



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년08월24일
(11) 등록번호 10-1770949
(24) 등록일자 2017년08월18일

(51) 국제특허분류(Int. Cl.)
G06F 11/07 (2006.01) G06F 11/30 (2006.01)
G06F 11/34 (2006.01)
(52) CPC특허분류
G06F 11/0721 (2013.01)
G06F 11/0745 (2013.01)
(21) 출원번호 10-2016-7003158
(22) 출원일자(국제) 2014년07월23일
심사청구일자 2016년10월31일
(85) 번역문제출일자 2016년02월03일
(65) 공개번호 10-2016-0034939
(43) 공개일자 2016년03월30일
(86) 국제출원번호 PCT/US2014/047908
(87) 국제공개번호 WO 2015/013460
국제공개일자 2015년01월29일
(30) 우선권주장
61/857,571 2013년07월23일 미국(US)
14/338,279 2014년07월22일 미국(US)
(56) 선행기술조사문헌
US06802039 B1*
US06594785 B1*
US20040230877 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
샤참 아사프
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
라넬 이타이
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
하임 마야
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(74) 대리인
특허법인코리아나

전체 청구항 수 : 총 26 항

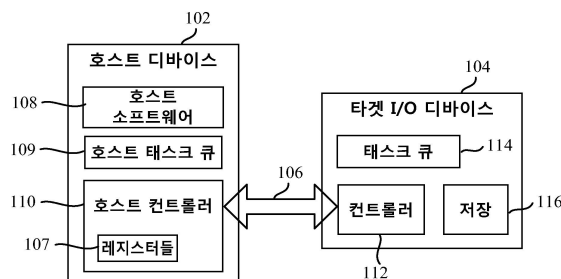
심사관 : 이동하

(54) 발명의 명칭 로버스트한 하드웨어/소프트웨어 에러 복구 시스템

(57) 요약

호스트 컨트롤러 및 호스트 소프트웨어가 협력하여 에러 검출 및 복구하는 방법이 제공된다. 호스트 컨트롤러는: 에러 조건을 검출하고, 에러 인터럽트 또는 레지스터를 세팅하며, 및/또는 호스트 컨트롤러에서 태스크 실행 또는 프로세싱을 정지할 수 있다. 호스트 소프트웨어는: 호스트 컨트롤러가 에러 인터럽트 또는 레지스터를 세팅한 결과로 에러 조건을 검출하고; 에러 핸들링을 수행하며, 및 에러 조건을 클리어할 수 있다. 그 다음, 호스트 컨트롤러는 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 태스크들의 실행 또는 프로세싱을 재개한다.

대표도 - 도1



(52) CPC특허분류

G06F 11/0772 (2013.01)

G06F 11/0793 (2013.01)

G06F 11/3024 (2013.01)

G06F 11/3409 (2013.01)

G06F 2201/86 (2013.01)

명세서

청구범위

청구항 1

호스트 컨트롤러로서,

하나 이상의 레지스터들에 액세스하기 위한 인터페이스;

상기 인터페이스에 커플링된 프로세싱 회로로서, 상기 프로세싱 회로는,

태스크들을 프로세싱하는 중 호스트 컨트롤러 하드웨어에 의해 에러 조건의 검출 시 에러 레지스터를 세팅하고;

상기 호스트 컨트롤러 하드웨어에 의한 태스크 실행 또는 태스크들의 프로세싱을 정지하며;

상기 프로세싱 회로에서 동작하는 호스트 소프트웨어에 의해 상기 에러 조건이 클리어되었다는 검출 시, 상기 호스트 컨트롤러 하드웨어에 의해 태스크들의 실행 또는 프로세싱을 재개하도록 구성된, 상기 프로세싱 회로를 포함하는 호스트 컨트롤러.

청구항 2

제 1 항에 있어서,

상기 프로세싱 회로는,

태스크 큐로부터 하나 이상의 태스크들을 프로세싱하고; 및

상기 태스크들을 프로세싱하면서 에러 조건들을 모니터링하도록 추가로 구성된, 호스트 컨트롤러.

청구항 3

제 1 항에 있어서,

상기 프로세싱 회로는,

상기 호스트 소프트웨어에 의해 상기 에러 조건이 클리어되었는지의 여부를 확인하기 위해 상기 에러 레지스터를 폴링하도록 추가로 구성된, 호스트 컨트롤러.

청구항 4

제 1 항에 있어서,

상기 호스트 컨트롤러는 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시 태스크들의 실행 또는 프로세싱을 정지하는, 호스트 컨트롤러.

청구항 5

제 1 항에 있어서,

상기 에러 레지스터는 상기 호스트 컨트롤러 외부의 에러 핸들링 소프트웨어에 액세스 가능한 공유 레지스터 공간 내에 위치한, 호스트 컨트롤러.

청구항 6

제 1 항에 있어서,

상기 에러 조건은 외부의 에러 핸들링 소프트웨어에 의해 클리어되는, 호스트 컨트롤러.

청구항 7

제 1 항에 있어서,

상기 태스크 실행 또는 태스크들의 프로세싱을 정지하는 것은 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시의 기지의 위치 또는 상태 내 하드웨어 동작들을 중지시키는 것을 포함하는, 호스트 컨트롤러.

청구항 8

호스트 컨트롤러에서 동작하는 방법으로서,

프로세싱 회로에 의해 태스크들을 프로세싱하는 중 호스트 컨트롤러 하드웨어에 의해 에러 조건의 검출 시 에러 레지스터를 세팅하는 단계;

상기 프로세싱 회로에 의한 태스크 실행 또는 태스크들의 프로세싱을 상기 호스트 컨트롤러 하드웨어에 의해 정지하는 단계; 및

상기 프로세싱 회로에서 동작하는 호스트 소프트웨어에 의해 상기 에러 조건이 클리어되었다는 검출 시 상기 호스트 컨트롤러 하드웨어에 의해 태스크들의 실행 또는 프로세싱을 재개하는 단계를 포함하는, 호스트 컨트롤러에서 동작하는 방법.

청구항 9

제 8 항에 있어서,

태스크 큐로부터 하나 이상의 태스크들을 프로세싱하는 단계; 및

상기 태스크들을 프로세싱하면서 에러 조건들을 모니터링하는 단계를 추가로 포함하는, 호스트 컨트롤러에서 동작하는 방법.

청구항 10

제 8 항에 있어서,

상기 호스트 소프트웨어에 의해 상기 에러 조건이 클리어되었는지의 여부를 확인하기 위해 상기 에러 레지스터를 폴딩하는 단계를 추가로 포함하는, 호스트 컨트롤러에서 동작하는 방법.

청구항 11

제 8 항에 있어서,

상기 호스트 컨트롤러는 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시 태스크들의 실행 또는 프로세싱을 정지하는, 호스트 컨트롤러에서 동작하는 방법.

