 있다. 다시, 이것은 이러한 GCS 기반 아키텍처 메커니즘들을 지원하지 않은 레거시 시스템들용으로 작성된 소프트웨어를 지원할 수 있다. 따라서, (위에서 설명된 다양한 조치가 예외 복귀 상태 잠금 파라미터와 관련하여 취해진) 프로세싱 회로부의 적어도 하나의 동작 상태는, 예외-복귀-상태-잠금 이네이블 제어 파라미터가 예외 복귀 상태 잠금 파라미터를 사용하는 예외 복귀 상태 정보의 잠금이 이네이블됨을 표시하는 이네이블 상태로 설정되고;

GCS-이네이블 파라미터가 GCS 데이터 구조의 사용을 지원하기 위한 GCS 모드가 이네이블됨을 표시하는 이네이블 상태로 설정되는, 동작 상태를 포함할 수 있다.

[0047] 위에서 논의된 바와 같이, 메모리 관리 회로부는 타겟 어드레스에 대응하는 메모리 영역이 GCS 영역으로서 명시되고 기록 요청이 GCS 영역에 액세스하도록 허용되는 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청 이외의 요청이라고 결정하는 것에 응답하여 타겟 어드레스를 명시하는 기록 요청을 거부할 수 있다. 메모리 관리 회로부는 또한 메모리 액세스 요청이 GCS 메모리 액세스 요청이고 타겟 어드레스에 대응하는 메모리 영역이 GCS 영역이 아니라고 결정하는 것에 응답하여 메모리 액세스 요청을 거부할 수 있다. 이는 GCS 메모리 액세스 요청들이 GCS 영역들에 액세스하기 위해서만 사용될 수 있고 다른 데이터를 저장하기 위해 사용되는 정규 메모리 영역들에 액세스하기 위해서는 사용될 수 없다는 것을 의미하기 때문에, 공격자에게 이용 가능한 공격 표면을 감소시킨다. 이는, GCS 메모리 액세스 요청들이 비-GCS 영역에 액세스하게 하기 위해, GCS-액세스 타입의 명령어의 어드레스 피연산자가 공격자에 의해 수정되면(예를 들어, 공격자가 프로그램 흐름의 부정확한 시퀀스를 강요함으로써 부정확하게 야기됨), 이는 오류로서 검출되고 메모리에 액세스하기 위한 요청이 거부될 것임을 의미한다. 이는 공격자가 GCS 데이터 구조를 또 다른 메모리 영역으로 대체함으로써 GCS-보호된 예외 복귀 상태를 위조할 수 있는 것을 방지함으로써 보안을 향상시킨다.

[0048] GCS 데이터 구조는 ROP 공격에 대해 예외 복귀 상태 정보를 보호하기 위해 사용되는 것으로 위에서 설명되지만, 동일한 데이터 구조는 또한 함수 복귀 어드레스와 같은 함수 복귀 상태 정보를 보호하기 위해 사용될 수 있다. GCS 데이터 구조로부터 획득된 정보가 예외 복귀 상태 정보인지 또는 함수 복귀 상태 정보인지를 검출하기 위한 메커니즘이 제공될 수 있어서(예를 들어, 이들 정보가 구별될 수 있도록 유효 예외 복귀 상태 레코드와 유효 함수 복귀 상태 레코드의 인코딩들을 구별함으로써), GCS 예외 복귀 상태 검증 명령어는 프로세싱 회로부로 하여금 GCS-보호된 예외 복귀 상태 정보로서 사용되는 획득된 GCS 레코드가 유효 예외 복귀 상태 레코드에 대응하는지 여부를 검사하게 하고, 따라서 예외 복귀에 대한 예외 복귀 상태 정보를 검증하기 위해 함수 복귀 상태 레코드를 사용함으로써 부정확한 거동을 야기하려는 시도들에 대해 감시하게 할 수 있다.

[0049] 다른 예들에서, GCS 데이터 구조는 예외 복귀 상태 정보를 보호하기 위해 사용될 수 있지만, 함수 복귀 상태 정보를 보호하기 위해서는 사용될 수는 없다(예를 들어, ROP 공격에 대해 함수 복귀 상태를 보호하기 위해 상이한 메커니즘이 사용될 수 있거나, 또는 함수 복귀 상태 정보에 대해 아키텍처 레벨에서 ROP 공격에 대해 어떠한 보호도 제공되지 않을 수 있음(대신, 함수 복귀에 대한 ROP 공격에 대한 방어에 대한 책임이 소프트웨어 개발자에게 있을 수 있음)). 따라서, GCS 데이터 구조가 예외 복귀 상태와 함수 복귀 상태 사이에서 공유되는 것이 필수적인 것은 아니다.

[0050] 위에서 논의된 기법들은 위에서 논의된 프로세싱 회로부와 메모리 관리 회로부를 구현하기 위해 제공되는 하드웨어 회로부를 갖는 데이터 프로세싱 장치 내에서 구현될 수 있다.

[0051] 그러나, 동일한 기법은 또한 타겟 코드의 실행을 위한 명령어 실행 환경을 제공하기 위해 호스트 데이터 프로세싱 장치 상에서 실행되는 컴퓨터 프로그램 내에서 구현될 수 있다. 그러한 컴퓨터 프로그램은, 소정의 명령어 세트 아키텍처에 따라 타겟 코드를 실제로 지원하는 하드웨어 장치 상에서 제공될 아키텍처 환경을, 심지어 호스트 데이터 프로세싱 장치 그 자체가 그 아키텍처를 지원하지 않더라도, 시뮬레이팅하도록 호스트 데이터 프로세싱 장치를 제어할 수도 있다. 컴퓨터 프로그램은 프로세싱 프로그램 논리, 레지스터 에뮬레이팅 프로그램 논리 및 프로세싱 회로부의 기능들을 에뮬레이팅하는 메모리 관리 프로그램 논리, GCS 예외 복귀 상태 푸시 명령어 및 GCS 예외 복귀 상태 팝 명령어에 대한 지원을 포함하는, 위에서 논의된 예외 복귀 상태 레지스터 저장소 및 메모리 관리 회로부, 및 위에서 논의된 바와 같은 예외 복귀 상태 잠금 파라미터를 가질 수도 있다. 그러한 시뮬레이션 프로그램은, 예를 들어, 하나의 명령어 세트 아키텍처에 대해 작성된 레거시 코드가 상이한 명령어 세트 아키텍처를 지원하는 호스트 프로세서 상에서 실행되고 있을 때, 유용할 수 있다. 또한, 시뮬레이션은 그 새로운 아키텍처 버전을 지원하는 프로세싱 하드웨어가 준비되기 전에 더 새로운 버전의 명령어 세트 아키텍처에 대한 소프트웨어 개발이 시작될 수 있게 할 수 있는데, 왜냐하면 시뮬레이팅된 실행 환경 상에서의 소프트웨어의 실행은 새로운 아키텍처를 지원하는 하드웨어 디바이스들의 진행 중인 개발과 병행하여 소프트웨어의 테스트를 가능하게 할 수 있기 때문이다. 시뮬레이션 프로그램은 비일시적 저장 매체일 수도 있는 저장 매체 상에서 저장될 수도 있다.

[0052] 데이터 프로세싱 장치의 구체적인 예

[0053] 도 1은 데이터 프로세싱 장치(2)의 예를 개략적으로 예시한다. 데이터 프로세싱 장치는 다수의 파이프라인 스테이지들을 포함하는 프로세싱 파이프라인(4)을 갖는다. 이 예에서, 파이프라인 스테이지들은 명령어 캐시(8)

로부터 명령어들을 페치하기 위한 페치 스테이지(6); 파이프라인의 나머지 스테이지들에 의해 프로세싱될 마이크로 동작들(디코딩된 명령어들)을 생성하기 위해 페치된 프로그램 명령어들을 디코딩하기 위한 디코드 스테이지(10); 마이크로 동작들을 위해 요구되는 피연산자가 레지스터 파일(14)에서 이용 가능한지 여부를 검사하고, 일단 주어진 마이크로 동작들을 위해 요구되는 피연산자들이 이용 가능하면 실행을 위한 마이크로 동작을 발행하기 위한 발행 스테이지(12); 결과 값들을 생성하기 위해 레지스터 파일(14)로부터 판독된 피연산자들을 프로세싱함으로써, 마이크로 동작들에 대응하는 데이터 프로세싱 동작들을 실행하기 위한 실행 스테이지(16); 및 프로세싱의 결과들을 레지스터 파일(14)에 라이트백하기 위한 라이트백 스테이지(18)를 포함한다. 이것은 가능한 파이프라인 아키텍처의 단지 하나의 예에 불과하며, 다른 시스템들은 추가적인 스테이지들 또는 상이한 구성의 스테이지들을 가질 수도 있다는 것이 인식될 것이다. 예를 들어, 비순차적 프로세서(out-of-order processor)에서, 프로그램 명령어들 또는 마이크로 동작들에 의해 명시되는 아키텍처 레지스터들을 레지스터 파일(14) 내의 물리적 레지스터들을 식별하는 물리적 레지스터 명시자들(physical register specifier)에 매핑하기 위해 레지스터 리네이밍 스테이지(register renaming stage)가 포함될 수 있다. 일부 예들에서, 디코드 스테이지(10)에 의해 디코딩된 프로그램 명령어들과 실행 스테이지에 의해 프로세싱된 대응하는 마이크로 동작들 사이에 1대 1 관계가 있을 수 있다. 프로그램 명령어들과 마이크로 동작들 사이에 일대다 또는 다대일 관계가 있는 것이 또한 가능하며, 따라서, 예를 들어, 단일 프로그램 명령어가 2개 이상의 마이크로 동작들로 분할될 수 있거나, 2개 이상의 프로그램 명령어들이 단일 마이크로 동작으로서 프로세싱되도록 융합될 수 있다.

[0054] 실행 스테이지(16)는 상이한 부류들의 프로세싱 동작을 실행하기 위한 다수의 프로세싱 유닛들을 포함한다. 예를 들어, 실행 유닛들은 레지스터들(14)로부터 판독된 스칼라 피연산자들에 대해 산술 동작 또는 논리 동작들을 수행하기 위한 스칼라 산술/논리 유닛(ALU)(20); 부동 소수점 값들에 대해 동작들을 수행하기 위한 부동 소수점 유닛(22); 분기 동작들의 결과를 평가하고 그에 따라 현재의 실행 지점을 나타내는 프로그램 카운터를 조정하기 위한 분기 유닛(24); 및 메모리 시스템(8, 30, 32, 34)에서 데이터에 액세스하기 위한 로드/저장 동작들을 수행하기 위한 로드/저장 유닛(26)을 포함할 수도 있다. 메모리 관리 회로부의 한 예인 메모리 관리 유닛(memory management unit; MMU)(28)은 데이터 액세스 명령어들의 피연산자 및 메모리 시스템에서의 데이터의 저장 위치들을 식별하는 물리적 어드레스들에 기초하여 로드/저장 유닛(26)에 의해 명시되는 가상 어드레스들 사이의 어드레스 변환을 수행하기 위해 제공된다. MMU는 메모리 시스템에 저장되는 페이지 테이블로부터 어드레스 변환 데이터를 캐싱하기 위한 변환 색인 버퍼(translation lookaside buffer; TLB)(29)를 구비하는데, 여기서 페이지 테이블의 페이지 테이블 엔트리들은, 예를 들어, 파이프라인 상에서 실행되는 주어진 프로세스가 주어진 메모리 영역으로부터 명령어들을 판독, 기록 또는 실행하는 것이 허용되는지의 여부를 지배하는 액세스 허가 및 어드레스 변환 매핑을 정의한다.

[0055] 이 예에서, 메모리 시스템은 레벨 1 데이터 캐시(30), 레벨 1 명령어 캐시(8), 공유 레벨 2 캐시(32) 및 메인 시스템 메모리(34)를 포함한다. 이것은 가능한 메모리 계층구조의 일례일 뿐이며, 캐시들의 다른 배열들이 제공될 수 있다는 것이 인식될 것이다. 실행 스테이지(16)에 도시된 특정 타입들의 프로세싱 유닛(20 내지 26)은 일례일 뿐이며, 다른 구현들은 상이한 세트의 프로세싱 유닛들을 가질 수 있거나, 동일 타입의 다수의 마이크로 동작들이 병렬로 핸들링될 수 있도록 동일 타입의 프로세싱 유닛의 다수의 인스턴스들을 포함할 수 있다. 도 1은 단지 가능한 프로세서 파이프라인 구현의 일부 컴포넌트들의 단순화된 표현에 불과하며, 프로세서는 간결성을 위해 예시되지 않은 많은 다른 요소들을 포함할 수도 있다는 것이 인식될 것이다. 도 1이 메모리(34)에 대한 액세스를 갖는 단일의 프로세서 코어를 도시하지만, 장치(2)는, 각각의 코어가 각각의 캐시(8, 30, 32)를 가지면서, 메모리(34)에 대한 액세스를 공유하는 하나 이상의 추가적인 프로세서 코어들을 또한 가질 수 있다.

[0056] 장치는 또한 예외의 핸들링을 제어하기 위한 예외 핸들링 회로부(40)를 포함한다. 예외 신호(42)에 응답하여, 예외 핸들링 회로부(40)는 프로세싱이 파이프라인(4)에 의해 수행되는 것을 중단시키고, 예외 신호(42)에 의해 표시된 예외 타입에 대응하는 예외 핸들러의 프로세싱으로 스위칭하도록 파이프라인(4)을 제어한다. 예를 들어, 예외는 프로세싱 파이프라인(4)에 의해 소프트웨어가 실행되는 것에 의해 직접 야기되지 않는 외부 이벤트에 의해 트리거되는 인터럽트, 예를 들어, 사용자가 프로세싱 장치를 포함하는 전자 디바이스 상의 버튼을 누르는 것과 같은 외부 이벤트를 시그널링하는 인터럽트, 프로세서 리셋이 어서트되는 것, 네트워크 인터페이스 또는 다른 입력/출력 디바이스로부터의 메시지의 수신 등일 수 있다. 또한, 예외는 명령어들이 파이프라인(4)에 의해 실행되는 것에 의해 직접 야기되는 소프트웨어 트리거형 예외(software-triggered exception)일 수 있다. 예를 들어, 더 많은 권한을 가진 실행 상태로 스위치를 트리거하기 위해 "감독자 호출" 인터럽트가 취해지게 하기 위한 명령어를 실행함으로써, 소프트웨어- 트리거형 예외가 소프트웨어에 의해 자발적으로 트리거될 수 있다. 대안적으로, 소프트웨어-트리거형 예외는 소프트웨어에 의해 비자발적으로 트리거되는 예외일 수 있으며, 이는 정의되지 않은 명령어 오류 또는 어드레스 결함과 같이 소프트웨어에 의해 실행되는 적어도 하나

의 명령어에서 발생하는 오류로 인해 발생한다.

[0057] 도 2는, 명령어들을 실행할 때 프로세싱 회로부(4)가 동작할 수 있는 상이한 실행 상태들(예외 레벨들로서 또한 지칭됨)을 예시하는 다이어그램이다. 이 예에서는, 4개의 예외 레벨들(EL0, EL1, EL2, EL3)이 있는데, 여기서 예외 레벨(EL0)은 최소 권한을 갖는 예외 레벨이고 예외 레벨(EL3)은 최대 권한을 갖는 예외 레벨이다. 일반적으로, 프로세싱 회로부는, 더 많은 권한을 갖는 예외 레벨에서 실행될 때, 더 낮은, 더 적은 권한을 갖는 예외 레벨들이 액세스할 수 없는 일부 메모리 위치들 또는 레지스터들(14)에 대한 액세스를 가질 수도 있다.

[0058] 이 예에서, 예외 레벨(EL0)은 예외 레벨(EL1)에서 실행되는 대응하는 운영 체제 또는 가상 기계들에 의해 관리되는 애플리케이션들을 실행하기 위한 것이다. 다수의 가상 기계들이 동일한 물리적 플랫폼 상에서 공존하는 경우, 그러면, EL2에서 동작되는 하이퍼바이저가 제공되어, 각각의 가상 기계들을 관리할 수도 있다. 하이퍼바이저가 가상 기계들을 관리하고 가상 기계들이 애플리케이션들을 관리하는 예들을 도 2가 도시하지만, 하이퍼바이저가 EL0에서 애플리케이션들을 직접적으로 관리하는 것도 또한 가능하다.

[0059] 필수적인 것은 아니지만, 일부 구현들은 프로세싱 회로부에 대해 별개의 하드웨어로 분할된 보안 및 비보안 동작 도메인을 구현할 수도 있다. 데이터 프로세싱 시스템(2)은, 보안 도메인에서 동작하는 소프트웨어 프로세스들과 관련되는 데이터 및 코드가 비보안 도메인에서 동작하는 프로세스들에 의한 액세스로부터 격리되는 것을 보장하기 위해, 프로세서 및 메모리 시스템 내에 구현되는 하드웨어 특징들을 가질 수도 있다. 예를 들어, 영국(UK) 케임브리지 소재의 Arm® Limited(암 리미티드)에 의해 제공되는 TrustZone®(트러스트존) 아키텍처와 같은 하드웨어 아키텍처가 사용될 수도 있다. 대안적으로, 다른 하드웨어 강제 보안 구획화 아키텍처(hardware enforced security partitioning architecture)가 사용될 수 있다. 보안 애플리케이션들(신뢰되는 서비스들)은 보안 도메인의 예외 레벨(EL0)에서 동작될 수도 있고 보안(신뢰되는) 운영 체제 또는 가상 기계들은 보안 도메인의 예외 레벨(EL1)에서 동작될 수도 있다. 일부 구현들에서, 보안 상태에서 EL2에 대한 지원이 없으며 하이퍼바이저는 오로지 비보안 EL2에서만 실행될 수도 있다. 다른 구현들에서, 도 2의 별표에 의해 나타내어지는 바와 같이 보안 EL2에서 실행되는 보안 하이퍼바이저에 대한 지원이 있을 수도 있다. 일부 예들에서, 예외 레벨(EL3)에서 실행되는, 비보안 도메인과 보안 도메인 사이의 이행을 관리하기 위한 보안 모니터 프로그램이 제공될 수도 있다. 다른 구현들은, 보안 모니터 프로그램이 필요로 되지 않을 수도 있도록, 보안 도메인 사이의 이행을 하드웨어에서 단속할 수 있다.

[0060] 현재 예외 레벨 ELx에서 프로세싱하는 동안 예외가 취해지면, 동일한 예외 레벨 ELx 또는 더 많은 권한을 가진 예외 레벨 ELy(여기서 y > x임)에서 예외가 취해질 수 있다. 예외 타입과 연관된 특성은 예외에 대해 타겟 예외 레벨을 나타낼 수 있다.

[0061] 어드레스 변환

[0062] MMU(28)에 의해 수행되는 하나의 태스크는 가상 어드레스들(virtual address; VA)과 물리적 어드레스들(physical address; PA) 사이의 어드레스 변환이다. 프로세싱 회로부(4) 상에서 실행되는 소프트웨어는 가상 어드레스들을 사용하여 메모리 위치들을 명시하지만, 그러나 이들 가상 어드레스들은 MMU(28)에 의해 액세스할 메모리 시스템 위치를 식별하는 물리적 어드레스들로 변환될 수 있다. 가상 어드레스들을 사용하는 것의 이점은, 관리 소프트웨어, 예컨대 운영 체제(Operating System; OS)가 소프트웨어에 제공되는 메모리의 모습을 제어하는 것을 허용한다는 것이다. OS는, 어떤 메모리가 보이는지, 그 메모리가 보이는 가상 어드레스, 및 그 메모리에 어떤 액세스가 허락되는지를 제어할 수 있다. 이것은, OS가 애플리케이션들을 샌드박스화하는(sandbox) 것(하나의 애플리케이션의 리소스들을 또 다른 애플리케이션에게 숨김) 및 기저의 하드웨어로부터 추상화를 제공하는 것을 허용한다. 가상 어드레스들을 사용하는 것의 또 다른 이점은, OS가 메모리의 다수의 조각의 물리적 영역들을 단일의 연속적인 가상 어드레스 공간으로서 애플리케이션에 제공할 수 있다는 것이다. 가상 어드레스들은, 그들의 애플리케이션을 작성할 때 시스템의 정확한 메모리 어드레스들을 알지 못할 소프트웨어 개발자에게도 또한 도움이 된다. 가상 어드레스들을 통해, 소프트웨어 개발자는 그들 자신이 물리적 메모리에 관여할 필요가 없다. 애플리케이션은, 어드레스 변환을 수행하는 것이 함께 작동할 OS 및 하드웨어에 달려 있다는 것을 알고 있다.

[0063] 실제로, 각각의 애플리케이션은 물리적 시스템의 상이한 위치들로 매핑될 그 자신의 가상 어드레스들의 세트를 사용할 수 있다. 운영 체제가 상이한 애플리케이션들 사이를 스위칭할 때, 그것은 맵을 다시 프로그래밍한다. 이것은 본 출원의 가상 어드레스들이 메모리의 올바른 물리적 위치로 매핑될 것이다는 것을 의미한다.

[0064] 가상 어드레스들은 매핑을 통해 물리적 어드레스들로 변환된다. 가상 어드레스들과 물리적 어드레스들 사이의

매핑은 변환 테이블(페이지 테이블로서 종종 지칭됨)에 저장된다. 변환 테이블은 메모리에 저장되며 소프트웨어, 전형적으로 OS 또는 하이퍼바이저에 의해 관리된다. 변환 테이블은 정적이지 않으며, 소프트웨어의 필요성이 변경됨에 따라 테이블은 업데이트될 수 있다. 이것은 가상 어드레스들과 물리적 어드레스들 사이의 매핑을 변경한다.

[0065] 프로세싱 회로부(4)가 실행 상태들의 소정의 서브세트에 있을 때(특히, 프로세싱 회로부(4)가 비보안 EL0 또는 비보안 EL1에 있을 때) 수행되는 메모리 액세스의 경우, 도 3에서 도시되는 바와 같이 2 스테이지 어드레스 변환이 사용된다(다른 실행 상태들의 경우, 스테이지 1 페이지 테이블을 사용하는 어드레스 변환의 하나의 스테이지가 충분함). 그러므로, 비보안 EL0 및 비보안 EL1로부터의 가상 어드레스들은 테이블의 두 세트들을 사용하여 변환된다. 이들 테이블은 가상화를 지원하며, 주어진 가상 기계(virtual machine; VM)(게스트 운영 체제에 대응하는 가상 기계 및 그 게스트 운영 체제에 의해 제어되는 애플리케이션들)에 의해 보이는 물리적 메모리의 모습을 하이퍼바이저가 가상화하는 것을 허용한다. OS에 의해 제어되는 변환의 세트를 스테이지 1로 칭한다. 스테이지 1 테이블은 가상 어드레스들을 중간 물리적 어드레스들(intermediate physical address; IPA)로 변환한다. 스테이지 1에서, OS는 IPA가 물리적 어드레스 공간들이라고 생각한다. 그러나, 하이퍼바이저는 스테이지 2로 칭해지는 변환의 제2 세트를 제어한다. 이 변환의 제2 세트는 IPA를 물리적 어드레스들로 변환한다.

[0066] 스테이지 1 및 스테이지 2 변환 테이블은, 스테이지 1 및 스테이지 2에 대해 각각 다수의 레벨들의 변환 테이블을 포함하는 계층적 테이블 구조들로서 구현된다. 스테이지 1 및 스테이지 2 테이블 둘 모두는 최대 4 개의 레벨들의 페이지 테이블, 즉, 레벨 0(L0), 레벨 1(L1), 레벨 2(L2) 및 레벨 3(L3)을 가질 수 있다.

[0067] 주어진 어드레스에 대한 물리적 어드레스 매핑을 찾아내기 위해, 하나 이상의 변환 테이블 룩업을 포함하는 변환 테이블 워크가 수행된다. 변환 테이블 워크는 가상 어드레스를 물리적 어드레스로 변환하는 데 필요한 룩업의 세트이다. 비보안 EL1&0 변환 체제(regime)의 경우, 이 세트는 스테이지 1 변환 및 스테이지 2 변환 둘 모두에 대한 룩업을 포함한다. 스테이지 1 및 스테이지 2 룩업을 사용하여 성공적인 변환 테이블 워크에 의해 반환되는 정보는 다음의 것이다:

[0068] • 필요한 물리적 어드레스(중간 어드레스로의 스테이지 1 매핑 및 물리적 어드레스로의 스테이지 2 매핑에 기초하여 변환됨).

[0069] • 타겟 메모리 영역에 대한 액세스 허가 및/또는 메모리 속성들, 이것은 그 메모리 영역에 대한 액세스를 제어하는 방법에 관한 정보를 제공함. 이들은 스테이지 1 테이블 구조에서 정의되는 스테이지 1 액세스 허가 및/또는 속성들 및 스테이지 2 테이블 구조에서 정의되는 스테이지 2 액세스 허가 및/또는 속성들을 포함할 수도 있다.

[0070] 스테이지 1 및 스테이지 2 구조들 중 하나의 주어진 구조를 순회하기 위해, 워크는, 변환 테이블 기본 어드레스 레지스터에서 명시되는 어드레스(스테이지 1의 경우 TTBR, 스테이지 2의 경우 VTTBR_EL2)에 기초하여, 초기 룩업을 위한 최상위 레벨(L0) 변환 테이블의 판독으로 시작된다. 각각의 변환 테이블 룩업은 다음의 것 중 하나를 나타내는 디스크립터를 반환한다:

[0071] • 엔트리는, 추구되고 있는 어드레스 매핑을 제공하는 스테이지 1 또는 스테이지 2 구조의 순회의 최종 엔트리이다. 엔트리가 최종 L3에 있는 경우, 이 엔트리는 페이지 디스크립터(D_Page)로 칭해지며, 한편 워크의 최종 엔트리를 제공하는 엔트리가 더 높은 레벨들 중 하나에 있는 경우, 그것은 블록 디스크립터(D_Block)로 칭해진다. 순회의 최종 엔트리는 출력 어드레스(output address; OA - 즉, 스테이지 1의 경우 IPA 또는 스테이지 2의 경우 PA), 및 액세스에 대한 허가 및 속성들을 포함한다. 블록 디스크립터가 변환 테이블 구조의 더 높은 레벨에서 발견되는 경우, 이것은 블록 디스크립터가 L3에서 단일의 엔트리에 의해 표현되는 4kB 메모리 페이지보다 더 큰 크기들의 메모리 영역을 나타낸다는 것을 의미한다(L1 및 L2에서 블록 디스크립터에 의해 표현되는 특정한 사이즈는 L1 또는 L2 테이블로 인덱싱하기 위해 사용되는 인덱스 비트들의 수에 좌우됨 - 이 예에서 L1 및 L2 블록 디스크립터는 1GB 및 2MB 영역들을 각각 나타냄).

[0072] • 추가적인 레벨의 룩업이 요구된다. 이 경우, 엔트리는 테이블 디스크립터(D_Table)로 칭해지는데, 그 이유는, 그것이 테이블의 추가적인 레벨에서 그 룩업을 위한 변환 테이블 기본 어드레스를 제공하기 때문이다. 테이블 디스크립터는 선택적으로 최종 변환에 적용될 수 있는 다른 계층적 속성들을 또한 제공할 수 있다. 레벨 1 및 2에서의 변환 테이블 엔트리들의 인코딩은 블록 디스크립터를 테이블 디스크립터와 구별한다.

[0073] • 디스크립터는 무효하다. 이 경우, 메모리 액세스는 변환 결함을 생성한다.

[0074] 레지스터 상태

[0075] 도 4는 예외 복귀를 핸들링하기 위해 사용되는 아키텍처 상태를 저장하고 템퍼링으로부터 예외 복귀 상태를 보호하기 위한 레지스터들을 포함하는, 데이터 프로세싱 장치의 레지스터들(14)의 서브세트의 예를 도시한다. 레지스터들(14)은 범용 피연산자 및 데이터 프로세싱 명령어들 및 로드/저장 명령어들의 결과들을 저장하기 위한 범용 레지스터들(general purpose register; GPR)(50)의 세트를 포함한다. 프로그램 카운터(program counter; PC) 레지스터(52)는 프로그램 흐름의 현재 포인트를 나타내는 명령어의 어드레스를 나타내는 프로그램 카운터 값을 저장하도록 제공된다. 프로세서 상태(processor state; PSTATE) 레지스터(54)는 프로세싱 회로부(4)의 현재 상태를 나타내는 하나 이상의 값들을 저장한다. 예를 들어, PSTATE 레지스터(54)는 현재 예외 레벨(EL)의 표시(56)를 포함한다. PSTATE 레지스터(54)는 또한, 디코드 스테이지(10)에서의 명령어들을 디코딩하기 위해 사용될 현재 명령어 세트의 표시들, 현재 보안 상태, 프로세싱 회로부(4)가 동작하고 있는 프로세서 모드의 다른 표시들뿐만 아니라 후속 조건부 명령어에 대해 테스트 조건이 만족되는지 여부를 평가하는 데 사용될 수 있는 이전 프로세싱 명령어들의 결과를 표시하는 데 사용되는 조건 코드 플래그를 포함할 수 있다(조건부 명령어는 테스트 조건이 충족되는 것에 대해 조건부로 함수를 실행하도록 프로세싱 회로부를 제어함). 또한, 아래에서 추가로 논의되는 바와 같이, PSTATE 레지스터(54)는 업데이트을 방지하기 위해 예외 복귀 상태 정보를 잠그기 위한 예외 복귀 상태 잠금 파라미터(EXLOCK)(58)를 포함한다.

[0076] 레지스터들(14)은 이 예에서 저장된 프로세서 상태 레지스터(SPSR)(60) 및 예외 링크 레지스터(ELR)(62)를 포함하는 다수의 예외 복귀 레지스터들을 포함한다. 이 예에서, SPSR(60) 및 ELR(62)은 뱅크된(banked) 레지스터들이며, 이는 이들 레지스터들의 다수의 별개의 인스턴스들이 상이한 예외 레벨들(EL1 내지 EL3)에 대응하여 제공됨을 의미한다. 주어진 예외 레벨(ELx)(여기서, ELx는 EL1, EL2 및 EL3 중 하나임)에서 핸들링될 예외가 취해질 때, 프로그램 카운터 레지스터(52)에 저장된 현재 프로그램 카운터는 ELR_ELx(62)에 복사되고, PSTATE 레지스터(54)에 저장된 현재 프로세서 상태는 SPSR_ELx(60)에 복사된다(즉, 주어진 예외 레벨(ELx)에 대응하는 ELR(62) 및 SPSR(60)의 버전에 복사됨). 프로그램 카운터의 복사된 값은 예외 복귀 어드레스라고 할 수 있다. 현재 프로세서 상태의 복사된 값은 저장된 프로세서 상태라고 할 수 있다. 예외 복귀 어드레스 및 저장된 프로세서 상태는 집합적으로 예외 복귀 상태 정보라고 할 수 있다. 주어진 예외 레벨(ELx)에서 프로세싱된 예외로부터의 예외 복귀 시, 저장된 프로세서 상태는 SPSR_ELx(60)로부터 PSTATE 레지스터(54)로 복사되고, 예외 복귀 어드레스는 ELR_ELx(62)로부터 프로그램 카운터 레지스터(52)로 복사되어, 예외가 취해졌을 때 프로세서가 있었던 상태로 프로세서를 복원한다. 레지스터 접미사(ELx)는 이러한 레지스터들에 기록하도록 명령어들이 허용되는 가장 적은 권한을 가진 예외 레벨이 예외(ELx)임을 나타낸다. 따라서, 예를 들어, SPSR_EL2는 EL2 또는 EL3에 있을 때 기록 가능하지만 EL0 또는 EL1에 있을 때는 기록 가능하지 않을 것이다.

[0077] 레지스터들(14)은 또한 보호형 제어 스택 포인터 레지스터(GCSPR)(64)를 포함하며, 이는 예외 레벨들(EL1, EL2, EL3) 각각과 연관된 버전(GCSPR_ELx)이 존재하도록 예외 레벨마다 다시 뱅크된다. GCS 포인터 레지스터(64)는 예외 시에 SPSR(60) 및 ELR(62)에 저장된 예외 복귀 상태를 보호하기 위해 사용되는 보호형 제어 스택(GCS)이라고 하는, 메모리 내의 데이터 구조를 가리키는 스택 포인터를 저장하기 위해 사용된다. 이것은 하기에서 더 자세히 논의된다. 또한, 레지스터들은 GCS 제어 레지스터(GCSCR)(66)를 포함하며, 이는 또한 예외 레벨들(EL1, EL2, EL3)에 대응하는 버전(GCSCR_ELx)을 제공하기 위해 예외 레벨마다 뱅크된다. 프로세싱 회로부(4)가 주어진 예외 레벨(ELx)에서 동작할 때, 대응하는 스택 포인터 및 GCS 제어 레지스터들(GCSPR_E1x 및 GCSCR_ELx)은 GCS 데이터 구조의 사용을 제어하기 위한 스택 포인터 및 제어 상태를 제공한다. GCS 제어 레지스터(66)는 적어도 GCS 데이터 구조의 사용을 보호하기 위한 아키텍처 측정이 이네이블되는지 또는 디스에이블되는지를 제어하기 위한 GCS 이네이블 파라미터(68), 및 예외 복귀 레지스터들(60, 62)의 수정을 방지하기 위해 예외 복귀 상태 잠금 파라미터(EXLOCK)(58)의 사용이 이네이블되는지 또는 디스에이블되는지를 제어하기 위한 예외-복귀-상태 잠금 이네이블 파라미터(EXLOCKEN)(69)를 포함한다. GCS 제어 레지스터(66)는 또한, GCS 데이터 구조의 사용이 이네이블 파라미터(68)에 의해 표시된 바와 같이 이네이블되는 GCS 모드에 있을 때 프로세싱 회로부의 거동을 제어하기 위한 다른 상태 정보를 저장할 수 있다는 것이 이해될 것이다.

[0078] 도 4는 레지스터들(60, 62, 64, 66)이 프로세싱 회로부(4)의 상이한 예외 상태들에 대응하는 이들 레지스터들의 다수의 버전을 제공하기 위해 뱅크되는 예를 도시하지만, 이는 필수적이지 않으며, 다른 예들에서 이들 레지스터들 중 임의의 것의 단일 버전이 제공될 수 있고, 예외 상태들 사이에서 공유된다. 그 경우, 소프트웨어는 주어진 예외 상태에서 공유 레지스터에 저장된 값이 상이한 예외 상태에서 공유 레지스터에 저장된 값과 다를 수

있도록 예외 상태들 사이에서 스위칭할 때 공유 레지스터에 저장된 값을 스위칭할 수 있다. 레지스터들(60, 62, 64, 66)의 또 다른 서브세트가 예외 상태들 사이에서 공유되는 동안, 레지스터들(60, 62, 64, 66)의 서브세트가 뱅크되는 것이 또한 가능하다.

[0079] 도 5는 GCS의 사용을 지원하지 않는 시스템에서 예외 엔트리 및 복귀 시 실행되는 명령어들의 예를 예시한다. 도 5의 좌측은 예외 엔트리 시퀀스를 예시한다. 예외 핸들링 회로부(40)에 의해 예외를 취하는 것에 응답하여, 예외 핸들링 회로부(40)는 프로세싱 회로부(4)가 프로그램 카운터 레지스터(52) 및 PSTATE 레지스터(54)로부터의 현재 프로그램 카운터 값 및 현재 프로세서 상태를 각각 예외가 취해질 예외 상태(ELx)와 연관된 예외 링크 레지스터(62) 및 SPSR(60)에 복사하도록 트리거한다. 예외에 대한 이러한 예외 복귀 상태 복사는 하드웨어에서 트리거되므로 예외 복귀 상태 복사를 야기하는 임의의 특정 명령어가 실행될 필요가 없다. 그 후, 예외 핸들링 회로부(40)는 예외 핸들링 루틴을 실행하도록 프로세싱 회로부를 제어한다. 예를 들어, 예외 엔트리에 대한 응답으로 초기에 실행되는 예외 핸들링 루틴은 예외의 특정 원인을 다루기 위해 더 특정한 핸들러로 분기하기 전에 일부 제네릭 제어 함수들(generic control function)을 수행하는 제네릭 예외 래퍼 핸들러(generic exception wrapper handler)일 수 있다. 따라서, 특정 핸들러로 분기하기 전에, 새로운 예외 복귀 상태가 레지스터(60, 62)에 복사되게 하는 추가 예외가 발생하더라도 저장된 예외 복귀 상태가 보존되는 것을 보장하기 위해, 예외 핸들링 루틴은 레지스터 상태 정보(적어도 SPSR(60) 및 ELR(62)로부터의 예외 복귀 상태 정보를 포함함)를 스택과 같은, 메모리 내의 소프트웨어-유지 데이터 구조로 푸시하는 명령어(72)를 실행한다. 예외 복귀 상태를 보존하기 위해 사용되는 데이터 구조가 조직되는 특정 방식은 소프트웨어-특유 방식으로 선택될 수 있고, 이 예외 복귀 상태가 보존되는 방법에 대한 아키텍처 정의가 없을 수 있다. 예외 복귀 상태를 메모리에 저장한 경우, 예외 핸들링 소프트웨어 코드는 예외의 원인을 다루기 위해 특정 핸들러 코드로 분기되는 분기 명령어(74)를 호출하는 함수를 실행한다.