청구항 12

제 8 항에 있어서,

상기 에러 레지스터는 상기 호스트 컨트롤러 외부의 에러 핸들링 소프트웨어에 액세스 가능한 공유 레지스터 공간 내에 위치한, 호스트 컨트롤러에서 동작하는 방법.

청구항 13

제 8 항에 있어서,

상기 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 상기 외부 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 단계를 추가로 포함하는, 호스트 컨트롤러에서 동작하는 방법.

청구항 14

하나 이상의 명령들을 가지는 비일시적 프로세서-판독가능 저장 매체로서,

상기 하나 이상의 명령들은, 적어도 하나의 프로세싱 회로에 의해 실행될 때, 상기 적어도 하나의 프로세싱 회로로 하여금:

프로세싱 회로에 의해 태스크들을 프로세싱하는 중 호스트 컨트롤러 하드웨어에 의해 에러 조건의 검출 시 에러 레지스터를 세팅하게 하고;

상기 프로세싱 회로에 의한 태스크 실행 또는 태스크들의 프로세싱을 호스트 컨트롤러 하드웨어에 의해 정지하게 하며; 및

상기 프로세싱 회로에서 동작하는 호스트 소프트웨어에 의해 상기 에러 조건이 클리어되었다는 검출 시 상기 호스트 컨트롤러 하드웨어에 의해 태스크들의 실행 또는 프로세싱을 재개하게 하는, 비일시적 프로세서-판독가능 저장 매체.

청구항 15

제 14 항에 있어서,

상기 하나 이상의 명령들은 또한 상기 적어도 하나의 프로세싱 회로로 하여금:

태스크 큐로부터 하나 이상의 태스크들을 프로세싱하게 하고; 및

상기 태스크들을 프로세싱하면서 에러 조건들을 모니터링하게 하는, 비일시적 프로세서-판독가능 저장 매체.

청구항 16

제 14 항에 있어서,

상기 하나 이상의 명령들은 또한 상기 적어도 하나의 프로세싱 회로로 하여금:

상기 에러 조건이 클리어되었는지의 여부를 확인하기 위해 상기 에러 레지스터를 폴딩하게 하는, 비일시적 프로세서-판독가능 저장 매체.

청구항 17

호스트 디바이스로서,

에러 레지스터를 포함한 하나 이상의 레지스터들;

태스크 실행 또는 태스크들의 프로세싱 중 에러를 검출하고 이러한 에러가 검출되면 상기 에러 레지스터를 세팅하도록 구성된 하드웨어를 포함하는 호스트 컨트롤러;

상기 호스트 컨트롤러 및 상기 하나 이상의 레지스터들에 커플링되며 호스트 소프트웨어를 실행하는 프로세싱 회로로서, 상기 호스트 소프트웨어는,

상기 호스트 컨트롤러에 의해 에러가 세팅되었는지의 여부를 확인하기 위해 상기 호스트 소프트웨어에 의해 상기 에러 레지스터를 폴딩하는 것;

상기 에러 레지스터가 세팅되었다는 검출 시 상기 호스트 소프트웨어에 의해 에러 핸들링을 수행하는 것; 및

에러 핸들링의 완료 시 호스트 소프트웨어에 의해 상기 에러 레지스터를 클리어하는 것에 의해 상기 호스트 컨트롤러 상에서 에러 핸들링을 수행하도록 구성된, 상기 프로세싱 회로를 포함하는 호스트 디바이스.

청구항 18

제 17 항에 있어서,

상기 프로세싱 회로 및 호스트 컨트롤러는 단일 반도체 디바이스로 통합되는, 호스트 디바이스.

청구항 19

제 17 항에 있어서,

상기 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나는:

상기 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 것;

상기 호스트 디바이스의 상기 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 것; 및

상기 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 상기 외부 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 것 중 적어도 하나를 추가로 수행하는, 호스트 디바이스.

청구항 20

호스트 디바이스 상의 호스트 소프트웨어에 의해 수행되는 방법으로서,

호스트 컨트롤러의 프로세싱 회로에 의해 하드웨어 레벨에서 에러가 세팅되었는지의 여부를 확인하기 위해 상기 호스트 소프트웨어에 의해 에러 레지스터를 폴링하는 단계;

상기 프로세싱 회로에 의해 상기 에러 레지스터가 세팅되었다는 검출 시 상기 호스트 소프트웨어에 의해 에러 핸들링을 수행하는 단계; 및

상기 프로세싱 회로에 의한 에러 핸들링의 완료 시 상기 호스트 소프트웨어에 의해 상기 에러 레지스터를 클리어하는 단계를 포함하는, 호스트 디바이스 상의 호스트 소프트웨어에 의해 수행되는 방법.

청구항 21

제 20 항에 있어서,

상기 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 단계;

상기 호스트 디바이스의 상기 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 단계; 및

상기 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 상기 외부 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 단계를 추가로 포함하는, 호스트 디바이스 상의 호스트 소프트웨어에 의해 수행되는 방법.

청구항 22

호스트 디바이스로서,

호스트 소프트웨어를 실행하는 프로세싱 회로;

상기 프로세싱 회로에 커플링된 호스트 컨트롤러로서, 상기 호스트 컨트롤러 및 호스트 소프트웨어는:

상기 호스트 컨트롤러에서 에러 조건을 하드웨어 레벨에서 검출하고;

상기 호스트 컨트롤러에 의해 에러 인터럽트 또는 레지스터를 하드웨어 레벨에서 세팅하며;

상기 호스트 컨트롤러에 의해 태스크 실행 또는 프로세싱을 정지하고;

상기 호스트 컨트롤러가 상기 에러 인터럽트 또는 레지스터를 세팅한 결과로 상기 호스트 소프트웨어에서 에러 조건을 검출하며;

상기 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나에 의해 에러 핸들링을 수행하고;

상기 호스트 소프트웨어에 의해 상기 에러 조건을 클리어하며; 및

상기 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 상기 호스트 컨트롤러에서 태스크들의 실행 또는 프로세싱을 재개하도록 구성된, 상기 호스트 컨트롤러를 포함하는, 호스트 디바이스.

청구항 23

제 22 항에 있어서,

상기 호스트 컨트롤러 및 호스트 소프트웨어는 상기 에러 인터럽트 또는 레지스터를 포함하는 공유 레지스터 공간에 액세스할 수 있는, 호스트 디바이스.

청구항 24

제 22 항에 있어서,

상기 호스트 소프트웨어에 의해 상기 에러 조건의 상세들을 확인하기 위해 검사 레지스터들을 추가로 포함하는, 호스트 디바이스.