[0080] 예외로부터 복귀할 때, 예외 핸들링 소프트웨어 코드는 메모리 내의 데이터 구조로부터 이전에 저장된 레지스터 상태를 판독하는 명령어(76)(예를 들어, 스택 데이터 구조로부터 레지스터 상태를 팝함), 레지스터 상태를 예외 복귀 레지스터들(60, 62)에 복원하는 명령어(77)(예를 들어, 이것은 시스템 제어 레지스터들에 기록하도록 허용되는 시스템 레지스터 수정 명령어일 수 있음), 및 레지스터들(60, 62)에 저장된 예외 복귀 상태에 기초하여 예외 복귀를 트리거하는 예외 복귀 명령어(78)를 실행한다. 따라서, 예외 복귀 명령어(78)는 이전에 저장된 프로세서 상태가 SPSR_ELx(60)로부터 PSTATE 레지스터(54)로 복사되게 하고, 이전에 저장된 예외 복귀 어드레스가 ELR_ELx(62)로부터 프로그램 카운터 레지스터(52)로 복사되게 한다. 이렇게 하면 이전에 저장된 예외 복귀 어드레스에 의해 표시된 명령어로의 분기가 야기되고, 이는 SPSR_ELx에 이전에 저장된 프로세서 상태 값에 표시된 프로세서 상태로 실행될 것이다.

[0081] 공격자는 예외 복귀 상태 정보가 명령어(72)에 의해 기록되는 것과 명령어(76)에 의해 판독되는 것 사이에 메모리에 유지되는 동안 이 정보를 템퍼링함으로써 복귀 배향 프로그래밍(return oriented programming; ROP) 공격을 시작하려고 시도할 수 있다. 공격자가 예외 복귀 상태를 손상시키는 데 성공할 수 있다면, 이는 예외 복귀가 예외가 취해진 프로그램의 한 지점으로부터 상이한 지점으로 복귀하게 하거나, 또는 PSTATE 레지스터(54)에 의해 표현되는 프로세서 상태를 예외가 취해진 때와는 상이한 상태(예를 들어, 더 큰 권한을 갖는 상태)로 복귀하게 할 수 있으며, 이 둘 모두는 보안 정보 또는 소프트웨어 코드를 잠재적으로 손상시킬 위험을 초래할 수 있는 바람직하지 않은 거동을 초래할 수 있다.

[0082] 도 6은 예외 복귀 상태에 대한 이러한 ROP 공격으로부터 보호하기 위한 기법을 예시한다. 예외 엔트리 시에, 레지스터들(60, 62)에서 예외에 대해 캡처된 예외 복귀 상태를 보호형 제어 스택(GCS) 데이터 구조(80)라고 불리는 보호형 데이터 구조로 푸시하도록 추가적인 명령어가 실행된다. 예외 복귀 상태가 푸시되는 어드레스는 현재 예외 상태(ELx)와 연관된 GCS 스택 포인터 레지스터(64)에 기초하여 결정된다. (어드레스 변환을 제어하기 위해 MMU(28)에 의해 사용되는) 페이지 테이블에 의해 명시되는 액세스 허가는 GCS 데이터 구조(80)를 저장하기 위해 사용되는 어드레스 영역을, GCS 액세스 명령어들의 제한된 서브세트에 의해서만 액세스되도록 제한되고, 그 제한된 서브세트에 있지 않은 다른 타입들의 명령어를 사용하여서는 액세스될 수 없는 특별한 "GCS" 메모리 타입으로서 정의한다. 따라서, 실행되고 있는 프로그램의 범용 로드/저장 명령어를 위한 어드레스 피연산자를 템퍼링하는 공격자는 명령어가 특별한 GCS 액세스 명령어들 중 하나가 아닐 때 그 명령어가 GCS 데이터 구조(80)에 액세스하게 할 수 없다. 예를 들어, GCS 메모리 타입은 가상 어드레스 공간과 중간 어드레스 공간 사이의 매핑을 제공하는 데 사용되는 대응하는 스테이지-1 변환 테이블 엔트리에 의해 직접적으로 또는 간접적으로 명시된 스테이지-1 액세스 허가 내에서 표시될 수 있다(예를 들어, GCS 메모리 타입 허가는 스테이지-1 변환 테이블 엔트리 자체의 인코딩에서, 또는 스테이지-1 변환 테이블 엔트리에 의해 참조되는 레지스터에서 표시될

수 있음). 반대로, GCS 액세스 명령어가 GCS 메모리 타입으로 지정되지 않은 메모리의 영역에 액세스하려고 시도하면, 공격자가 GCS 데이터 구조(80) 이외의 영역에 액세스하도록 GCS 액세스 명령어를 속이는 것을 방지하기 위해 결함이 시그널링될 수 있다.

[0083] 예외 복귀 시, 도 5에 논의된 바와 같이 메모리 내의 소프트웨어 특유 데이터 구조로부터의 예외 복귀 상태(예외 복귀를 제어하기 위해 예외 복귀 상태 명령어(78)에 의해 후속적으로 사용되도록 의도됨)의 임의의 소프트웨어-제어식 복원과는 별개로, GCSPR_ELx(64) 내의 GCS 스택 포인터에 기초하여 결정되는 GCS 데이터 구조(80) 상의 위치로부터 보호된 예외 복귀 상태를 팝하기 위해 추가적인 GCS 팝 명령어가 실행되고, 보호된 예외 복귀 상태를 의도된 예외 복귀 상태와 비교하고, 미스매치가 검출되는 경우 결함을 시그널링한다. 이는 의도된 복귀 상태가 명령어(76)에 의해 팝되기 전에 메모리에서 템퍼링되었는지 여부를 검출하는 수단을 제공한다. 의도된 예외 복귀 상태가 GCS 데이터 구조(80)로부터 획득된 보호된 예외 복귀 상태와 일치하면, 예외 복귀 명령어(78)는 의도된 예외 복귀 상태에 기초하여 예외 복귀를 트리거하도록 진행할 수 있다.

[0084] 따라서, 도 7에 도시된 바와 같이, 예외 엔트리 및 복귀 코드는 예외 엔트리 시의 추가적인 GCS 예외 상태 푸시 명령어(81) 및 예외 복귀 시의 GCS 예외 상태 검증(팝 및 비교) 명령어(82)를 포함하도록 확장되어, 명령어(72)에 의한 메모리로의 예외 복귀 상태 푸시 및 명령어(76)에 의한 재판독(read back)을 위한 추가된 보호를 제공한다. 도 7은 GCS 데이터 구조(80)로부터 보호된 예외 복귀 상태를 팝하고 레지스터들(60, 62)에서 의도된 예외 복귀 상태를 보호된 예외 복귀 상태에 대해 검증하는 단일 명령어(82)를 도시하지만, 다른 예들은 이것을 2개의 별개의 명령어들로 분리할 수 있는데, 하나는 메모리로부터 보호된 예외 복귀 상태를 팝하기 위한 것이고, 또 다른 하나는 의도된 예외 복귀 상태가 팝된 예외 복귀 상태와 일치하는지 여부를 검증하기 위한 것이다. 또한, 일부 예들에서, GCS 예외 복귀 상태 팝 명령어(82)는 GCS 데이터 구조(80)로부터의 예외 복귀 상태만을 팝할 수 있지만, 예외 복귀 상태 레지스터들(60, 62)에서의 예외 복귀 상태에 대해 이를 검증하지는 않는다(예를 들어, GCS 예외 복귀 상태 팝 명령어(82)는 GCS-보호된 예외 복귀 상태를 예외 복귀 상태 레지스터들(60, 62)로 직접 팝할 수 있고, 이로부터 예외 복귀를 직접 제어하기 위해 예외 복귀 명령어(78)에 의해 GCS-보호된 예외 복귀 상태가 사용될 수 있음 - 이 경우 소프트웨어가 명령어들(72, 76)을 실행하여 예외 복귀 상태를 일반 비-GCS 메모리의 별도의 소프트웨어-관리 스택으로 푸시/팝할 필요가 없음).

[0085] 예외 복귀 상태를 GCS 구조(80)로 푸시하고 GCS 구조로부터 예외 복귀 상태를 팝하기 위해 제공되는 명령어들(81, 82)은 메모리의 GCS 영역인 것으로서 스테이지-1 액세스 허가에서 명시된 메모리 영역에 액세스하도록 허용된 GCS 액세스 명령어들로 간주된다.

[0086] GCS 이네이블 파라미터(68)가 GCS 모드가 이네이블됨을 나타내는 이네이블 상태로 설정될 때 위에서 논의된 다양한 GCS 관련 기능들이 수행될 수 있다는 것이 인식될 것이다. GCS 이네이블 파라미터(68)가 GCS 모드가 디스에이블되는 것을 표시하기 위해 디스에이블 상태로 설정될 때(예를 들어, 이것은 GCS 모드와 함께 사용하도록 설계되지 않은 레거시 코드와의 호환 가능성에 유용할 수 있음), GCS 액세스 명령어들은 아키텍처 기능을 갖지 않는 비동작 명령어들로서 취급될 수 있거나 또는 정의되지 않은 명령어들로서 취급될 수 있고(GCS 액세스 명령어들이 실행되는 경우 결함이 시그널링되게 함), GCS 모드가 이네이블될 때 특정 비-GCS 액세스 명령어들에 대해 수행될 수 있는 추가적인 검사가 생략될 수 있다(아래에서 논의되는 예시적인 흐름도들을 참조).

[0087] GCS 모드는 주어진 예외에 대해 캡처된 유효한 예외 복귀 상태의 사본을 유지하기 위해 사용되는 메모리 내의 보호된 영역을 지원하는 데 유용할 수 있으며, 이는 예외 복귀를 직접 제어하는 데 사용될 수 있거나 또는 예외 복귀를 제어하는 데 사용될 "의도된" 예외 복귀 상태를 검증하는 데 사용될 수 있다.

[0088] 그러나, GCS 예외 상태 푸시 명령어(80)가 예외 엔트리(70)에 응답하여 예외 복귀 상태의 하드웨어 트리거된 캡처과 별개이기 때문에, 명령어(81)에 응답하여 GCS 데이터 구조(80)로 푸시되는 예외 복귀 상태가 예외를 취하는 것에 응답하여 캡처된 예외 복귀 상태와 동일하지 않도록, 공격자가 예외 복귀 상태 레지스터들(60, 62)의 콘텐츠를 수정하려고 시도할 수도 있을 때의 시간 윈도우(90)(도 7에서 "윈도우 A"로 라벨링됨)가 존재한다. 유사하게, 예외 복귀에서, 예외 복귀 상태 정보가 레지스터들(60, 62)에서 수정될 수 있는 경우, GCS 예외 복귀 상태 팝 명령어(82)를 실행하는 것과 예외 복귀 명령어(78)를 실행하는 것 사이에 시간 윈도우(92)("윈도우 B"로 라벨링됨)가 존재하며, 이는 GCS(80)로부터 획득되지 않았거나 GCS(80)로부터 획득된 상태에 대해 검증되지 않은 프로세서 상태 및 프로그램 카운터 값으로 예외 복귀가 트리거되게 할 수 있다.

[0089] 이러한 문제를 해결하기 위해, 예외 복귀 상태 잠금 파라미터(EXLOCK)(58)가 PSTATE 레지스터(54)에 제공된다. EXLOCK(58)은 잠금 상태 및 잠금 해제 상태로 설정될 수 있다. 일례에서, 잠금 상태에서 EXLOCK=1이고 잠금 해제 상태에서 EXLOCK=0이다. 다른 예들은 반대 매핑(opposite mapping)을 사용할 수 있거나, 또는 EXLOCK(58)

은 예외 복귀 상태 잠금 파라미터 및 다른 정보 둘 모두를 인코딩하는 다중-비트 파라미터일 수 있고, 여기서 EXLOCK(58)의 상태들의 제1 서브세트는 잠금 상태로 간주될 수 있고, EXLOCK(58)의 상태들의 제2 서브세트는 잠금 해제 상태로 간주될 수 있다. 따라서, EXLOCK(58)이 잠금 상태 및 잠금 해제 상태를 나타내도록 인코딩될 수 있는 다양한 방법이 있다.

[0090] EXLOCK이 잠금 해제 상태에 있을 때, 예외 복귀 상태 레지스터들에 대한 수정이 허용될 수 있다(수정되도록 시도되는 예외 복귀 상태 레지스터들(60, 62)의 특정 뱅크된 버전을 업데이트하기에 충분한 권한을 갖는 현재의 예외 레벨(56)과 같이, 임의의 다른 검사가 충족되는 경우). EXLOCK(58)이 잠금 상태에 있을 때, EXLOCK(58)이 잠금 해제 상태에 있을 때 실행되는 경우 예외 복귀 상태 정보를 수정하도록 그렇지 않은 경우 허용될, 명령어를 사용하여 시도되더라도, 레지스터들(60, 62)에서 예외 복귀 상태에 저장된 정보에 대한 수정은 금지된다. 적어도 예외가 취해지기 전에 현재였던 동일한 예외 레벨에서 취해진 예외에 대해, EXLOCK 파라미터(58)는, 예외 엔트리 시에 캡처되는 것과 GCS 데이터 구조(80)로 푸시되는 것 사이, 및 GCS 데이터 구조(80)로부터 복원된 보호된 예외 복귀 상태에 대한 검증과 예외 복귀(78)를 위한 사용 사이에서 예외 복귀 상태 정보에 대한 템퍼링을 방지하기 위해, 도 7에 도시된 바와 같이, 윈도우 A(90) 및 윈도우 B(92) 동안의 수정에 대해 예외 복귀 상태 레지스터들(60, 62)을 잠그도록 설정될 수 있다. 현재 예외 레벨보다 더 많은 권한을 가진 예외 레벨에서 취해진 예외는 예외를 취할 시 잠금 파라미터를 설정될 필요가 없는데, 이는 예외가 취해진 후에 실행되는 더 많은 권한을 가진 코드를 신뢰하여 그 복귀 상태를 적절히 관리할 수 있기 때문이다.

[0091] 따라서, 예외 복귀 상태가 GCS 데이터 구조(80)에 저장되기 전에 그리고 GCS 데이터 구조(80)로부터 복원된 후에 이 상태가 수정되는 것을 방지하기 위한 아키텍처 메커니즘이 제공된다. 예외적으로, 이 상태는 잠겨 있고 수정될 수 없다. PSTATE.EXLOCK(58)은 (적어도 레지스터(60 또는 62)에 저장된 값을 변경하려고 시도하는 기록에 대해) ELR(62) 및 SPSR(60)에 대한 기록 액세스를 잠글 수 있고, 예외를 취할 시에 설정된다. GCS 버퍼(80)에 예외 복귀 상태를 저장하기 위한 푸시 절차(81)를 실행할 때, PSTATE.EXLOCK은 잠금 해제되어, 레지스터들(60, 62)에 대한 미래의 기록을 허용한다. GCS 버퍼(80)로부터 예외 복귀 상태를 복원하기 위한 팝 절차(82)를 실행할 때, PSTATE.EXLOCK은 재잠금되어, 미래의 기록을 방지하고 GCS로부터 복원된 상태가 불변임을 보장한다. 잠겨 있는 경우, 상태(ELR/SPSR)에 기록을 시도하면 결함이 시그널링된다(추가 예외가 취해지게 할 수 있음). 대안적으로, 이러한 결함은 레지스터들(60, 62) 중 하나 또는 그 둘 모두에 기록되는 값이 그 레지스터에서의 현재 값과 상이한 경우에만 발생할 수 있다. 이렇게 하면 복귀 상태가 변경되지 않는 시나리오들에 대해 고속 코드 경로를 허용할 수 있다.

[0092] EXLOCK 파라미터(85)의 추가적인 검사가 수행될 수 있어서, 다음 중 임의의 것이 발생하는 경우 결함이 트리거될 수 있다:

[0093] • EXLOCK 파라미터(85)가 현재 잠금 상태에 있을 때 예외 엔트리(70);

[0094] • EXLOCK 파라미터(85)가 현재 잠금 해제 상태에 있을 때 GCS 예외 상태 푸시 절차(81);

[0095] • EXLOCK 파라미터(85)가 현재 잠금 상태에 있을 때 GCS 예외 상태 팝 절차(82); 및

[0096] • EXLOCK 파라미터(85)가 현재 잠금 해제 상태에 있을 때 예외 복귀(78).

[0097] 이들 이벤트들 중 임의의 것은 공격자가 GCS 보호 명령어들 중 하나를 놓치도록 제어 흐름을 손상시켰거나 의도하는 상이한 순서로 코드가 실행되게 한다는 표시일 수 있으며, 이는 ROP 공격이 시도된다는 신호일 수 있다.

[0098] 예외-복귀-상태-잠금 이네이블 제어 파라미터, EXLOCKEN(69)은 이네이블 상태 또는 디스에이블 상태(EXLOCK(58)에 대해 위에서 논의된 것과 유사한 다양한 방식으로 다시 인코딩될 수 있음)로 설정될 수 있으며, 이네이블 상태는 예외 복귀 상태 잠금 파라미터(58)를 사용하여 위에서 논의된 예외 복귀 상태 잠금 특징부들이 이네이블됨을 나타내고, 디스에이블 상태는 이러한 예외 복귀 상태 잠금 특징부들이 디스에이블됨을 나타낸다. 아래에 도시된 바와 같이, GCS 모드가 디스에이블 상태로 설정된 GCS 이네이블 파라미터(68)를 사용하여 디스에이블되거나 또는 EXLOCKEN(69)이 디스에이블 상태로 설정된 경우, EXLOCK 파라미터(58)에 기초한 다양한 검사(및 EXLOCK 파라미터(58)를 잠금 상태 또는 잠금 해제 상태로 설정하기 위한 액션들)는 디스에이블될 수 있다.