청구항 25

호스트 디바이스 상에서 동작하는 방법으로서,

프로세싱 회로에 의해 호스트 컨트롤러에서 에러 조건을 하드웨어 레벨에서 검출하는 단계;

상기 프로세싱 회로에 의해 상기 호스트 컨트롤러에서 에러 인터럽트 또는 레지스터를 하드웨어 레벨에서 세팅하는 단계;

상기 호스트 컨트롤러에 의해 상기 호스트 컨트롤러에서 태스크 실행 또는 프로세싱을 정지하는 단계;

상기 호스트 컨트롤러가 상기 에러 인터럽트 또는 레지스터를 세팅한 결과로 호스트 소프트웨어에서 에러 조건을 검출하는 단계;

상기 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나에 의해 에러 핸들링을 수행하는 단계;

상기 호스트 소프트웨어에 의해 상기 에러 조건을 클리어하는 단계; 및

상기 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 상기 호스트 컨트롤러에서 태스크들의 실행 또는 프로세싱을 재개하는 단계를 포함하는, 호스트 디바이스 상에서 동작하는 방법.

청구항 26

제 25 항에 있어서,

상기 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나는:

상기 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 것;

상기 호스트 디바이스의 상기 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 것; 및

상기 호스트 컨트롤러가 통신하는 타겟 디바이스에 상기 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 것 중 적어도 하나를 추가로 수행하는, 호스트 디바이스 상에서 동작하는 방법.

발명의 설명

기술 분야

[0001] 35 U.S.C. § 119 하에서 우선권 주장

[0002] 본 실용특허출원은 “Robust Hardware/Software Error Recovery System” 라는 명칭으로 2013년 7월 23일에 출원된 미국 특허 가출원 제 61/857,571 호를 우선권 주장하며, 이는 양수인에게 양수되어 이로써 본 명세서에 명백히 참조로서 통합된다.

[0003] 여러 양태들은 일반적으로 하드웨어/소프트웨어 시스템들의 에러 복구에 관한 것이며, 더욱 상세하게는 하드웨어 및 소프트웨어에 걸쳐 에러 복구가 동기화되는 시스템, 방법, 및/또는 디바이스에 관한 것이다.

배경 기술

[0004] 호스트 시스템은 I/O 호스트 컨트롤러 (하드웨어), 입/출력 (I/O) 호스트 소프트웨어 (예컨대, 드라이버), 및 타겟 I/O 디바이스를 포함할 수 있다. 호스트 컨트롤러는 태스크들 (예컨대, 데이터, 커맨드들 등) 을 전송/수신하기 위해 타겟 I/O 디바이스와 통신할 수 있다. 호스트 소프트웨어는 호스트 컨트롤러에 태스크들을 제공할 수 있다.

[0005] 하드웨어 동작 에러들은 예를 들어 컨트롤러 내에서 발생하는 경우가 있다. 하드웨어는 소프트웨어에 비해 빠르게 작동하기 때문에, 소프트웨어로서는 개입, 예를 들어 이러한 에러들로부터 복구되거나 에러들을 정정하기가 어렵다. 많은 경우에, 개입을 요구했던 이벤트는 이미 지나가 있고, 소프트웨어가 반응하기까지 걸리는 시간 동안 호스트 시스템의 상태가 변해 있다. 더욱이, 소프트웨어가 그 상태 (예컨대, 호스트 컨트롤러

및 호스트 시스템의 상태)를 조사하려고 시도하는 때, 하드웨어(예컨대, 컨트롤러)는 이미 동작을 수행하고 있을 수 있다.

[0006] 따라서, 에러들에서 복구되기 위해 소프트웨어가 하드웨어의 동작에 효율적으로 개입하도록 하는 메커니즘을 제공할 필요성이 있다.

발명의 내용

과제의 해결 수단

[0007] 하나 이상의 레지스터들에 액세스하기 위한 인터페이스 및 인터페이스에 커플링된 프로세싱 회로를 포함하는 호스트 컨트롤러가 제공된다. 프로세싱 회로는 (a) 태스크 큐로부터 하나 이상의 태스크들을 프로세싱하고; (b) 태스크들을 프로세싱하면서 에러 조건들을 모니터링하며; (c) 태스크들을 프로세싱하는 중 에러 조건의 검출 시 에러 레지스터를 세팅하고; (d) 태스크 실행 또는 태스크들의 프로세싱을 정지하며; (e) 에러 조건이 클리어되었는지의 여부를 확인하기 위해 에러 레지스터를 폴링하고; 및/또는 (f) 에러 조건이 클리어되었다는 검출 시, 태스크들의 실행 또는 프로세싱을 재개하도록 구성될 수 있다. 호스트 컨트롤러는 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시 태스크들의 실행 또는 프로세싱을 정지할 수 있다. 에러 레지스터는 호스트 컨트롤러 외부의 에러 핸들링 소프트웨어에 액세스 가능한 공유 레지스터 공간 내에 위치할 수 있다. 한 예시에서, 에러 조건은 외부 에러 핸들링 소프트웨어에 의해 클리어될 수 있다. 일부 구현들에서, 태스크들은 외부 저장 디바이스에 관독하거나 기록하는 동작 중 적어도 하나를 포함할 수 있다.

[0008] 호스트 컨트롤러에서 동작하는 방법 또한 제공된다. 호스트 컨트롤러는 에러 조건들을 모니터링하면서 태스크 큐로부터 하나 이상의 태스크들을 프로세싱할 수 있다. 태스크들을 프로세싱하는 중 에러 조건의 검출 시 호스트 컨트롤러에 의해 에러 레지스터가 세팅될 수 있다. 추가적으로, 태스크 실행 또는 태스크들의 프로세싱은 호스트 컨트롤러에 의해 정지될 수 있다. 호스트 컨트롤러는 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시 태스크들의 실행 또는 프로세싱을 정지할 수 있다. 그 다음, 호스트 컨트롤러는 에러 조건이 클리어되었는지의 여부를 확인하기 위해 에러 레지스터(또는 일부 다른 플래그, 레지스터, 및/또는 인디케이터)를 폴링할 수 있다. 에러 레지스터는 호스트 컨트롤러 외부의 에러 핸들링 소프트웨어에 액세스 가능한 공유 레지스터 공간 내에 위치할 수 있다. 에러 조건이 클리어되었다는 검출 시, 호스트 컨트롤러는 태스크들의 실행 또는 프로세싱을 재개할 수 있다. 호스트 컨트롤러는 추가로, 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 외부 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령할 수 있다.

[0009] 하나 이상의 명령들을 가지는 비일시적 프로세서-관독가능 저장 매체 또한 제공되며, 하나 이상의 명령들은 적어도 하나의 프로세싱 회로에 의해 실행될 때, 적어도 하나의 프로세싱 회로로 하여금: (a) 태스크 큐로부터 하나 이상의 태스크들을 프로세싱하게 하고; (b) 태스크들을 프로세싱하면서 에러 조건들을 모니터링하게 하며; (c) 태스크들을 프로세싱하는 중 에러 조건의 검출 시 에러 레지스터를 세팅하게 하고; (d) 태스크 실행 또는 태스크들의 프로세싱을 정지하게 하며; (e) 에러 조건이 클리어되었는지의 여부를 확인하기 위해 에러 레지스터를 폴링하게 하고; 및/또는 (f) 에러 조건이 클리어되었다는 검출 시 태스크들의 실행 또는 프로세싱을 재개하게 한다.

[0010] 하나 이상의 레지스터들, 호스트 컨트롤러, 및 호스트 컨트롤러와 하나 이상의 레지스터들에 커플링된 프로세싱 회로를 포함하는 호스트 디바이스 또한 제공된다. 프로세싱 회로는 호스트 소프트웨어를 실행하는데, 호스트 소프트웨어는: (a) 호스트 컨트롤러에 의해 에러가 세팅되었는지의 여부를 확인하기 위해 에러 레지스터를 폴링하는 것; (b) 에러 레지스터가 세팅되었다는 검출 시 에러 핸들링을 수행하는 것; (c) 에러 핸들링의 완료 시 에러 레지스터를 클리어하는 것에 의해 호스트 컨트롤러 상에서 에러 핸들링을 수행하도록 구성될 수 있다. 프로세싱 회로 및 호스트 컨트롤러는 단일 반도체 디바이스로 통합될 수 있다. 한 예시에서, 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나는: (a) 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 것; (b) 호스트 디바이스의 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 것; 및/또는 (c) 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 것 중 적어도 하나를 추가로 수행할 수 있다.

[0011] 호스트 디바이스 상의 소프트웨어에 의해 수행되는 방법 또한 제공된다. 에러 레지스터는 호스트 컨트롤러에 의해 에러가 세팅되었는지의 여부를 확인하기 위해 호스트 소프트웨어에 의해 폴링될 수 있다. 에러 레지스터가 세팅되었다는 검출 시 에러 핸들링이 수행될 수 있다. 에러 레지스터는 에러 핸들링의 완료 시 클리어될 수 있다. 한 예시에서, 방법은 (a) 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클

리어하는 단계; (b) 호스트 디바이스의 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 단계; 및/또는 (c) 호스트 컨트롤러가 통신하는 외부 타겟 디바이스에 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 단계를 추가로 포함할 수 있다.

[0012] 추가적으로, 호스트 소프트웨어를 실행하는 프로세싱 회로 및 프로세싱 회로에 커플링된 호스트 컨트롤러를 포함하는 호스트 디바이스가 제공된다. 호스트 컨트롤러 및 호스트 소프트웨어는 (a) 호스트 컨트롤러에서 에러 조건을 검출하고; (b) 호스트 컨트롤러에 의해 에러 인터럽트 또는 레지스터를 세팅하며; (c) 호스트 컨트롤러에 의해 태스크 실행 또는 프로세싱을 정지하고; (d) 호스트 컨트롤러가 에러 인터럽트 또는 레지스터를 세팅한 결과로 호스트 소프트웨어에서 에러 조건을 검출하며; (e) 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나에 의해 에러 핸들링을 수행하고; (f) 호스트 소프트웨어에 의해 에러 조건을 클리어하며; 및/또는 (g) 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 호스트 컨트롤러에서 태스크들의 실행 또는 프로세싱을 재개하도록 구성된다. 호스트 컨트롤러 및 호스트 소프트웨어는 에러 인터럽트 또는 레지스터를 포함하는 공유 레지스터 공간에 액세스할 수 있다. 호스트 소프트웨어는 에러 조건의 상세들을 확인하기 위해 레지스터들을 검사할 수 있다.

[0013] 호스트 디바이스 상에서 동작하는 방법 또한 제공되며, (a) 호스트 컨트롤러에서 에러 조건을 검출하는 단계; (b) 호스트 컨트롤러에 의해 에러 인터럽트 또는 레지스터를 세팅하는 단계; (c) 호스트 컨트롤러에 의해 태스크 실행 또는 프로세싱을 정지하는 단계; (d) 호스트 컨트롤러가 에러 인터럽트 또는 레지스터를 세팅한 결과로 호스트 소프트웨어에서 에러 조건을 검출하는 단계; (e) 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나에 의해 에러 핸들링을 수행하는 단계; (f) 호스트 소프트웨어에 의해 에러 조건을 클리어하는 단계; 및/또는 (g) 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 호스트 컨트롤러에서 태스크들의 실행 또는 프로세싱을 재개하는 단계를 포함한다. 호스트 소프트웨어 및 호스트 컨트롤러 중 적어도 하나는 (a) 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 것; (b) 호스트 디바이스의 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 것; 및/또는 (c) 호스트 컨트롤러가 통신하는 타겟 디바이스에 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 것 중 적어도 하나를 추가로 수행할 수 있다.

도면의 간단한 설명

[0014] 도 1 은 소프트웨어가 에러를 복구하거나 정정할 수 있도록 에러 발생 시 호스트 디바이스가 동작들을 정지하도록 구성된 시스템으로서, 버스를 통해 외부 타겟 I/O 디바이스에 커플링된 호스트 디바이스를 포함하는 시스템의 블록 다이어그램이다.

도 2 는 하드웨어/소프트웨어 시스템 내 에러 검출의 예시적인 접근법을 도시한다.

도 3 은 태스크 시퀀스를 정지 및 폐기하기 위해 호스트 컨트롤러 상에서 동작하는 예시적인 방법을 도시한다.

도 4 는 하드웨어 및 소프트웨어 간에 에러 정정 및 복구를 용이하게 하도록 구성된 호스트 디바이스를 도시하는 블록 다이어그램이다.

도 5 는 에러 핸들링을 용이하게 하도록 구성될 수 있는 예시적인 호스트 컨트롤러의 블록 다이어그램이다.

도 6 은 에러 검출 및 복구를 수행하기 위해 호스트 컨트롤러 및 호스트 소프트웨어 간에 동작하는 예시적인 방법을 도시한다.

도 7 은 에러 검출 및 복구를 위해 호스트 컨트롤러에 의해 동작하는 예시적인 방법을 도시한다.

도 8 은 에러 검출 및 복구를 위해 호스트 소프트웨어에 의해 동작하는 예시적인 방법을 도시한다.

발명을 실시하기 위한 구체적인 내용

[0015] 첨부된 도면들과 연계하여 하기에 설명되는 설명은 여러 구성들의 설명으로서 의도된 것이며 본원에서 설명되는 개념들 및 특징들이 실시될 수도 있는 구성들만을 나타내도록 의도된 것은 아니다. 다음의 설명은 여러 개념들의 완전한 이해를 제공하기 위한 목적으로 특정 상세들을 포함한다. 그러나, 이들 개념들이 이들 특정 상세들 없이 실시될 수 있음이 당업자에게는 명백할 것이다. 일부 경우들에 있어서, 널리 공지된 회로들, 구조들, 기술들 및 구성요소들은 설명된 개념들 및 특징들을 불명료하게 하는 것을 회피하기 위해 블록 다이어그램 형태로 도시된다.