[0099] 도 8은 GCS 모드가 현재 예외 레벨(ELx)에 대해 GCS 제어 레지스터(66)에 저장된 GCS 이네이블 파라미터(68)에 의해 이네이블될 때 발행되는 메모리 액세스에 대해 MMU(28)에 의해 수행되는 액세스 허가 검사를 도시하는 흐름도이다. 단계(100)에서, 타겟 메모리 영역에 대응하는 타겟 가상 어드레스를 명시하는 메모리 액세스 요청이

발행된다. 단계(102)에서, MMU(28)는 도 3에 도시된 바와 같이 스테이지-1 및 스테이지-2 어드레스 변환 구조들에 정의된 관련 스테이지-1(및 필요한 경우, 스테이지-2) 어드레스 매핑들에 기초하여 어드레스 변환을 개시한다. 임의의 알려진 어드레스 변환 기법이 변환을 위한 어드레스 매핑을 정의하는 데 사용될 수 있다. 단계(104)에서, MMU(28)는, 타겟 가상 어드레스에 대한 어드레스 매핑을 제공하는 타겟 스테이지-1 변환 테이블 엔트리에 의해 직접적으로 또는 간접적으로 명시된 스테이지-1 허가 정보가, 타겟 메모리 영역이 GCS 영역임을 명시하는지를 여부를 결정한다. 예를 들어, 스테이지-1 허가 정보는 타겟 스테이지-1 변환 테이블 엔트리 자체에 인코딩될 수 있거나, 또는 타겟 스테이지-1 변환 테이블 엔트리에 의해 참조되는 레지스터 필드에 저장될 수 있다. 다른 예들은 타겟 스테이지-1 변환 테이블 엔트리에 의해 타겟 VA에 매핑되는 중간 어드레스에 대응하는 스테이지-2 변환 테이블 엔트리에 의해 (레지스터에 대한 참조를 통해 직접적으로 또는 간접적으로) 명시된 스테이지-2 허가 정보를 사용하여 GCS 영역들을 정의할 수 있다. 또한, (변환 테이블 구조들과는 별도인) 별도의 데이터 구조가 어느 메모리 영역들이 GCS 영역들인지를 정의하는 것이 가능할 것이다.

[0100] 타겟 메모리 영역이 GCS 영역이면, 단계(106)에서, MMU(28)는 메모리 액세스 요청이 GCS 영역에 액세스하도록 허용된 명령어의 GCS-액세스 타입들의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청인지 여부를 결정한다. 그렇지 않다면, 단계(107)에서, MMU(28)는 메모리 액세스 요청이 기록 요청인지 여부를 결정한다. 메모리 액세스 요청이 기록 요청이면, 단계(108)에서, GCS 구조(80)에 액세스할 수 있는 다른 일반적인 로드/저장 명령어 타입들을 방지하기 위해 결함이 시그널링된다. 메모리 액세스 요청이 판독 요청이면, 결함을 시그널링할 필요가 없고, 방법은 단계(110)로 진행한다(다른 예들은 또한 비-GCS 판독 요청들에 대한 결함을 GCS 메모리 영역으로 시그널링하도록 선택할 수 있지만, 이는 판독 요청들이 GCS-보호된 예외 복귀 상태 정보를 손상시키지 않을 것이기 때문에 선택적임).

[0101] 메모리 액세스 요청이 GCS 메모리 액세스 요청이면, 단계(110)에서 MMU(28)는 메모리 액세스 요청에 대해 임의의 다른 액세스 허가가 만족되는지 여부를 결정한다. 예를 들어, 이것은 타겟 영역에 대응하는 스테이지-1 또는 스테이지-2 변환 테이블 엔트리들에서 명시된 다른 파라미터들, 또는 예를 들어, 프로세싱 회로부(4)의 보안 상태, 예외 레벨 또는 다른 동작 모드들에 기초한 다른 보안 검사에 기초할 수 있다. 임의의 다른 요구되는 액세스 허가가 만족되지 않으면, 단계(108)에서 결함이 다시 시그널링된다. 메모리 액세스 요청에 대해 충족될 필요가 있는 임의의 다른 액세스 허가가 모두 충족될 것으로 결정되면, 단계(112)에서 메모리 액세스 요청이 허용될 수 있다.

[0102] 단계(104)에서, MMU(28)가 타겟 메모리 영역이 GCS 영역이 아니라고 결정하면, 단계(114)에서 MMU(28)는 메모리 액세스 요청이 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청인지 여부를 결정한다. 이는 단계(106)에서와 동일한 검사이지만, 단계(106)와 비교하여 반대의 결과를 갖는다. 단계(114)에서, 메모리 액세스 요청이 GCS 메모리 액세스 요청이면, 단계(108)에서 결함이 시그널링된다. 이는 GCS 메모리 액세스 요청들이 메모리의 비-GCS 영역들에 액세스하는 데 사용될 수 없음을 보장하며, 이는 소프트웨어 개발자가 비-GCS 액세스에 대한 GCS-액세스 타입들의 명령어를 사용하려고 시도되는 것을 회피할 것이며, 이는 공격자에게 이용 가능한 공격 표면을 증가시키고, 공격자가 GCS 이외의 영역을 가리키도록 수정되는 GCS-액세스 타입의 명령어의 어드레스 피연산자의 검출을 허용한다. GCS-액세스 타입들의 명령어가 GCS 영역들에 액세스하는 것을 제한하고, 비-GCS-액세스 타입들의 명령어가 비-GCS 영역들에 액세스하는 것을 제한함으로써, 보안이 개선될 수 있다.

[0103] 단계(114)에서 메모리 액세스 요청이 GCS 메모리 액세스 요청이 아닌 것으로 결정되면, 단계(116)에서 MMU(28)는 메모리 액세스 요청에 대해 임의의 다른 액세스 허가가 충족되는지 여부를 결정한다. 예를 들어, 이는 위에서 논의된 바와 같이 보안 상태, 예외 레벨 또는 동작 모드에 기초한 검사를 포함할 수 있다. 모든 요구되는 액세스 조건들이 충족되면, 단계(118)에서 메모리 액세스 요청이 허용된다. 임의의 액세스 허가가 위반되면, 단계(108)에서 결함이 시그널링된다.

[0104] 도 8은 다수의 상이한 검사가 실패하는 것에 응답하여 결함을 시그널링하는 단일 단계(108)를 도시하지만, 일부 구현들은 결함의 원인에 따라 상이한 결함 타입들을 시그널링할 수 있다는 것이 인식될 것이다.

[0105] 도 8은 GCS 모드가 현재 예외 레벨(ELx)에 대한 GCS 제어 레지스터(66)의 GCS 모드 이네이블 파라미터(68)에 의해 이네이블되는 동안 메모리 액세스 요청이 수신되는지에 대해 수행되는 검사를 도시한다. GCS 모드가 현재 디스에이블되면, 단계들(104, 106, 107, 114)은 생략될 수 있고, 수행되는 검사는 단계들(110, 116)에서 설명된 바와 같이 GCS 데이터 구조(80)의 사용과 관련 없는 다른 액세스 허가에 기초한 검사일 수 있다.

[0106] 도 8에 도시된 바와 같이, GCS 데이터 구조(80)에 대한 보호는 MMU(28)에 의해 아키텍처 레벨에서 제공되며,

MMU(28)는 (예를 들어, 변환 테이블을 사용하여) 전용 GCS 메모리 영역 타입의 정의를 지원하는데, 이는 그 영역 내의 해당 어드레스들이, 다른 로드/저장 명령어들에 의해서가 아닌 GCS-액세스 타입들의 명령어들의 제한된 서브세트에 의해 액세스되는 것이 요구될 때, 여분의 보호를 받는다는 것을 나타낸다. GCS-액세스 타입들의 명령어들은 적어도 도 7에서 위에서 논의된 GCS 예외 복귀 상태 푸시 명령어(81) 및 GCS 예외 복귀 상태 팝/검증 명령어(82)를 포함할 수 있다. 의도된 예외 복귀 상태에 대해 또한 검증하지 않고 GCS 데이터 구조(80)로부터 예외 복귀 상태를 팝하는 독립형 GCS 예외 복귀 상태 팝 명령어와 같은 다른 타입들의 GCS-액세스 타입들의 명령어들이 또한 제공될 수 있다. 예를 들어, 함수 복귀에 사용되는 복귀 어드레스를 보호하기 위해 함수 복귀 상태를 푸시 및 복원하기 위한 다른 GCS-액세스 타입들의 명령어들이 제공될 수 있다.

[0107] 도 9는 예외 엔트리에 응답하여 수행되는 단계들을 예시하는 흐름도이다. 단계(120)에서, 예외 핸들링 회로부(40)는 예외 신호(42)에 기초하여, 예외가 취해질 것임을 검출하고 예외 엔트리 절차를 트리거한다. 예외 엔트리에 응답하여, 단계(122)에서, 프로그램 카운터 및 현재 프로세서 상태는 예외가 취해질 예외 레벨(ELx)에 각각 대응하는 예외 링크 레지스터(62) 및 저장된 프로세서 상태 레지스터(60)의 버전으로 레지스터들(52, 54)로부터 복사된다. 단계(124)에서, 프로세싱 회로부(4) 또는 예외 핸들링 회로부(40)는 예외 엔트리 프로그램 코드로부터 실행될 제1 명령어의 어드레스를 나타내기 위해 프로그램 카운터 레지스터(52)를 업데이트하고, 예외가 취해질 프로세서 상태를 나타내기 위해 PSTATE 레지스터(54)를 업데이트하여, PSTATE 레지스터(54)에 의해 표시되는 새로운 상태에서 실행될 예외 엔트리 코드로의 분기를 야기한다. 단계(126)에서, 프로세싱 회로부(4)는 (GCS 이네이블 파라미터(68)에 기초하여) GCS 모드가 현재 이네이블되는지 여부를 결정하고, (GCS 예외-복귀-상태 잠금 이네이블 파라미터 EXLOCKEN(69)에 기초하여) 예외 복귀 상태의 잠금이 이네이블되는지 여부를 결정한다(사용할 파라미터들(68, 69)의 버전은 현재 예외 레벨 ELx에 대응하는 GCS 제어 레지스터(66)의 뱅크된 버전 중 하나에 저장된 버전임). 또한, 프로세싱 회로부(4)는, 예외가 취해지기 전의 현재의 예외 레벨과 동일한 예외 레벨(ELx)에서 이 예외가 취해져야 하는지를 결정한다. GCS 모드가 이네이블되고 예외 복귀 상태의 잠금이 이네이블될 때, 예외가 취해지기 전의 현재 예외 레벨과 동일한 예외 레벨에서 예외가 취해져야 할 때, 단계(128)에서 프로세싱 회로부(4)는 예외 복귀 상태 잠금 파라미터(EXLOCK)(58)를 잠금 상태로 설정한다. 그렇지 않으면, EXLOCK(58)은 단계(129)에서 잠금 해제 상태로 설정된다. 도 9는 EXLOCK(58)이 동일한 예외 레벨(EL)에서 취해진 예외에 대해서만 잠금 상태로 설정되는 예를 도시하지만, 다른 예들은 또한 더 많은 권한을 가진 예외 레벨로 취해진 예외에 대해 EXLOCK(58)을 잠금 상태로 설정될 수 있다.

[0108] 도 10은 보호된 예외 복귀 상태를 GCS 데이터 구조(80)로 푸시하기 위한 예외 복귀 상태 푸시 절차를 도시하는 흐름도이다. 단계(130)에서, 명령어 디코더(10)는 GCS 이네이블 파라미터(68)가 GCS 모드가 이네이블됨을 표시하는 이네이블 상태에 있는 동안 GCS 예외 복귀 상태 푸시 명령어를 디코딩한다(GCS 모드가 디스에이블된 경우, GCS 예외 복귀 상태 푸시 명령어는 비동작 명령어 또는 정의되지 않은 명령어로서 해석될 수 있고, 도 10의 잔여 단계들은 수행되지 않음). 단계(132)에서, 프로세싱 회로부(4)는 명령어 디코더(10)에 의해 제어되어, EXLOCKEN(69)이 예외 복귀 상태의 잠금이 이네이블됨을 나타내는지 여부를 검사한다(GCS 제어 레지스터(66)의 뱅크된 버전이 상이한 예외 레벨들에 대해 제공되는 경우, 단계(132)에서의 검사를 위해 사용할 파라미터(69)의 버전은 현재 예외 레벨에 대응하는 버전임). 그렇다면, 단계(134)에서, 프로세싱 회로부(4)는 예외 복귀 상태 잠금 파라미터(EXLOCK(58))가 현재 잠금 해제 상태에 있는지 여부를 검사하도록 제어된다. EXLOCK(58)이 잠금 해제 상태에 있으면, 단계(136)에서 결함이 시그널링되는데, 그 이유는 예외 복귀 상태 잠금이 이네이블될 때의 유효한 예외 엔트리 시퀀스에 대해, 도 7에 도시된 바와 같이, 예외 복귀 상태가 GCS 데이터 구조(80)로 푸시되고 있을 때 예외 복귀 상태 잠금 파라미터(58)가 현재 잠겨야 하기 때문이다(EXLOCK(58)이 도 9에 설명된 바와 같이 예외를 취할 때 잠기지 않은 더 많은 권한을 가진 예외 레벨로 취해진 예외의 경우, 예외 복귀 상태를 관리하기 위해 더 많은 권한을 가진 예외 핸들러가 신뢰될 수 있으므로 GCS 예외 복귀 상태 푸시 명령어를 실행할 필요가 전혀 없으므로, GCS 예외 복귀 상태 푸시 명령어를 실행할 때 EXLOCK(58)이 잠기도록 요구하는 것이 그러한 더 많은 권한을 가진 예외 핸들러에 문제를 일으키지 않음). EXLOCK(58)이 현재 잠금 해제 상태에 있는 경우 결함을 시그널링하는 것은 예외 엔트리에 후속하는 것 이외에 GCS 푸시 절차가 실행되게 하기 위해 프로그램 흐름의 템퍼링에 기초하여 공격에 대한 대응책을 제공한다. 따라서, 단계(134)에서의 검사는 공격자에 의해 제어되는 임의의 예외 복귀 상태(예외 엔트리에 응답하여 캡처된 예외 복귀 상태가 아님)가 GCS 데이터 구조(80)로 푸시되는 위험을 감소시킨다.

[0109] 단계(132)에서 예외 복귀 상태 잠금이 디스에이블되는 것으로 발견되거나(EXLOCKEN(69)이 디스에이블 상태로 설정됨) 또는 단계(134)에서 EXLOCK(58)이 잠금 상태에 있으면, 단계(138)에서 프로세싱 회로부(4)는 예외 복귀 상태 레지스터들(60, 62)로부터 예외 복귀 상태 정보를 획득하고, 단계(140)에서 프로세싱 회로부(4)는 획득된 예외 복귀 상태 정보를 현재 예외 레벨(ELx)에 대응하는 GCS 포인터 레지스터(GCSPR_ELx)(64)에 저장된 현재 스

택 포인터에 기초하여 결정된 GCS 데이터 구조(80) 상의 위치로 푸시하도록 하나 이상의 메모리 액세스 요청들을 발행한다.

[0110] 단계(140)에서, 프로세싱 회로부(4)는 EXLOCKEN(69)이 예외 복귀 상태의 잠금이 이네이블됨을 나타내는지 여부를 검사하고, 예외 복귀 상태의 잠금이 이네이블되면 단계(144)로 진행한다(또는 대안적으로, 단계(132)에서 수행된 동일한 검사는 또한 예외 복귀 상태의 잠금이 이네이블되는지 또는 디스에이블되는지에 따라 단계(144)가 조건부로 수행되게 할 수 있으므로, EXLOCKEN 파라미터(69)를 다시 검사할 필요가 없을 수 있음). 예외 복귀 상태의 잠금이 이네이블되면, 단계(144)에서, 예외 복귀 상태 잠금 파라미터(58)는 잠금 해제 상태로 설정된다(이 단계는 예외 복귀 상태의 잠금이 디스에이블되면 생략됨). 예외 복귀 상태의 잠금이 이네이블되는지 또는 디스에이블되는지에 관계없이, 단계(146)에서, 프로세싱 회로부(4)는 현재의 실행 상태(예외 레벨)(ELx)에 대응하는 GCS 포인터 레지스터들(64) 내의 스택 포인터를 업데이트하여 스택으로의 예외 복귀 상태의 푸시를 고려한다(예를 들어, 스택 포인터는 레지스터들(60, 62)의 콘텐츠에 기초하여 형성된 예외 복귀 상태 레코드의 크기에 대응하는 양만큼 감소되거나 증가될 수 있음).

[0111] 도 11은 GCS 데이터 구조(80)로부터 예외 복귀 상태를 팝하기 위해 수행되는 단계들을 도시하는 흐름도이다. 단계(150)에서, 명령어 디코더(10)는 GCS 이네이블 파라미터(68)에 의해 이네이블된 것으로 표시된 바와 같은 GCS 모드 동안 GCS 예외 복귀 상태 팝 명령어를 디코딩한다(다시 말하면, GCS 모드가 디스에이블되면, GCS 예외 복귀 상태 팝 명령어는 NOP(no-operation instruction) 또는 정의되지 않은 명령어로서 취급될 수 있고, 도 11의 잔여 단계들은 수행되지 않을 수 있음).

[0112] GCS 예외 복귀 상태 팝 명령어에 응답하여, 단계(152)에서, 프로세싱 회로부(4)는 EXLOCKEN(69)이 예외 복귀 상태의 잠금이 이네이블됨을 표시하는 이네이블 상태로 설정되는지를 검사한다(GCS 제어 레지스터(66)의 뱅크된 버전이 상이한 예외 레벨들에 대해 제공되는 경우, 단계(152)에서의 검사를 위해 사용할 파라미터(69)의 버전은 현재 예외 레벨에 대응하는 버전임). 예외 복귀 상태의 잠금이 이네이블되면, 단계(154)에서 프로세싱 회로부(4)는 EXLOCK 파라미터(58)가 잠금 상태로 설정되는지 여부를 검사한다. EXLOCK(58)이 현재 잠금 상태에 있으면, 단계(156)에서 결함이 시그널링되는데, 이는 EXLOCK이 잠길 때 예외 복귀 상태 팝 명령어가 실행되지 않아야 하는 예상되는 용례(usage)로부터의 이탈을 나타내기 때문이다. 이 결함은 GCS 데이터 구조(80)로부터, 보호된 예외 복귀 상태의 부정확한 검증을 야기하기 위해 프로그램 흐름을 손상시키려는 가능한 공격을 시그널링할 수 있다.

[0113] 단계(152)에서 예외 복귀 상태의 잠금이 디스에이블되는 것으로 결정되었거나 또는 단계(154)에서 EXLOCK(58)이 현재 잠금 해제 상태에 있으면, 단계(158)에서 프로세싱 회로부(4)는 현재 예외 레벨(ELx)에 대응하는 GCS 포인터 레지스터(66) 내의 스택 포인터에 기초하여 결정되는 GCS 데이터 구조(80) 상의 위치로부터 GCS-보호된 복귀 상태 정보를 팝하기 위해 하나 이상의 메모리 액세스 요청들을 발행한다. 단계(160)에서, 레지스터 GCSPR_ELx(66) 내의 GCS 스택 포인터는 GCS 데이터 구조(80)로부터의 예외 복귀 상태의 팝을 고려하도록 업데이트된다(예를 들어, 스택 포인터는 도 10의 단계(146)에서 이루어진 업데이트에 대해 반대 방향으로 (그러나 대응하는 양만큼) 증가되거나 감소됨).