[0016] 본 개시물 전반에 걸쳐 나타난 여러 개념들은 통신 시스템들, 네트워크 구조들, 전자 디바이스들, 모바일 디바

이스들, 컴퓨팅 디바이스들, 및/또는 통신 표준들의 넓은 분야에 걸쳐 다양하게 구현될 수 있다.

[0017] **개요**

[0018] 본 개시물의 여러 특징들 및 양태들은 에러 발생 시 소프트웨어가 에러를 핸들링 (예컨대, 에러의 복구 또는 정정을 수행) 할 수 있도록 하드웨어의 동작을 동결, 정지 또는 유예시키는 것과 관계가 있다. 예를 들어, 호스트 디바이스는 호스트 컨트롤러 및 소프트웨어를 포함할 수 있는데, 여기서 호스트 컨트롤러는 에러 발생 시의 기지의 위치들 (예컨대, 큐의 특정 포인트) 및/또는 상태에서 하드웨어 동작들을 중지시키도록 구성 또는 설계된다. 호스트 컨트롤러 및 임의의 외부 타겟 디바이스(들)이 동결 또는 유예 상태일 때, 호스트 컨트롤러는 모든 활동을 동결, 정지, 또는 유예시켜 소프트웨어가 에러(들)을 조사하거나 핸들링할 수 있게 한다. 호스트 컨트롤러가 에러를 접하여 소프트웨어 인터럽트를 촉발시키면, 소프트웨어에 의해 활동을 재개하도록 명령받을 때까지 모든 하드웨어 활동을 정지한다. 마찬가지로, 소프트웨어가, 무슨 이유에서건, 하드웨어 활동에 개입하기로 결정하면, 소프트웨어는 호스트 컨트롤러에게 동작들을 정지 또는 일시 정지하도록 명령한다. 호스트 컨트롤러는 현재 커맨드를 완료하고 기지의 위치 (예컨대, 큐의 특정 포인트 내) 또는 상태에서 그 동작을 정지/유예시킬 수 있다. 외부 타겟 디바이스는 호스트 컨트롤러가 활동을 개시하기를 대기할 수 있으므로, 호스트 컨트롤러가 그 자신의 활동을 정지 또는 유예하면 타겟 디바이스 역시 정지 또는 유예된다. 호스트 컨트롤러는 현재 동결 또는 일시 정지 상태에 있음을 (예컨대, 호스트 디바이스 상에서 실행되는) 소프트웨어에 나타낼 수 있다. 실행되는 병렬 프로세스들이 없는 동안 (즉, 하드웨어가 정지 또는 유예되는 동안), 소프트웨어는 개입하고, 호스트 컨트롤러 및 타겟 디바이스 내 현재 상태를 조사하여, 에러 핸들링 프로시저 (예컨대, 에러 복구 프로시저, 에러 정정 프로시저, 또는 임의의 다른 프로시저) 를 실행시킬 수 있다. 소프트웨어에 의해 프로시저가 완료되면, 그것은 호스트 컨트롤러에 그것의 동작들을 재개하도록 지시할 수 있다. 하드웨어/소프트웨어 경합 조건들 및 불확실성들의 제거는 에러 복구 및 다른 하드웨어/소프트웨어 프로시저들을 단순화한다.

[0019] **예시적 동작 환경**

[0020] 도 1 은 소프트웨어가 에러를 복구하거나 정정할 수 있도록 에러 발생 시 호스트 디바이스가 동작들을 정지하도록 구성된 시스템으로서, 버스 (106) 를 통해 외부 타겟 I/O 디바이스 (104) 에 커플링된 호스트 디바이스 (102) 를 포함하는 시스템의 블록 다이어그램이다. 호스트 디바이스 (102) 는 호스트 소프트웨어 (108), 태스크 큐 (109), 하나 이상의 레지스터들 (107), 및/또는 호스트 컨트롤러 (110) 를 포함할 수 있다. 타겟 I/O 디바이스 (104) 는 컨트롤러 (112), 태스크 큐 (114), 및 저장 디바이스 (116) 를 포함할 수 있다. 호스트 태스크 큐 (109) 는 타겟 I/O 디바이스 (104) 에 전송되는 태스크들 (예컨대, 동작들 또는 커맨드들) 을 보유할 수 있다. 예를 들어, 이러한 호스트 태스크 큐 (109) 는 호스트 소프트웨어 (108) 에 의해 호스트 컨트롤러 (110) 에 태스크들을 제공하는데 사용될 수 있으며, 태스크들이 타겟 I/O 디바이스 (104) 에 전송될 때까지 그것들을 보유하는데 사용될 수 있다. (예컨대, 호스트 컨트롤러 내 또는 호스트 디바이스 (102) 내 다른 곳의) 레지스터들 (107) 은 소프트웨어 (108) 및/또는 호스트 컨트롤러 (110) 간에 상태 정보를 전송하도록 서브할 수 있다. 예를 들어, 레지스터들 (107) 은 호스트 태스크 큐 (109) 내 어떤 태스크들이 타겟 I/O 디바이스 (104) 에 전송되었는지 어떤 태스크들이 전송되지 않았는지에 대한 정보를 포함할 수 있다.

[0021] 한 구현에서, 타겟 I/O 디바이스 (104) 는 호스트 디바이스 (102) 로부터 분리되어 있거나 그 외부에 있을 수 있으며 또는 호스트 디바이스 (102) 와 함께 단일 반도체 칩의 일부로 통합될 수도 있다. 한 예시에서, 타겟 I/O 디바이스 (104) 는 JEDEC (Joint Electron Device Engineering Council)에 의한 eMMC (Embedded Multi-Media Controller) 규격에 부응하는 플래시 저장 디바이스일 수 있다. 이러한 예에서, 호스트 디바이스 (102) 는 타겟 I/O 디바이스 (104) 로의/로부터의 (예컨대, 호스트 태스크 큐 (109) 내에서 식별된) 데이터를 판독 및/또는 기록할 수 있다.

[0022] 도 2 는 하드웨어/소프트웨어 시스템 내 에러 검출의 예시적인 접근법을 도시한다. 호스트 소프트웨어 (108) 는 태스크들 $1 \cdots n$ 을 생성 (204) 하여, 호스트 컨트롤러 (110) 에 그들 (206a) 을 제공할 수 있다. 그 다음, 호스트 컨트롤러 (110) 는 태스크들 $1 \cdots n$ (206b) 을 타겟 컨트롤러 (112) 에 전송하는데, 타겟 컨트롤러 (112) 는 그들이 타겟 I/O 디바이스 (104) 에 의해 실행되거나 프로세싱 (208) 될 수 있는 태스크 큐 (114) 에 그들 (206c) 을 제공한다. 본원에 개시된 “태스크” 는 데이터 및/또는 비-데이터 태스크들일 수 있다.

타겟 I/O 디바이스 (104) 는 각 태스크의 실행 또는 프로세싱 시 호스트 디바이스 (102) 에 확인응답들 또는 인디케이터들을 되돌려 보낼 수 있다.

[0023] 한 특징에 있어서, 에러 검출 (210) 시 호스트 컨트롤러 (110) 는 그 실행 또는 프로세싱을 동결/정지 (212) 하도록 구성될 수 있다. 이러한 동결/정지 포인트는, 예를 들어, 현재 태스크 (예컨대, 커맨드들의 특정 조합/복수), 커맨드 (예컨대, 기본 명령들의 특정 조합/복수), 명령 (예컨대, 현재 명령) 의 프로세싱을 종료하는 시점일 수 있다. 그 다음, 호스트 컨트롤러 (110) 는 호스트 소프트웨어 (108) 에 의해 에러 인터럽트가 촉발/검출 (216) 되도록 유발하는 하나 이상의 에러 레지스터들 (214) 을 세팅한다. 호스트 소프트웨어 (108) 에 의해 이러한 에러 인터럽트의 검출 (216) 시, 호스트 소프트웨어 (108) 는 레지스터들로부터 에러 조건을 확인 (218) (예컨대, 에러의 타입을 결정하는 등) 하고, 태스크 시퀀스를 정지 (220) (예컨대, 태스크 생성 및/또는 프로세싱을 정지) 하고, 또한 에러 핸들링을 수행 (222) 할 수 있다. 한 예시에서, 이러한 에러 핸들링은 하나 이상의 오류 태스크들 $t \cdots w$ 을 클리어/제거하는 단계 (224a) 를 포함할 수 있다. 예를 들어, 이것은 호스트 태스크 큐 (109) 로부터 특정 태스크들 $t \cdots w$ 을 제거하는 단계 및/또는 호스트 컨트롤러 (110) 에 그들을 제거하거나 그러한 특정 태스크들을 실행하지 않도록 명령하는 단계를 수반할 수 있다. 추가적으로, 태스크들 $t \cdots w$ 이 호스트 소프트웨어 (108) 로부터 타겟 I/O 디바이스 (104) 에 이미 전송되면, 호스트 소프트웨어 (108) 는 타겟 디바이스 컨트롤러 (112) 에 하나 이상의 태스크들 $t \cdots w$ 을 클리어 및/또는 타겟 디바이스 태스크 큐 (114) 로부터 그들을 제거하라는 추가 메시지들을 전송 (224b) 할 수 있다.

[0024] 그의 에러 핸들링 동작들을 완료 (222) 시, 호스트 소프트웨어 (108) 는 그 후 에러 인터럽트 및/또는 에러 레지스터들을 클리어 (228) 하고 및/또는 호스트 컨트롤러 (110) 에게 그렇게 하도록 (230) 명령할 수 있다. 에러 핸들링이 완료되었다는 표시를 수신할 시, 호스트 컨트롤러는 호스트 태스크 큐 (109) 에 남아있는 태스크들의 실행을 재개 (232) 할 수 있다. 이 방식으로, 호스트 컨트롤러 (110) 는 그들 간 경합 조건들을 회피하면서, 호스트 소프트웨어 (108) 가 에러 복구를 핸들링하도록 협력한다.

[0025] 도 3 은 태스크 시퀀스를 정지 및 폐기하기 위해 호스트 컨트롤러 상에서 동작하는 예시적인 방법을 도시한다. 에러 조건의 검출 시, 호스트 소프트웨어는 정지 인디케이터를 발행 (302) 하거나 (예컨대, 제 1 구성 가능한 레지스터 내의 정지 비트에 “1” 을 기록), 호스트 컨트롤러에 “정지” 명령, 인디케이터, 플래그, 또는 인터럽트를 발행하기 위한 다른 방법을 사용할 수 있다. 그 다음, 호스트 소프트웨어는 정지 인디케이터 (예컨대, 제 1 구성 가능한 레지스터 내의 정지 비트) 가 완료 또는 세팅 (예컨대, “1” 또는 High로 세팅) 될 때까지 정지 인디케이터를 폴링 (304) 할 수 있다. 즉, 호스트 소프트웨어가 이전에 정지 인디케이터를 세팅했다라도, 그것은 정지 인디케이터가 실제로 세팅되었음을 확인하기 위해 모든 다른 프로세스들이 중지되었는지 확인하기 위해 검사해야만 할 수도 있다. 유의할 점은, 정지 인디케이터를 폴링하는 대신, 정지 인디케이터가 완전히 세팅되면 호스트 소프트웨어에 통지하는데 인터럽트, 또는 다른 통지 방법이 사용될 수 있다는 점이다.

[0026] 그 다음, 호스트 컨트롤러는 그 태스크 큐 내 추가적 태스크들이 있는지의 여부를 확인할 수 있다. 예를 들어, 호스트 컨트롤러는 실행될 또는 오류 중인 임의의 태스크들이 있는지의 여부를 확인하기 위해 태스크 레지스터를 판독 (306) 할 수 있다. 태스크 레지스터가 클리어 (태스크_레지스터 = 0) 하다면 모든 태스크들이 실행되었음 (태스크 큐 내 오류 태스크들이 없음) 을 의미하며, 이 때 호스트 컨트롤러는 정지 인디케이터를 클리어 (308) (예컨대, 제 1 구성 가능한 레지스터 내의 정지 비트는 “0” 또는 Low로 세팅된다) 하고 동작들을 재개할 수 있다.

[0027] 달리, 태스크 레지스터가 세팅 (예컨대, 태스크_레지스터 = “1” 또는 High) 되면, 호스트 컨트롤러는 태스크 큐 내 모든 태스크들을 클리어 또는 특정 태스크들을 클리어하기 위해 큐 클리어 인디케이터를 세팅 (310) 할 수 있다. 그 다음, 호스트 컨트롤러는 태스크들이 태스크 큐로부터 클리어되었다는 것을 확인하기 위해 큐 클리어 인디케이터 (또는 대응 레지스터) 를 폴링 (312) 할 수 있다. 이러한 에러 복구/정정 단계들이 완료되면, 호스트 컨트롤러는 정지 인디케이터를 리셋 (정지 비트 = 0 또는 Low) (314) 시키고, 태스크들의 실행 또는 프로세싱을 재개할 수 있다.