[0114] 단계들(162, 164 및 166)은, GCS 예외 복귀 상태 팝 명령어가 또한 프로세싱 회로부(4)를 제어하여 현재 예외 복귀 레지스터들(60, 62)에 저장된 의도된 예외 복귀 상태 정보가 단계(158)에서 GCS 데이터 구조(80)로부터 팝된 GCS-보호된 예외 복귀 상태 정보와 일치하는지 여부를 검증하는 구현에서, GCS 예외 복귀 상태 팝 명령어에 응답하여 선택적으로 수행될 수 있는 단계들이다. 단계(162)에서, 프로세싱 회로부(4)는 현재 예외 레벨(ELx)에 대응하는 예외 링크 레지스터(62) 및 SPSR(60) 내의 현재 값들에 의해 표현되는 의도된 예외 복귀 상태 정보를 판독한다. "의도된" 예외 복귀 상태 정보라는 용어는 GCS 예외 복귀 상태 팝 명령어를 실행할 때의 레지스터들(62, 60)에서의 예외 복귀 상태의 현재 값들(이는 대응하는 예외 엔트리가 발생했을 때 그들 레지스터들에 기록된 값들과 동일할 수도 있고 동일하지 않을 수도 있음)을 지칭하기 위해 사용된다. 단계(164)에서, 프로세싱 회로부(4)는 레지스터들(60, 62)로부터 획득된 의도된 예외 복귀 상태 정보가 메모리의 보호된 GCS 영역에 저장된 GCS 데이터 구조(80)로부터 판독된 GCS-보호된 예외 복귀 상태 정보와 일치하는지 여부를 결정하고, 미스매치가 존재하면 단계(166)에서 결함이 시그널링된다.

[0115] 의도된 예외 복귀 상태 정보가 단계(164)에서 GCS-보호된 예외 복귀 상태 정보와 일치하거나, 또는 GCS 예외 복귀 상태 팝 명령어가 검증을 트리거하지 않기 때문에 단계들(162, 164, 166)이 구현되지 않으면(그러나, 대신에, 예를 들어, GCS-보호된 복귀 상태 정보를 예외 복귀 상태 레지스터들(60, 62)에 직접 팝함), 단계(168)에서, 프로세싱 회로부(4)는 EXLOCKEN(69)이 예외 복귀 상태의 잠금이 이네이블됨을 나타내는지 여부를 검사

하고, 예외 복귀 상태의 잠금이 이네이블되면 단계(170)로 진행한다(또는 대안적으로, 단계(152)에서 수행되는 동일한 검사는 또한, 단계(170)가 예외 복귀 상태의 잠금이 이네이블되는지 또는 디스에이블되는지에 따라 조건부로 수행되게 할 수 있음). 예외 복귀 상태의 잠금이 이네이블되면, 단계(170)에서, 예외 복귀 상태 잠금 파라미터(58)는 잠금 상태로 설정된다(예외 복귀 상태의 잠금이 디스에이블되면 이 단계는 생략됨).

[0116] 도 12는 예외 복귀 명령어에 응답하여 수행되는 단계들을 예시하는 흐름도이다. 단계(180)에서, 명령어 디코더(10)는 예외 복귀 명령어를 디코딩하고, 이에 응답하여 후속 단계들을 수행하도록 프로세싱 회로부(4)(특히, 실행 스테이지(16))를 제어하기 위한 제어 신호들을 생성한다. 단계(182)에서, 프로세싱 회로부(4)는 GCS 이네이블 파라미터(68)가 GCS 모드가 현재 이네이블되어 있음을 표시하는지 여부, EXLOCKEN 파라미터(69)가 예외 복귀 상태의 잠금이 또한 이네이블되어 있음을 표시하는지 여부(다시 말하면, GCS 제어 레지스터(66)의 뱅크된 버전이 상이한 예외 레벨들에 대해 제공되는 경우, 단계(182)에서의 검사를 위해 사용할 파라미터들(68, 69)의 버전은 현재 예외 레벨에 대응하는 버전임), 및 예외 복귀가 현재 예외 레벨과 동일한 예외 레벨에서의 프로세싱으로의 복귀인지 여부를 결정한다. GCS 모드 및 예외 복귀 상태의 잠금 둘 모두가 이네이블되고 예외 복귀가 현재 예외 레벨에서 수행되는 프로세싱으로의 복귀를 야기하는 경우, 단계(184)에서 프로세싱 회로부(4)는 EXLOCK 파라미터(58)가 현재 잠금 해제 상태에 있는지 여부를 검사한다. EXLOCK(58)이 현재 잠금 해제 상태에 있으면, 단계(186)에서 결함이 시그널링되는데, 이는 예외 복귀가 이전의 예외 복귀 상태 팝 명령어(82)를 따르지 않았을 수 있고, 따라서 레지스터들(60, 62)에 명시된 잠재적으로 안전하지 않은 예외 복귀 상태를 사용하여 예외 복귀를 수행하는 것이 안전하지 않을 수 있음을 나타내기 때문이다.

[0117] GCS 모드가 현재 디스에이블되거나, EXLOCKEN 파라미터(69)가 예외 복귀 상태의 잠금을 디스에이블하도록 설정되거나, 예외 복귀가 현재 예외 레벨보다 더 적은 권한을 가진 예외 레벨로 진행되거나, EXLOCK 파라미터(58)가 현재 잠금 상태에 있으면, 예외 복귀가 진행될 수 있고, 단계(188)에서 예외 복귀 어드레스는 ELR_ELx(62)로부터 프로그램 카운터 레지스터(52)로 복사되고, 저장된 프로세서 상태는 SPSR_ELx(60)로부터 PSTATE 레지스터(54)로 복사된다. EXLOCK 파라미터(58)가 예외시에 저장되고 예외 복귀시에 복원되는 프로세서 상태의 일부이기 때문에, 단계(188)는 예외 복귀 상태 잠금 파라미터(58)를 대응하는 예외를 취할 때 EXLOCK(58)에 의해 표시되는 동일한 상태로 변경할 수 있다는 점에 유의한다. 따라서, EXLOCK(58)이 예외 복귀 이후에 잠금 상태에 있는지 또는 잠금 해제 상태에 있는지 여부는 대응하는 예외가 취해진 EXLOCK의 상태에 좌우될 것이다.

[0118] 도 13은 레지스터들(60, 62)에서 예외 복귀 상태 정보의 수정이 허용될 수 있는지 여부를 제어하기 위한 단계들을 도시하는 흐름도이다. 단계(200)에서, 명령어 디코더(10)는 타겟 예외 레벨과 연관된 예외 링크 레지스터(62) 또는 저장된 프로세서 상태 레지스터(60) 중 하나 또는 그 둘 모두에서 예외 복귀 상태 정보(예를 들어, 예외 복귀 어드레스 또는 저장된 프로세서 상태)의 수정을 요청하는 명령어를 디코딩한다(더 많은 권한을 가진 예외 레벨에서 실행되는 명령어들이 더 적은 권한을 가진 예외 레벨과 연관된 예외 복귀 상태를 수정하는 것이 가능하기 때문에, 타겟 예외 레벨은 현재 예외 레벨(56)일 수도 있고 아닐 수도 있음). 단계(202)에서, 프로세싱 회로부(4)는 이네이블 파라미터(68)에 의해 이네이블된 것으로 표시된 바와 같은 GCS 모드 및 EXLOCKEN 파라미터(69)가 예외 복귀 상태의 잠금이 이네이블됨을 나타내는지 여부를 결정한다. GCS 모드가 디스에이블되거나 예외 복귀 상태의 잠금이 디스에이블되면, 단계(204)에서, 임의의 다른 검사(예컨대, 현재 예외 레벨이 명시된 레지스터의 수정을 허용할 충분한 권한을 갖는지 여부를 검사하는 것)가 충족되면, 예외 복귀 상태 정보에 대한 수정이 허용될 수 있다. 단계(202)에서 GCS 모드 및 예외 복귀 상태의 잠금 모두가 이네이블된다고 결정되면, 단계(206)에서 프로세싱 회로부(4)는 예외 복귀 상태 잠금 파라미터(EXLOCK)(58)가 현재 잠금 상태에 있는지 여부를 결정한다. EXLOCK(58)이 잠금 해제 상태에 있으면, 단계(204)에서 다시, 임의의 다른 검사가 충족되면 수정이 허용될 수 있다. 단계(206)에서 EXLOCK 파라미터(58)가 잠금 상태에 있으면, 단계(208)에서 예외 복귀 상태 정보의 수정을 방지하기 위해 결함이 시그널링된다(그리고 그러므로 도 7에 도시된 바와 같이 윈도우 A(90) 또는 윈도우 B(92)에서 예외 복귀 상태로 템퍼링하려는 시도들에 대해 감시함).

[0119] 도 13의 예에서, 단계(208)에서의 결함의 시그널링은 레지스터들(60, 62) 중 하나의 예외 복귀 상태 정보에 기록될 값이 그 레지스터에 저장된 이전 값과 동일하거나 상이한지 여부와 무관하다. 따라서, 단계(200)에서 직면된 명령어의 실행이 실제로 예외 복귀 상태의 변화를 야기하지 않았을지라도, 결함이 시그널링될 것이다. 이 접근법은 결함 시그널링이 레지스터들(60, 62)에 대한 판독 및 기록될 새로운 값에 대한 비교에 좌우되도록 하기 위해 회로 논리를 제공할 필요가 없기 때문에 하드웨어에서 구현하는 것이 더 간단할 수 있다.

[0120] 대안이 도 14에 도시되어 있다. 도 14의 단계들(200, 202, 204, 206 및 208)은 도 13의 대응하는 단계들과 동일하다. 도 14는 레지스터들(60, 62) 중 하나에 기록될 새로운 값이 그 레지스터에서의 예외 복귀 상태 정보의 오래된 값과 상이한지 여부를 검사하는 단계(210)에서의 추가적인 검사를 포함한다. 새로운 값이 오래된 값과

다르면, 결함은 단계(208)에서 시그널링된다. 새로운 값이 오래된 값과 동일하면, 단계(204)에서, 시도된 "수정"은 그것이 예외 복귀 상태 정보를 변경하지 않을 것이므로 허용될 수 있다. 이 접근법은 (예외를 취한 지 거의 직후에 예외로부터의 복귀하기 위해) 예를 들어, 매우 짧은 예외 핸들링 루틴을 사용하는 고속 예외 경로들과 같은 일부 코드 경로들의 속도를 높일 수 있으므로 예외 복귀 상태를 메모리에 저장/복원할 필요가 없으므로 공격자에 의한 템퍼링의 위험을 감소시킨다. 이러한 고속 예외 코드 경로들에 대해, GCS 예외 복귀 상태 푸시/팝 명령어들은 사용되지 않을 수 있고, 따라서 EXLOCK은 예외 루틴 전체에 걸쳐 설정된 채로 유지될 수 있다. 그럼에도 불구하고, 예외 핸들링 루틴이 레지스터들(60, 62)에서 예외 복귀 상태를 수정하려고 시도하는 시스템 레지스터 수정 명령어를 실행하는 경우(가능성이 낮지만, 가능하기는 함), 시스템 레지스터 수정 명령어에 의해 기록되는 새로운 값이 실제로 오래된 값과 동일하여 예외 복귀 상태를 손상시킬 위험이 없는 경우에, EXLOCK이 잠겨 있는 동안 트리거될 대응하는 결함이 회피될 수 있다. 이렇게 하면 이러한 코드 경로들의 성능을 향상시키는 데 도움이 될 수 있다.

[0121] 위에서 논의된 흐름도들은 특정 순차적 순서로 수행되는 단계들을 도시하지만, 동일한 기능들이 또한 상이한 순서로 또는 적어도 부분적으로 병렬로 수행될 수 있다는 것이 인식될 것이다.

[0122] 도 15는 사용될 수 있는 시뮬레이터 구현을 예시한다. 앞서 설명된 실시예들은 관심이 있는 기법들을 지원하는 특정 프로세싱 하드웨어를 동작시키기 위한 장치 및 방법들과 관련하여 본 발명을 구현하지만, 본 명세서에서 설명되는 실시예들에 따라 컴퓨터 프로그램의 사용을 통해 구현되는 명령어 실행 환경을 제공하는 것이 또한 가능하다. 그러한 컴퓨터 프로그램들은, 그들이 하드웨어 아키텍처의 소프트웨어 기반 구현을 제공하는 한, 종종 시뮬레이터들로 지칭된다. 다양한 시뮬레이터 컴퓨터 프로그램들은 에뮬레이터들, 가상 기계들, 모델들, 및 동적 이진 변환기(dynamic binary translator)들을 포함한 이진 변환기들을 포함한다.전형적으로, 시뮬레이터 구현은 호스트 프로세서(330) 상에서 실행되어, 호스트 운영 체제(320)를 선택적으로 실행하여, 시뮬레이터 프로그램(310)을 지원할 수 있다. 일부 배열들에서, 하드웨어와 제공된 명령어 실행 환경 사이에 다수의 시뮬레이션 계층들이 있을 수 있고/있거나, 동일한 호스트 프로세서 상에 제공된 다수의 별개의 명령어 실행 환경들이 있을 수 있다.역사적으로, 강력한 프로세서들이 합리적인 속도로 실행되는 시뮬레이터 구현들을 제공하기 위해 요구되었지만, 그러한 접근법은 호환 가능성 또는 재사용 이유들을 위해 또 다른 프로세서에 고유한 코드를 실행하려는 요구가 있을 때와 같은 소정 상황들에서 정당화될 수 있다.예를 들어, 시뮬레이터 구현은 호스트 프로세서 하드웨어에 의해 지원되지 않는 추가적인 기능성을 명령어 실행 환경에 제공할 수 있거나, 또는 상이한 하드웨어 아키텍처와 전형적으로 연관된 명령어 실행 환경을 제공할 수 있다. 시뮬레이션의 개요가 문헌["Some Efficient Architecture Simulation Techniques", Robert Bedichek, Winter 1990 USENIX Conference, Pages 53 - 63]에서 주어진다.

[0123] 실시예들이 특정 하드웨어 구성들 또는 특징들을 참조하여 앞서 설명된 범위 내에서, 시뮬레이팅된 실시예에서, 동등한 기능성이 적합한 소프트웨어 구성들 또는 특징들에 의해 제공될 수 있다. 예를 들어, 특정 회로부는 시뮬레이팅된 실시예에서 컴퓨터 프로그램 논리로서 구현될 수 있다. 마찬가지로, 메모리 하드웨어, 예컨대 레지스터 또는 캐시는 시뮬레이팅된 실시예에서 소프트웨어 데이터 구조로서 구현될 수도 있다. 전술된 실시예들에서 언급된 하드웨어 요소들 중 하나 이상의 하드웨어 요소들이 호스트 하드웨어(예를 들어, 호스트 프로세서(330)) 상에 존재하는 배열들에서, 일부 시뮬레이팅된 실시예들은, 적합한 경우, 호스트 하드웨어를 사용할 수 있다.

[0124] 시뮬레이터 프로그램(310)은 컴퓨터 판독 가능 저장 매체(이는 비일시적 매체일 수 있음)에 저장될 수 있고, 타겟 코드(300)(이는 애플리케이션들, 운영 체제들 및 하이퍼바이저를 포함할 수 있음)에 프로그램 인터페이스(명령어 실행 환경)를 제공하는데, 이는 시뮬레이터 프로그램(310)에 의해 모델링되는 하드웨어 아키텍처의 인터페이스와 동일하다.따라서, 앞서 설명된 GCS 액세스 명령어들을 포함하는, 타겟 코드(300)의 프로그램 명령어들은 시뮬레이터 프로그램(310)을 사용하여 명령어 실행 환경 내로부터 실행될 수 있어서, 위에서 논의된 장치(2)의 하드웨어 특징들을 실제로 갖지 않는 호스트 컴퓨터(330)가 이들 특징들을 에뮬레이팅할 수 있게 할 수 있다. 유사하게, EXLOCK 파라미터(58)에 기초한 검사는 시뮬레이터 프로그램(310)을 사용하여 (예를 들어, 프로세싱 프로그램 논리(312) 또는 명령어 디코딩 프로그램 논리(314)의 일부를 사용하여) 에뮬레이팅될 수 있다.