[0028] 예시적인 호스트 디바이스

[0029] 도 4 는 하드웨어 및 소프트웨어 간에 에러 정정 및 복구를 용이하게 하도록 구성된 호스트 디바이스를 도시하는 블록 다이어그램이다. 호스트 디바이스 (402) 는 프로세싱 회로 (404), 호스트 컨트롤러 (406), 프로세

서-판독가능 저장 매체/디바이스 (408), 메모리 디바이스 (430), 트랜시버 회로 (412), 및 버스 (410) 를 포함할 수 있다. 호스트 컨트롤러 (406) 는 메모리 디바이스 (430) 내 태스크 큐 (422) 로부터 태스크들을 획득하고 태스크들을 실행하며 및/또는 트랜시버 회로를 통해 태스크들 (예컨대, 데이터 및 커맨드들을 포함하는) 을 타겟 디바이스로 전송할 수 있다. 예를 들어, 호스트 컨트롤러 (406) 는 타겟 디바이스 (예컨대, 외부 저장 디바이스) 상에서 판독 및/또는 기록 동작들을 수행할 수 있다. 호스트 컨트롤러 (406) 는 호스트 컨트롤러 (406), 메모리 디바이스 (430) 또는 다른 곳 안에 있을 수 있는 하나 이상의 레지스터들 (420) 내에 프로세싱되고 있는 태스크들에 대한 상태 정보를 유지할 수 있다. 호스트 컨트롤러 (406) 는 에러의 검출 시 태스크 큐 (422) 내 태스크들의 프로세싱을 정지하도록 구성될 수 있으며, 에러 조건의 발생을 호스트 소프트웨어에 알리기 위해 소프트웨어 인터럽트를 촉발시키거나 플래그를 (예컨대, 레지스터 내에서) 세팅한다.

[0030] 프로세싱 회로는 호스트 소프트웨어 및/또는 기능적 회로들을 실행하도록 구성될 수 있다. 예를 들어, 에러 검출 모듈/회로 (414) 는 호스트 컨트롤러 (406) 가 에러 플래그를 세팅하거나 에러 인터럽트를 촉발시키는 때를 검출 또는 모니터링하도록 구성될 수 있다. 에러 핸들링 모듈/회로 (416) 는 호스트 컨트롤러 (406) 상에서 에러 복구 또는 정정을 수행하도록 구성될 수 있다. 예를 들어, 이러한 에러 핸들링 모듈/회로 (416) 는 태스크 큐 (422) 내 하나 이상의 태스크들을 클리어하고 및/또는 하나 이상의 레지스터들 (420) 을 리셋할 수 있다. 에러 조건 클리어링 모듈/회로 (418) 는 소프트웨어 인터럽트를 클리어하고, 호스트 컨트롤러 (406) 에 의해 세팅된 하나 이상의 플래그들을 클리어하고, 및/또는 이외의 경우 에러가 핸들링되었으며 호스트 컨트롤러 (406) 가 프로세싱을 재개할 수 있음을 호스트 컨트롤러 (406) 에 알릴 수 있다. 에러 핸들링이 클리어되었음을 통지받을 시, 호스트 컨트롤러 (406) 는 태스크 큐 내 태스크들의 프로세싱을 재개할 수 있다. 프로세서-판독가능 저장 매체/디바이스 (408) 는 그 기능들을 실행하기 위해 프로세싱 회로 (404) 에 의해 실행될 수 있는, 에러 검출 명령들, 에러 핸들링 명령들 (426), 및/또는 에러 조건 클리어링 명령들 (428) 을 포함할 수 있다.

[0031] 도 5 는 에러 핸들링을 용이하게 하도록 구성될 수 있는 예시적인 호스트 컨트롤러 (502) 의 블록 다이어그램이다. 본 예시에서, 호스트 컨트롤러 (502) 는 하나 이상의 레지스터들 (508) 에 커플링된 컨트롤러 프로세싱 회로 (504), 및/또는 입/출력 인터페이스 회로 (510) 를 포함할 수 있다. 컨트롤러 프로세싱 회로 (504) 는 태스크 프로세싱 모듈/회로, 에러 조건 검출 모듈/회로 (512), 태스크 정지 모듈/회로 (514), 에러 조건 인디케이터 모듈/회로 (516), 및/또는 에러 클리어링 폴링 모듈/회로 (518) 를 포함할 수 있다.

[0032] 태스크 프로세싱 모듈/회로는 태스크 큐로부터 태스크를 검색하고, 검색된 태스크를 프로세싱하며, 그 후 태스크 큐 내 다음 태스크를 프로세싱할 수 있다. 이러한 태스크들은, 예를 들어, 외부 타겟 디바이스로부터/로 판독하거나 기록하는 동작들을 수행하는 것을 포함할 수 있다. 에러 조건 검출 모듈/회로 (512) 는 태스크들을 프로세싱하는 중 에러가 발생했는지를 확인할 수 있다. 에러 조건의 검출에 반응하여, 태스크 정지 모듈/회로 (514) 는 호스트 컨트롤러 (502) 에 의해 추가적 프로세싱을 정지, 유예 및/또는 동결 (예컨대, 임의의 추가적 태스크들의 프로세싱을 정지) 시킬 수 있다. 컨트롤러 프로세싱 회로는 현재 태스크를 끝내거나 종료시킨 후, 임의의 추가적 프로세싱을 정지시킬 수 있다. 에러 조건 인디케이터 모듈/회로 (516) 는 플래그, 레지스터, 및/또는 하드웨어/소프트웨어 인터럽트를 세팅하여, 호스트 컨트롤러 (502) 에 의해 에러 조건이 검출되었음을 표시할 수 있다. 이것은 외부 또는 내부 소프트웨어가 에러 조건을 해결하기 위해 에러 핸들링을 수행하도록 허용할 수 있다. 에러 클리어링 폴링 모듈/회로 (518) 는 에러 조건이 핸들링 되어 태스크 프로세싱 모듈/회로가 동작을 재개할 수 있는지 및/또는 그 시기를 확인할 수 있다.

[0033] 한 예시에서, 하나 이상의 동작 명령들을 획득하기 위해 호스트 컨트롤러 (502) 는 (예컨대 I/O 인터페이스 회로 (510) 를 통해) 저장 디바이스 (506) 에 커플링될 수 있다. 예를 들어, 저장 디바이스 (506) 는 태스크 큐로부터의 태스크들을 프로세싱하는 태스크 프로세싱 명령들 (519), 태스크 프로세싱 중 에러들을 검출하는 에러 조건 검출 명령들 (520), 에러가 검출되면 태스크 프로세싱을 정지하는 태스크 정지 명령들 (522), 플래그, 레지스터, 또는 소프트웨어가 에러 복구의 목적으로 개입할 수 있게 하는 다른 인디케이터를 세팅하는 에러 조건 인디케이터 명령들 (524), 및/또는 호스트 컨트롤러가 사전에 세팅된 플래그, 레지스터 또는 (예컨대, 에러 조건이 해결되었음을 표시하기 위한) 인디케이터를 클리어하기 위해 모니터링, 검출 및/또는 폴링할 수 있게 하는 에러 클리어 폴링 명령들 (526) 을 포함할 수 있다.