[0125] 그러므로, 시뮬레이터 프로그램(310)은 위에서 설명되는 프로세싱 회로부(4)의 상태를 시뮬레이팅하는 프로세싱 프로그램 논리(312)를 구비할 수도 있다. 예를 들어, 프로세싱 프로그램 논리(312)는 타겟 코드(300)의 시뮬레이팅된 실행 동안 발생하는 이벤트들에 응답하여 실행 상태(EL0-EL3)의 이행을 제어할 수도 있다. 명령어 디코딩 프로그램 논리(314)는 타겟 코드(300)의 명령어들을 디코딩하고 이들을 호스트 장치(330)의 네이티브 명령어 세트의 명령어들의 대응하는 세트들에 매핑한다. 레지스터 에뮬레이팅 프로그램 논리(316)는, 예컨대 호스트

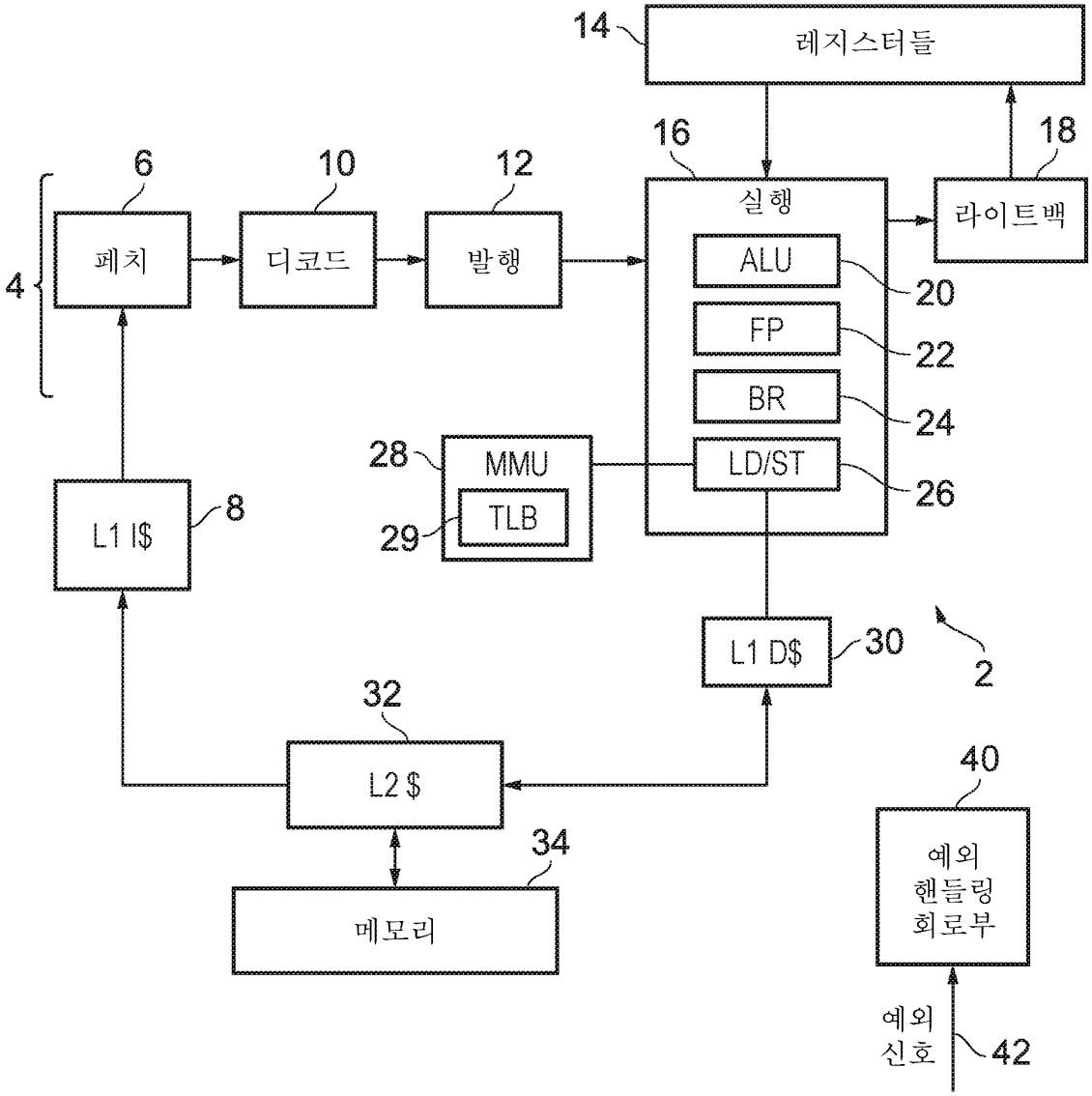
장치(330)의 레지스터들 또는 메모리(332)의 데이터에 액세스하는 것에 의해, 타겟 코드에 의해 요청되는 레지스터 액세스를 호스트 장치(330)의 호스트 하드웨어 상에서 유지되는 대응하는 데이터 구조들에 매핑한다. 메모리 관리 프로그램 논리(318)는 상기의 하드웨어 구현 실시예에서 설명되는 MMU(28)에 대응하는 방식으로 어드레스 변환, 페이지 테이블 워크(page table walk) 및 액세스 제어 검사를 구현하지만, 그러나, 타겟 코드(300)에 대해 정의되는 페이지 테이블에 기초하여 어드레스 변환에서 스테이지 2 매핑에 의해 획득되는 시뮬레이팅된 물리적 어드레스들을, 호스트 메모리(332)에 액세스하기 위해 사용되는 호스트 가상 어드레스들로 매핑하는 추가적인 기능을 또한 갖는다. 이러한 호스트 가상 어드레스들은 그들 자체가 호스트에 의해 지원되는 표준 어드레스 변환 메커니즘들을 사용하여 호스트 물리적 어드레스들로 변환될 수 있다(호스트 물리적 어드레스들로의 호스트 가상 어드레스들의 변환은 시뮬레이터 프로그램(310)에 의해 제어되는 것의 범위 밖에 있음). 예외 핸들링 프로그램 논리(320)는 타겟 코드(300)의 실행 동안 발생하는 시뮬레이팅된 예외(전술한 EXLOCK-기반 검사로 인해 트리거되는 결함들을 포함함)의 핸들링을 제어한다.

[0126] 본 출원에서, 단어들 "... 하도록 구성된"은 장치의 요소가 정의된 동작을 수행할 수 있는 구성을 갖는다는 것을 의미하는 데 사용된다. 이러한 맥락에서, "구성"은 하드웨어 또는 소프트웨어의 상호접속의 배열 또는 방식을 의미한다. 예를 들어, 장치는 정의된 동작을 제공하는 전용 하드웨어를 가질 수 있거나, 프로세서 또는 다른 프로세싱 디바이스가 기능을 수행하도록 프로그래밍될 수 있다. "하도록 구성된"은, 장치 요소가, 정의된 동작을 제공하기 위해 어떤 방식으로든 변경될 필요가 있음을 암시하지는 않는다.

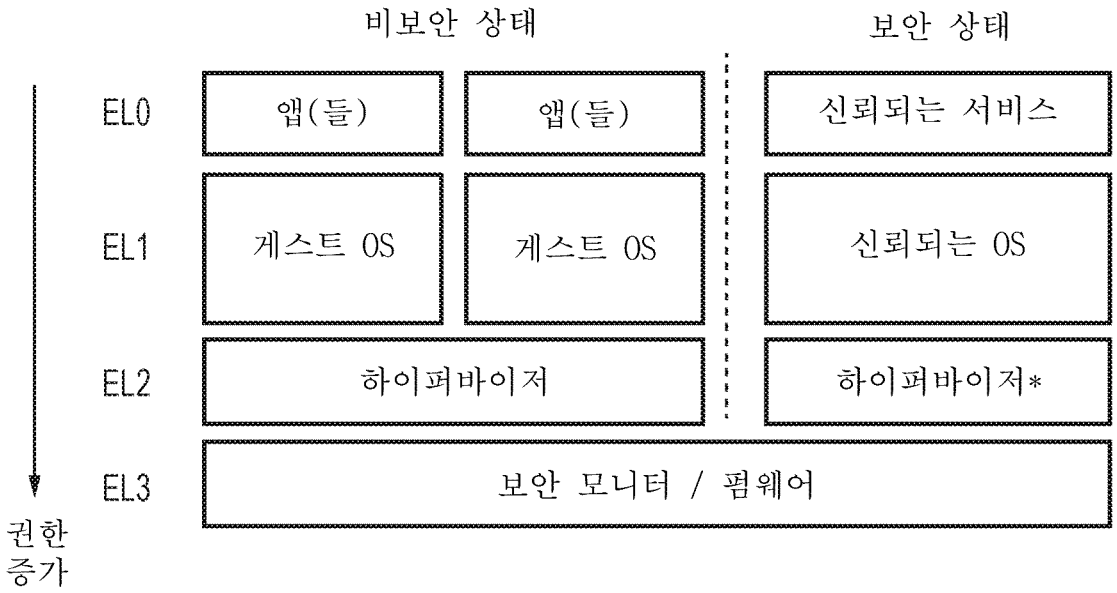
[0127] 본 발명의 예시적인 실시예들이 첨부 도면들을 참조하여 본 명세서에서 상세히 설명되었지만, 본 발명은 그러한 정확한 실시예들로 제한되지 않으며, 첨부된 청구항들에 의해 한정된 바와 같은 본 발명의 범위로부터 벗어남이 없이 실시예들에서 다양한 변경들 및 수정들이 당업자에 의해 이루어질 수 있다는 것이 이해되어야 한다.

도면

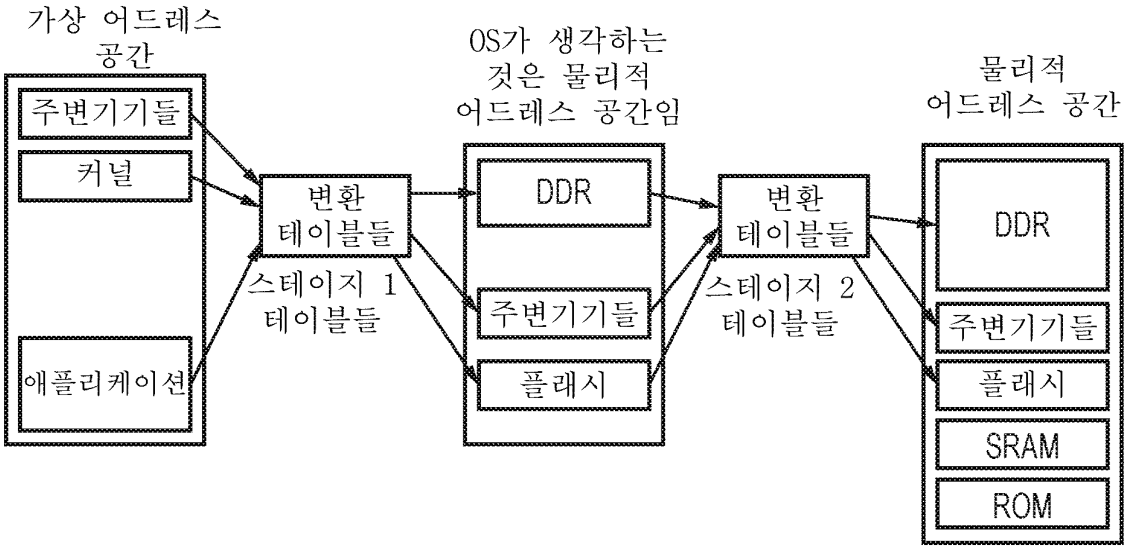
도면1


14
레지스터들
16
18
6
10
12
실행
라이트백
4
페치
디코드
발행
ALU
20
FP
22
BR
24
28
MMU
LD/ST
26
29
TLB
L1 I$
8
L1 D$
30
2
32
L2 $
40
예외
핸들링
회로부
34
메모리
예외
신호
42

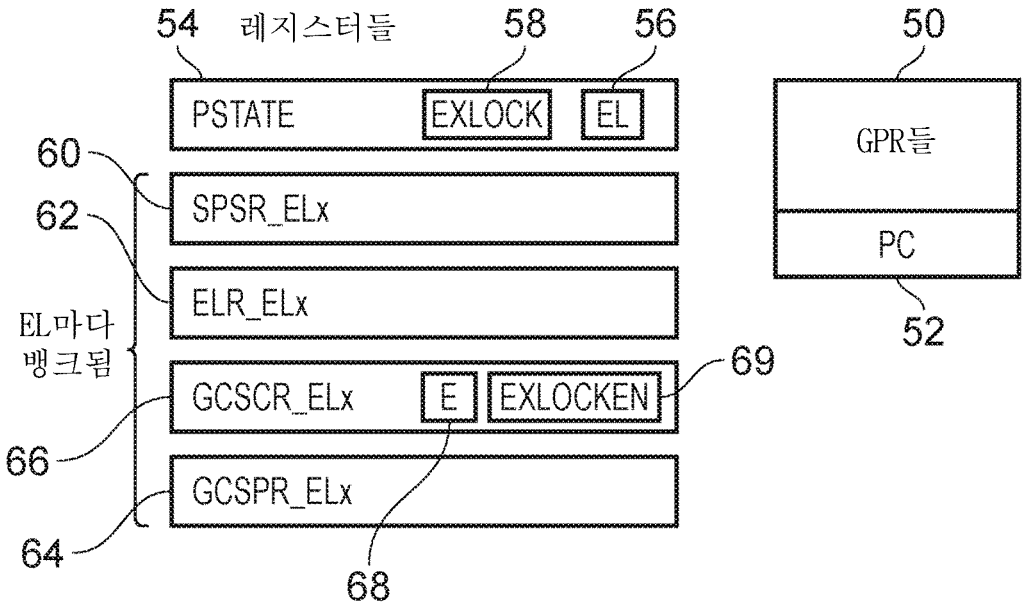
도면2


비보안 상태
보안 상태
EL0
앱(들)
앱(들)
신뢰되는 서비스
EL1
게스트 OS
게스트 OS
신뢰되는 OS
EL2
하이퍼바이저
하이퍼바이저*
EL3
보안 모니터 / 펌웨어
권한
증가

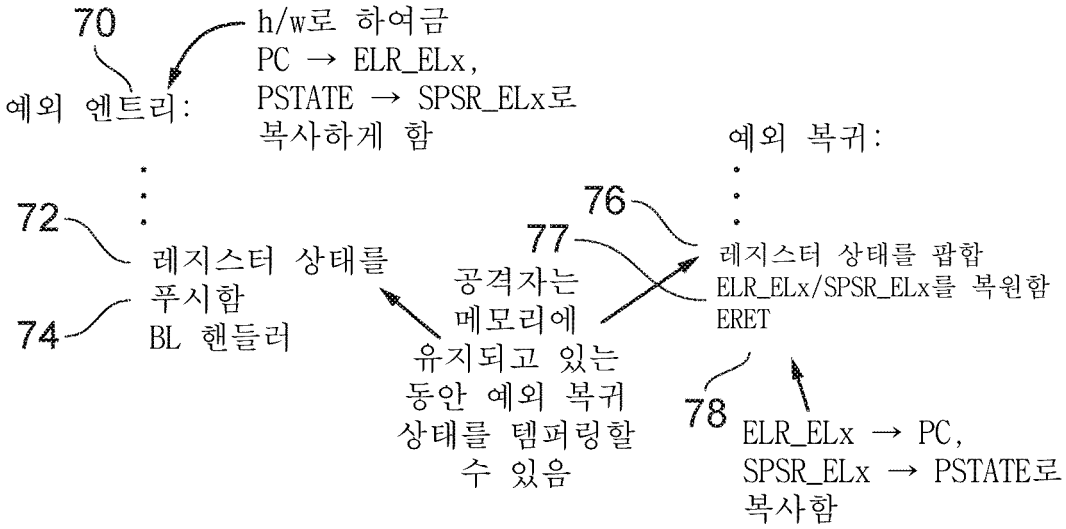
도면3


가상 어드레스
공간
주변기기들
커널
애플리케이션
변환
테이블들
스테이지 1
테이블들
OS가 생각하는
것은 물리적
어드레스 공간임
DDR
주변기기들
플래시
변환
테이블들
스테이지 2
테이블들
물리적
어드레스 공간
DDR
주변기기들
플래시
SRAM
ROM

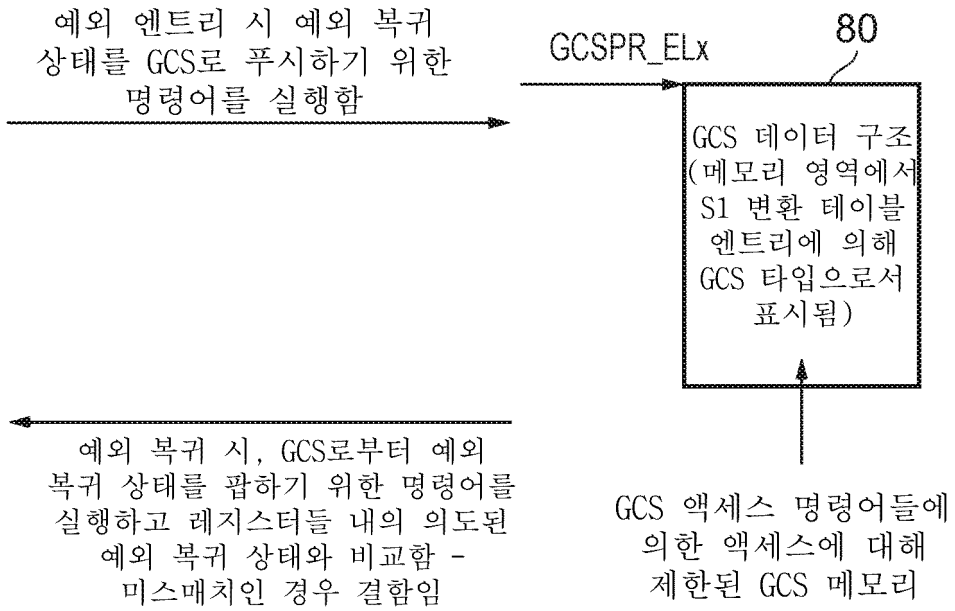
도면4


54
레지스터들
58
56
50
PSTATE
EXLOCK
EL
GPR들
60
SPSR_ELx
62
PC
ELR_ELx
52
EL마다
뱅크됨
69
GCSCR_ELx
E
EXLOCKEN
66
GCSPR_ELx
64
68

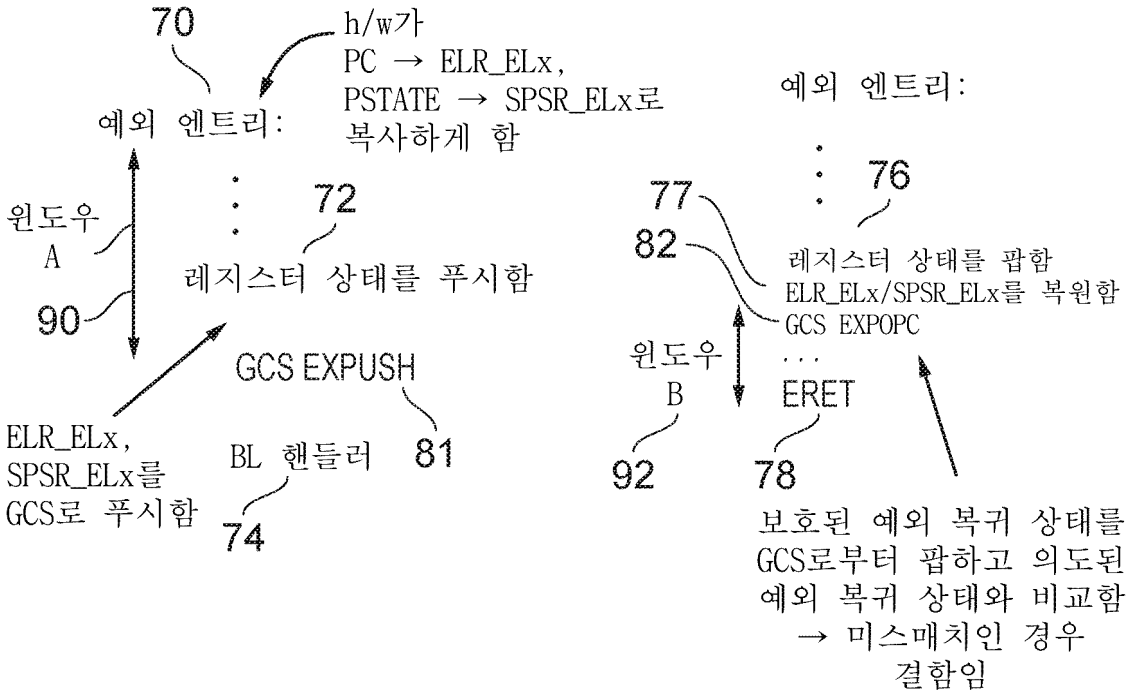
도면5


70
h/w로 하여금
PC → ELR_ELx,
PSTATE → SPSR_ELx로
복사하게 함
예외 엔트리:
예외 복귀:
72
76
77
레지스터 상태를
푸시함
74
BL 핸들러
공격자는
메모리에
유지되고 있는
동안 예외 복귀
상태를 템퍼링할
수 있음
레지스터 상태를 팝함
ELR_ELx/SPSR_ELx를 복원함
ERET
78
ELR_ELx → PC,
SPSR_ELx → PSTATE로
복사함

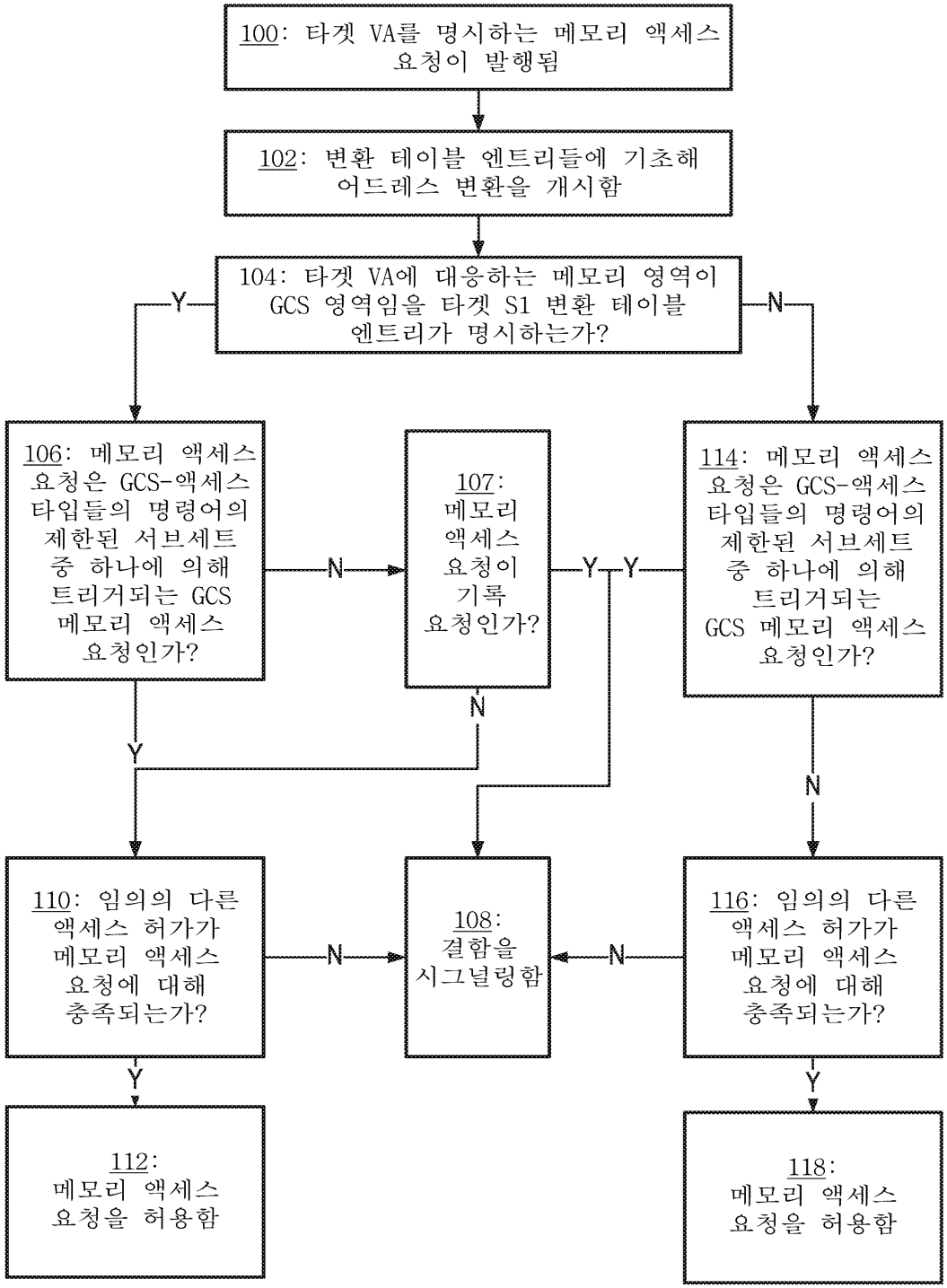
도면6


예외 엔트리 시 예외 복귀 상태를 GCS로 푸시하기 위한 명령어를 실행함
GCSPR_ELx
80
GCS 데이터 구조 (메모리 영역에서 S1 변환 테이블 엔트리에 의해 GCS 타입으로서 표시됨)
예외 복귀 시, GCS로부터 예외 복귀 상태를 팝하기 위한 명령어를 실행하고 레지스터들 내의 의도된 예외 복귀 상태와 비교함 - 미스매치인 경우 결함임
GCS 액세스 명령어들에 의한 액세스에 대해 제한된 GCS 메모리

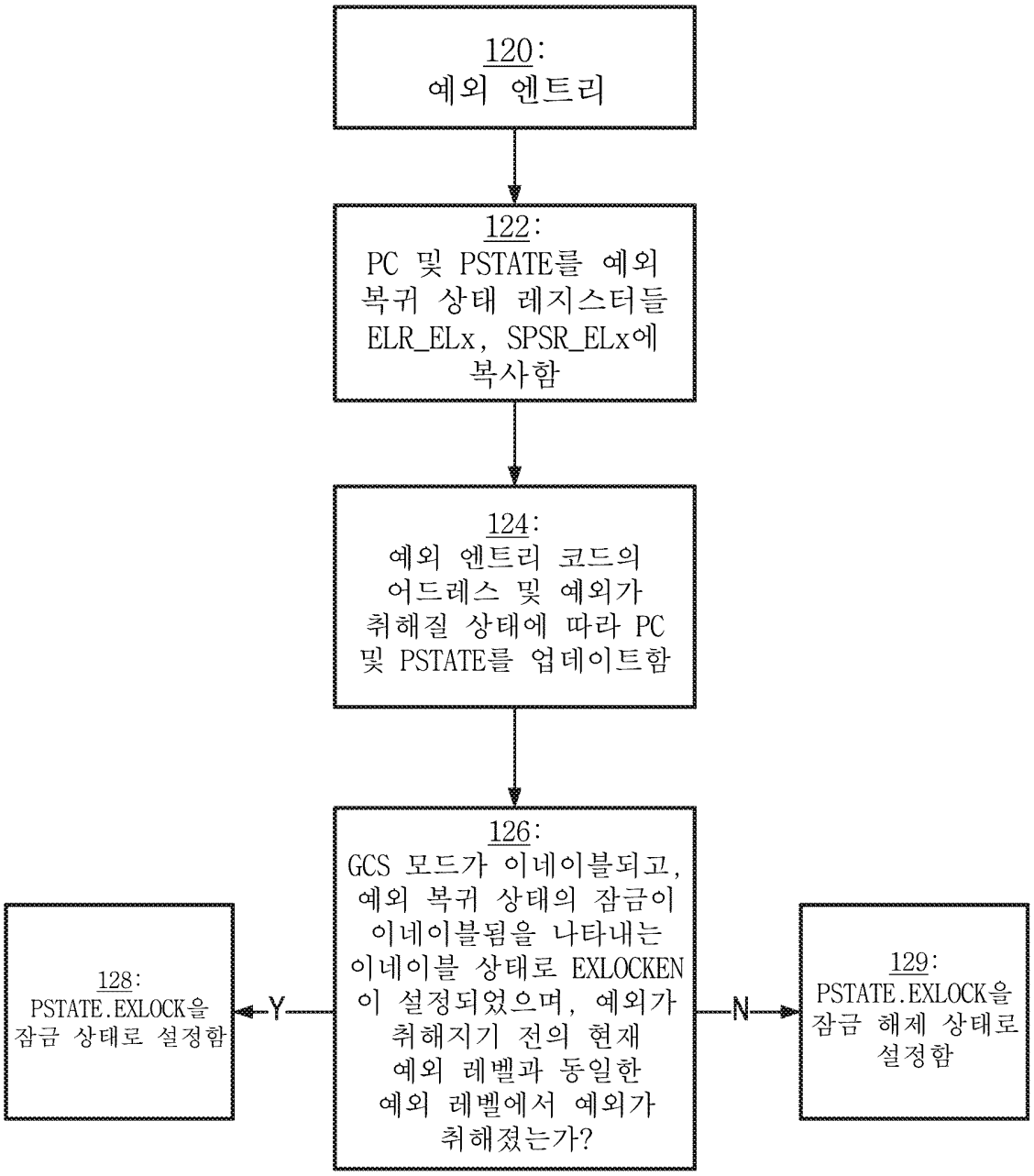
도면7


70
h/w가 PC → ELR_ELx, PSTATE → SPSR_ELx로 복사하게 함
예외 엔트리:
72
윈도우 A
90
레지스터 상태를 푸시함
GCS EXPUSH
81
ELR_ELx, SPSR_ELx를 GCS로 푸시함
BL 핸들러
74
예외 엔트리:
76
77
82
레지스터 상태를 팝함
ELR_ELx/SPSR_ELx를 복원함
GCS EXPOPC
윈도우 B
ERET
92
78
보호된 예외 복귀 상태를 GCS로부터 팝하고 의도된 예외 복귀 상태와 비교함 → 미스매치인 경우 결함임

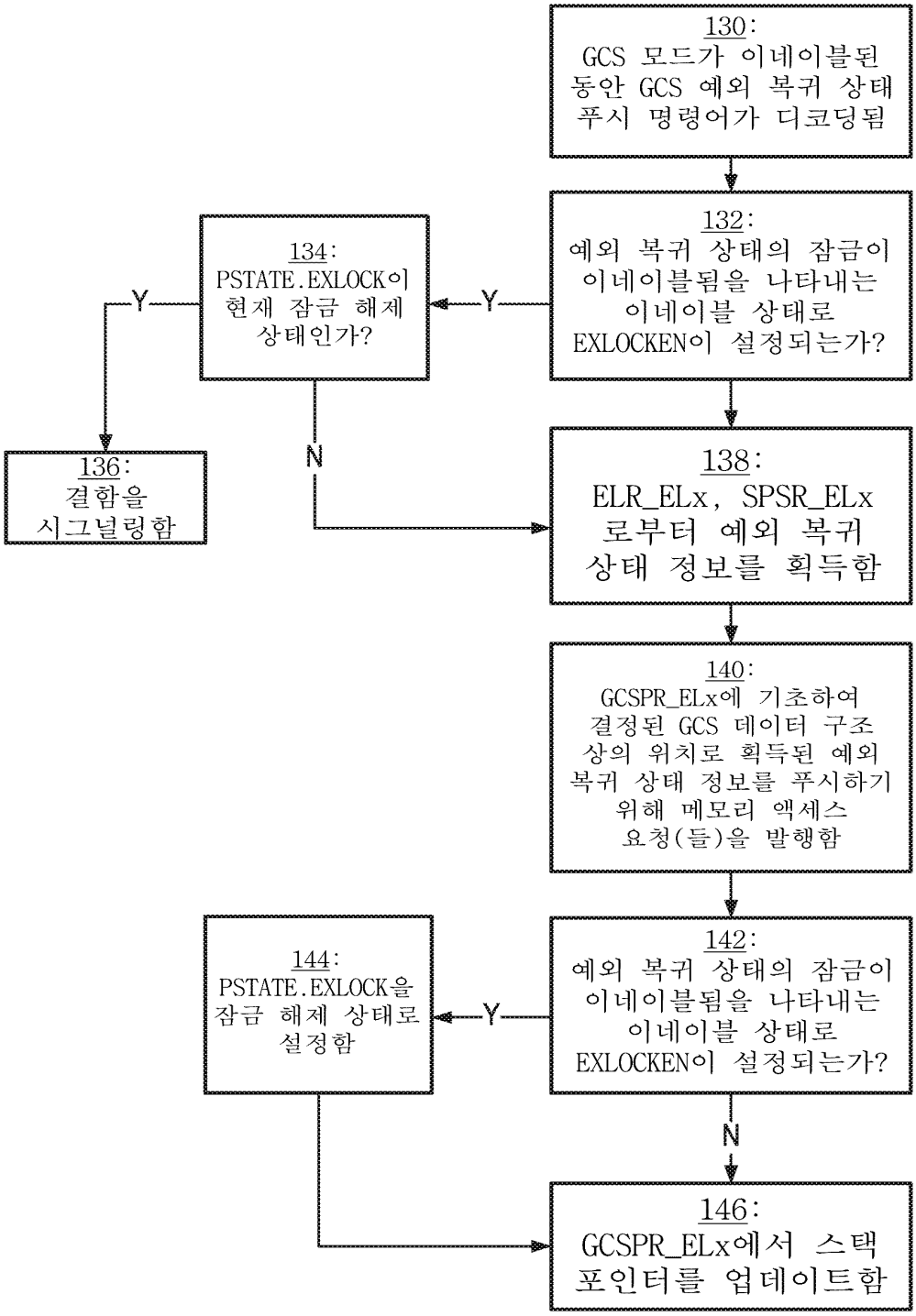
도면8


100: 타겟 VA를 명시하는 메모리 액세스 요청이 발행됨
102: 변환 테이블 엔트리들에 기초해 어드레스 변환을 개시함
104: 타겟 VA에 대응하는 메모리 영역이 GCS 영역임을 타겟 S1 변환 테이블 엔트리가 명시하는가?
Y
N
106: 메모리 액세스 요청은 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청인가?
107: 메모리 액세스 요청이 기록 요청인가?
114: 메모리 액세스 요청은 GCS-액세스 타입들의 명령어의 제한된 서브세트 중 하나에 의해 트리거되는 GCS 메모리 액세스 요청인가?
110: 임의의 다른 액세스 허가가 메모리 액세스 요청에 대해 충족되는가?
108: 결함을 시그널링함
116: 임의의 다른 액세스 허가가 메모리 액세스 요청에 대해 충족되는가?
112: 메모리 액세스 요청을 허용함
118: 메모리 액세스 요청을 허용함

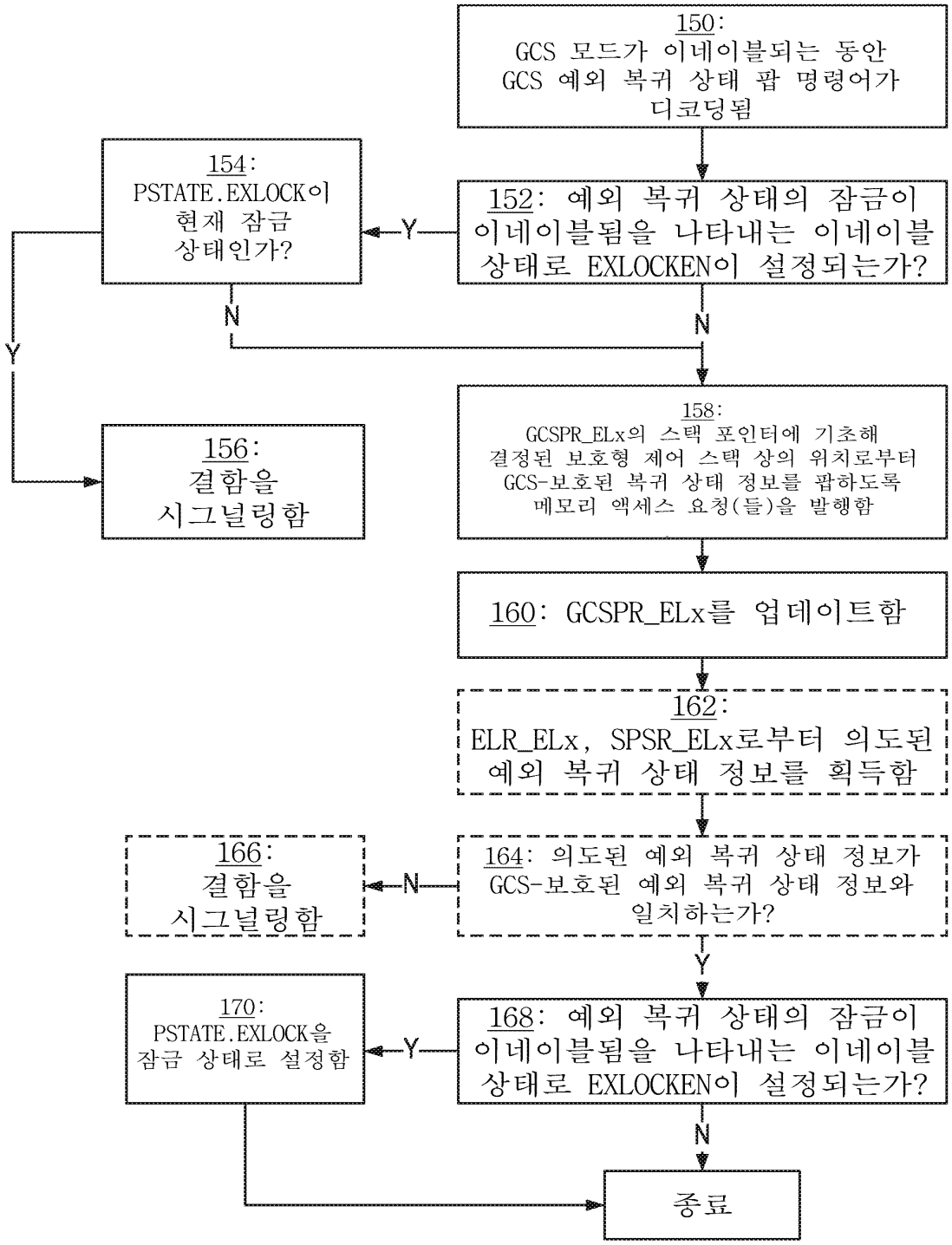
도면9


120:
예외 엔트리
122:
PC 및 PSTATE를 예외 복귀 상태 레지스터들 ELR_ELx, SPSR_ELx에 복사함
124:
예외 엔트리 코드의 어드레스 및 예외가 취해질 상태에 따라 PC 및 PSTATE를 업데이트함
126:
GCS 모드가 이네이블되고, 예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되었으며, 예외가 취해지기 전의 현재 예외 레벨과 동일한 예외 레벨에서 예외가 취해졌는가?
Y
N
128:
PSTATE.EXLOCK을 잠금 상태로 설정함
129:
PSTATE.EXLOCK을 잠금 해제 상태로 설정함

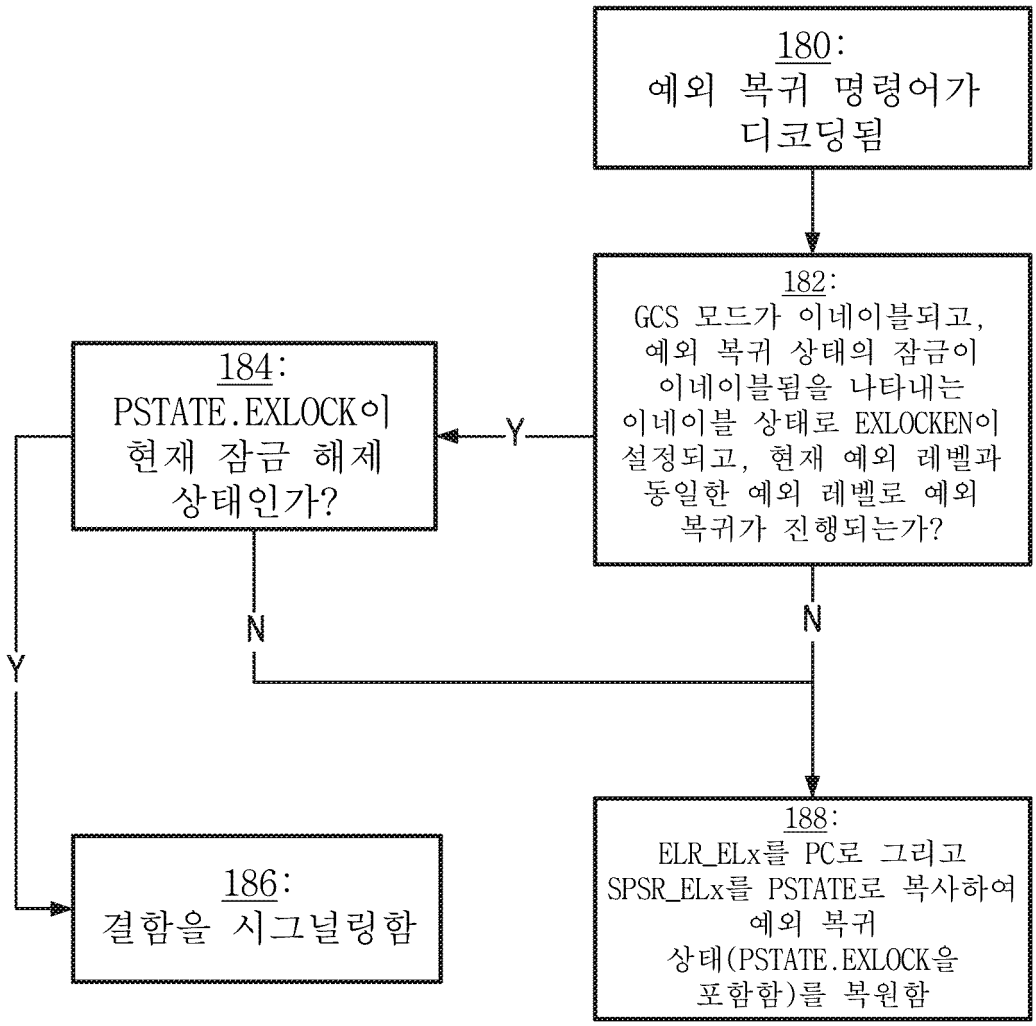
도면10


130:
GCS 모드가 이네이블된 동안 GCS 예외 복귀 상태 푸시 명령어가 디코딩됨
132:
예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
Y
134:
PSTATE.EXLOCK이 현재 잠금 해제 상태인가?
Y
N
136:
결함을 시그널링함
138:
ELR_ELx, SPSR_ELx로부터 예외 복귀 상태 정보를 획득함
140:
GCSPR_ELx에 기초하여 결정된 GCS 데이터 구조 상의 위치로 획득된 예외 복귀 상태 정보를 푸시하기 위해 메모리 액세스 요청(들)을 발행함
142:
예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
Y
144:
PSTATE.EXLOCK을 잠금 해제 상태로 설정함
N
146:
GCSPR_ELx에서 스택 포인터를 업데이트함

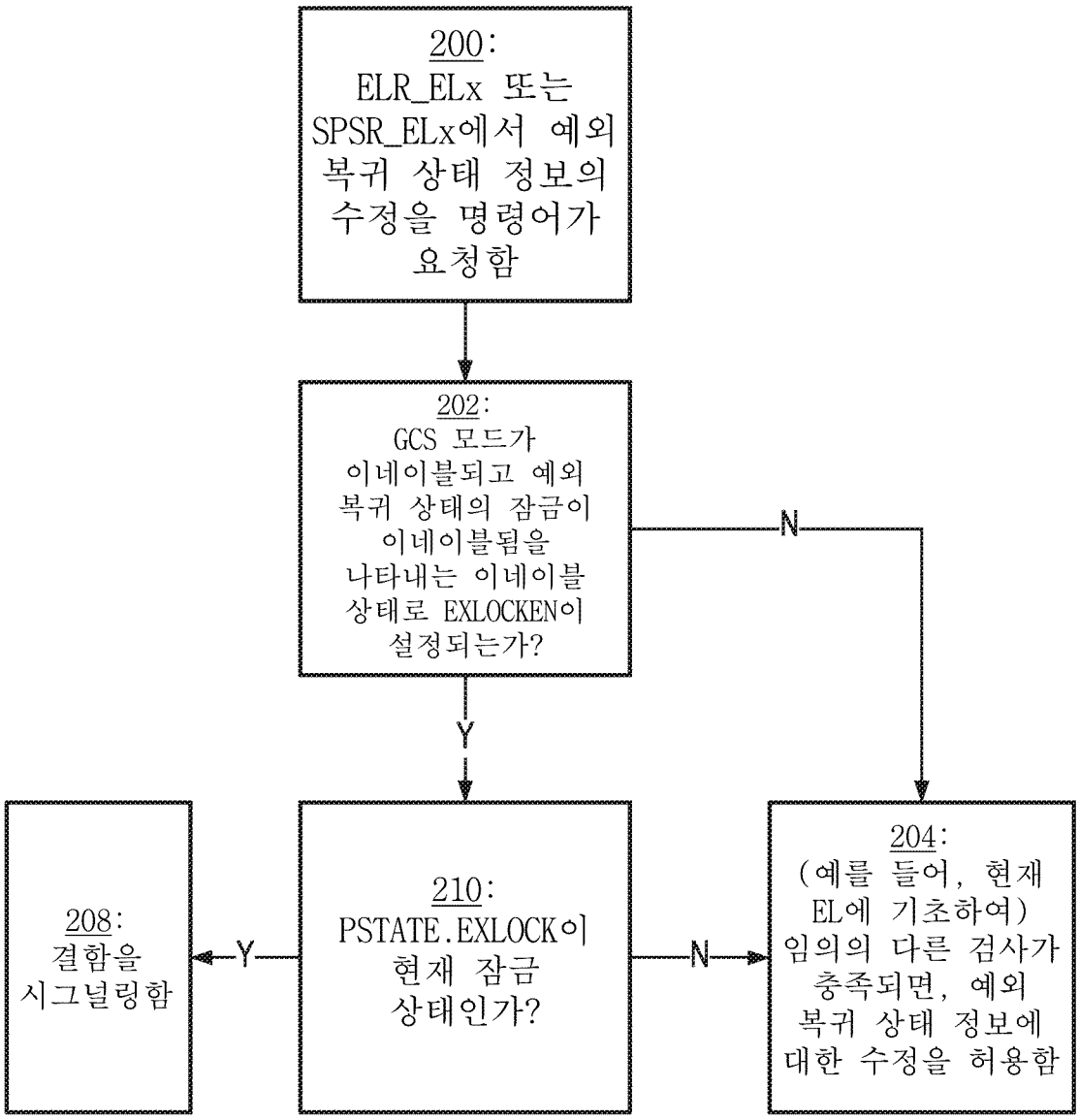
도면11


150: GCS 모드가 이네이블되는 동안 GCS 예외 복귀 상태 팝 명령어가 디코딩됨
152: 예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
Y
154: PSTATE.EXLOCK이 현재 잠금 상태인가?
N
Y
156: 결함을 시그널링함
N
158: GCSPR_ELx의 스택 포인터에 기초해 결정된 보호형 제어 스택 상의 위치로부터 GCS-보호된 복귀 상태 정보를 팝하도록 메모리 액세스 요청(들)을 발행함
160: GCSPR_ELx를 업데이트함
162: ELR_ELx, SPSR_ELx로부터 의도된 예외 복귀 상태 정보를 획득함
164: 의도된 예외 복귀 상태 정보가 GCS-보호된 예외 복귀 상태 정보와 일치하는가?
N
166: 결함을 시그널링함
Y
168: 예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
Y
170: PSTATE.EXLOCK을 잠금 상태로 설정함
N
종료

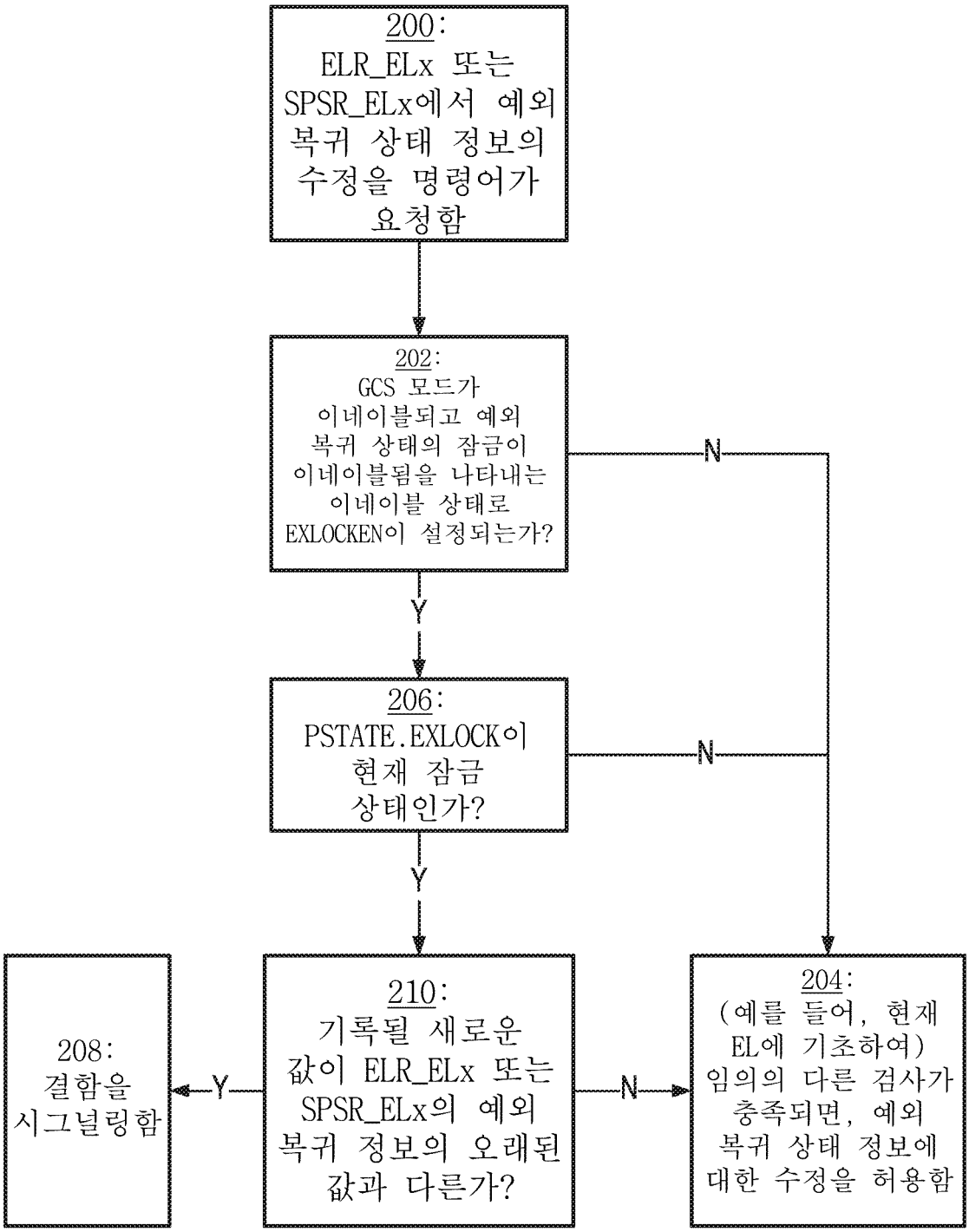
도면12


180:
예외 복귀 명령어가
디코딩됨
182:
GCS 모드가 이네이블되고,
예외 복귀 상태의 잠금이
이네이블됨을 나타내는
이네이블 상태로 EXLOCKEN이
설정되고, 현재 예외 레벨과
동일한 예외 레벨로 예외
복귀가 진행되는가?
Y
184:
PSTATE.EXLOCK이
현재 잠금 해제
상태인가?
N
Y
N
186:
결함을 시그널링함
188:
ELR_ELx를 PC로 그리고
SPSR_ELx를 PSTATE로 복사하여
예외 복귀
상태(PSTATE.EXLOCK을
포함함)를 복원함

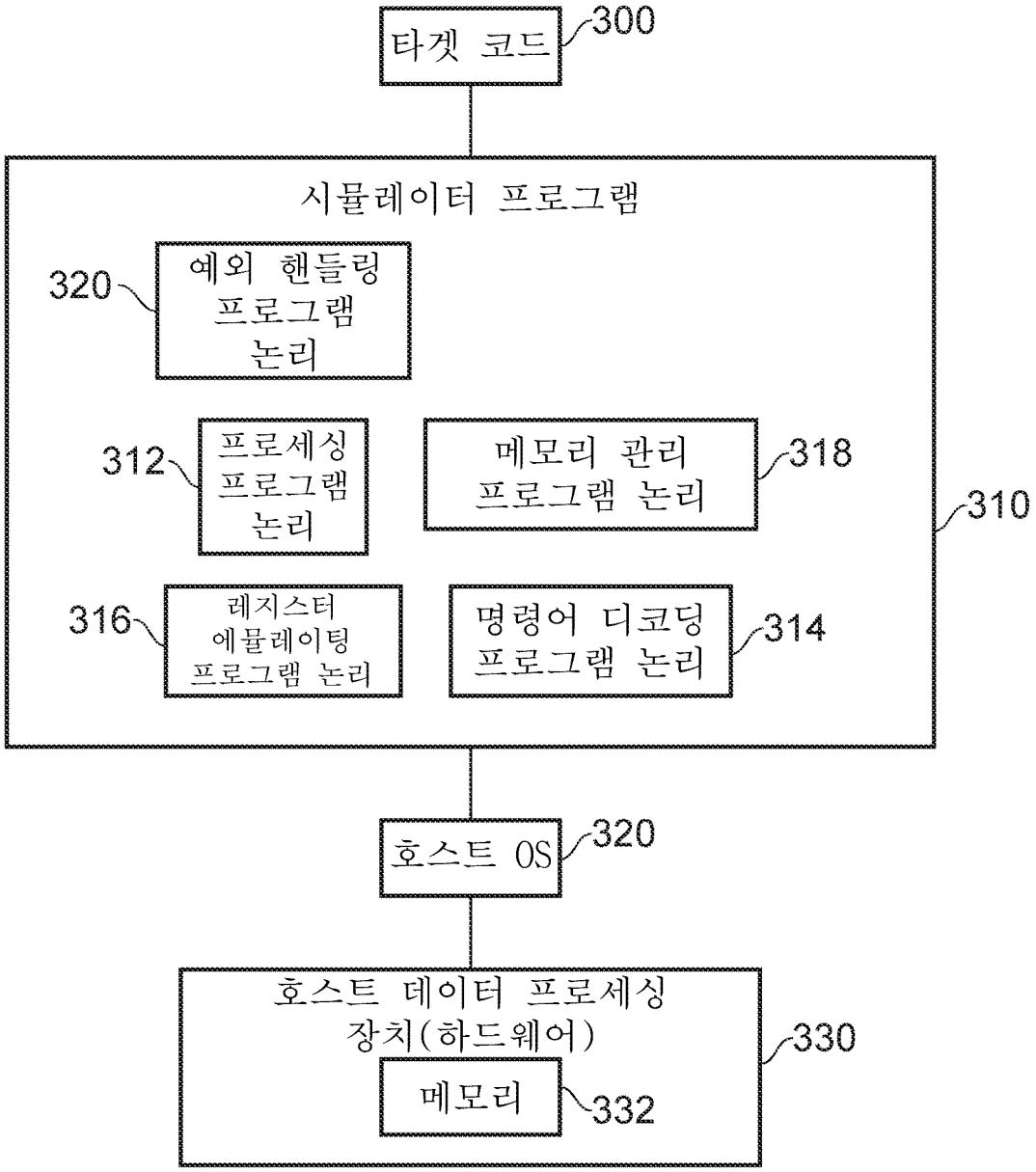
도면13


200:
ELR_ELx 또는 SPSR_ELx에서 예외 복귀 상태 정보의 수정을 명령어가 요청함
202:
GCS 모드가 이네이블되고 예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
N
Y
208:
결함을 시그널링함
Y
210:
PSTATE.EXLOCK이 현재 잠금 상태인가?
N
204:
(예를 들어, 현재 EL에 기초하여) 임의의 다른 검사가 충족되면, 예외 복귀 상태 정보에 대한 수정을 허용함

도면14


200:
ELR_ELx 또는 SPSR_ELx에서 예외 복귀 상태 정보의 수정을 명령어가 요청함
202:
GCS 모드가 이네이블되고 예외 복귀 상태의 잠금이 이네이블됨을 나타내는 이네이블 상태로 EXLOCKEN이 설정되는가?
N
Y
206:
PSTATE.EXLOCK이 현재 잠금 상태인가?
N
Y
208:
결함을 시그널링함
Y
210:
기록될 새로운 값이 ELR_ELx 또는 SPSR_ELx의 예외 복귀 정보의 오래된 값과 다른가?
N
204:
(예를 들어, 현재 EL에 기초하여) 임의의 다른 검사가 충족되면, 예외 복귀 상태 정보에 대한 수정을 허용함

도면15


타겟 코드
300
시뮬레이터 프로그램
320
예외 핸들링 프로그램 논리
312
프로세싱 프로그램 논리
메모리 관리 프로그램 논리
318
310
316
레지스터 에뮬레이팅 프로그램 논리
명령어 디코딩 프로그램 논리
314
호스트 OS
320
호스트 데이터 프로세싱 장치(하드웨어)
330
메모리
332