[0034] 도 6 은 에러 검출 및 복구를 수행하기 위해 호스트 컨트롤러 및 호스트 소프트웨어 간에 동작하는 예시적인 방법을 도시한다. 호스트 컨트롤러는: (a) (예컨대, 호스트 소프트웨어로부터의 통지에 기초하여) 에러 조건을 검출 (602) 하고, (b) 에러 인터럽트 또는 레지스터를 세팅 (604) 하며, 및/또는 (c) 태스크 실행 또는 프로세싱을 정지 (606) 할 수 있다. 호스트 소프트웨어는: (d) 호스트 컨트롤러가 에러 인터럽트 또는 레지스터

를 세팅한 결과로 에러 조건을 검출 (608) 하고, (e) 에러 핸들링을 수행 (610) 하며, 및/또는 (에러 핸들링의 종료 시) 에러 조건을 클리어 (612) 할 수 있다. 그 다음, 호스트 컨트롤러는 호스트 소프트웨어에 의해 에러 조건이 클리어되었다는 검출 시 태스크들의 실행 또는 프로세싱을 재개 (614) 할 수 있다. 호스트 소프트웨어는 호스트 컨트롤러에 커플링된 프로세싱 회로에 의해 실행될 수 있음을 유의하여야 한다. 프로세싱 회로 및 호스트 컨트롤러는 단일 반도체 디바이스로 통합될 수 있다. 호스트 컨트롤러 및 호스트 소프트웨어는 에러 인터럽트 또는 레지스터를 포함하는 공유 레지스터 공간에 액세스할 수 있다.

[0035] 한 예시에서, 적어도 하나의 호스트 소프트웨어 및 호스트 컨트롤러에 의해 에러 핸들링을 수행하는 단계는: (i) 호스트 디바이스의 태스크 큐 상에 보류 중인 모든 태스크를 클리어하는 단계; (ii) 호스트 디바이스의 태스크 큐 상에 보류 중인 특정 태스크들을 클리어하는 단계; 및/또는 (iii) 호스트 컨트롤러가 통신하는 타겟 디바이스에 타겟 디바이스 내 일부 또는 모든 태스크를 클리어하도록 명령하는 단계 중 적어도 하나를 포함한다.

[0036] 적어도 하나의 호스트 소프트웨어 및 호스트 컨트롤러에 의해 에러 핸들링을 수행하는 다른 예시들에서, 보류 태스크들은 호스트 디바이스의 태스크 큐 및/또는 타겟 디바이스 내 태스크 큐로부터 클리어될 수 없거나 클리어될 필요가 없다.

[0037] 호스트 컨트롤러는 현재 태스크, 현재 커맨드, 및 현재 명령 중의 하나를 종료할 시 태스크 실행 또는 프로세싱을 정지할 수 있다는 것을 유의하여야 한다.

[0038] 도 7 은 에러 검출 및 복구를 위해 호스트 컨트롤러에 의해 동작하는 예시적인 방법을 도시한다. 호스트 컨트롤러는 (예컨대, 태스크 큐로부터의) 태스크들을 프로세싱 (702) 할 수 있다. 예를 들어, 태스크는 외부 타겟 디바이스로부터의/로의 판독 및/또는 기록 동작일 수 있다. 호스트 컨트롤러는 태스크들을 프로세싱하면서 에러 조건들을 모니터링 (704) 할 수 있다. 호스트 컨트롤러는 태스크들을 프로세싱하는 중 에러 조건의 검출 시 에러 레지스터를 세팅 (706) 할 수 있다. 추가적으로, 호스트 컨트롤러는 태스크 실행/프로세싱을 정지 (708) 한다. 호스트 컨트롤러에 의한 프로세싱의 정지는 또한 레지스터들의 상태를 동결시키므로, 호스트 소프트웨어에 의해 에러 핸들링이 수행될 수 있다. 정지된 동안, 호스트 컨트롤러는 에러 조건이 클리어되었는지의 여부를 확인하기 위해 에러 레지스터를 폴링 (710) 할 수 있다. 유의할 점은, 다른 예시들에서, 상이한 레지스터, 인디케이터, 플래그 또는 메커니즘이 에러 조건이 해결 또는 클리어되었음을 호스트 컨트롤러에 표시하는데 사용될 수 있다는 점이다. 에러 조건이 클리어되었다는 검출 시 호스트 컨트롤러는 실행/프로세싱을 재개 (712) 할 수 있다.

[0039] 도 8 은 에러 검출 및 복구를 위해 호스트 소프트웨어에 의해 동작하는 예시적인 방법을 도시한다. 에러 인터럽트 레지스터로부터 에러 인터럽트를 수신/검출 (802) 시, 호스트 소프트웨어는 최종 응답 및/또는 최종 응답 인덱스 레지스터를 판독 (804) 할 수 있다. 호스트 소프트웨어는 또한 실패한 태스크 인덱스 및 커맨드를 확인하기 위해 에러 핸들링 레지스터를 판독 (806) 할 수 있다. 그 다음, 호스트 소프트웨어는 태스크 정지 시퀀스 또는 동작들을 수행 (808) 하고 에러 검출 및 핸들링의 시퀀스를 진행 (810) 할 수 있다.

[0040] 앞서 검토된 양태들, 배열들, 및 실시예들이 특정 상세들 및 특정성에 따라 검토된 것에 반해, 도 1, 2, 3, 4, 5, 6, 7 및/또는 8 에 도시된 하나 이상의 구성요소들, 단계들, 특징들 및/또는 기능들은 단일 구성요소, 단계, 특징 또는 기능들로 재배열 및/또는 결합될 수 있으며 또는 몇몇 구성요소들, 단계들, 또는 기능들로 구현될 수 있다. 추가적인 요소들, 구성요소들, 단계들, 및/또는 기능들은 또한 본 개시물을 벗어나지 않으면서 추가되거나 이용되지 않을 수 있다. 도 1, 2, 4 및/또는 5 에 도시된 장치, 디바이스들 및/또는 구성요소들은 도 2, 3, 6, 7 및/또는 8 에 기술된 하나 이상의 방법들, 특징들, 파라미터들, 및/또는 단계들을 수행하거나 채용하도록 구성될 수 있다. 본원에서 기술된 신규 알고리즘들은 또한 소프트웨어 내에서 효율적으로 구현되거나 및/또는 하드웨어 내에 내장될 수 있다.

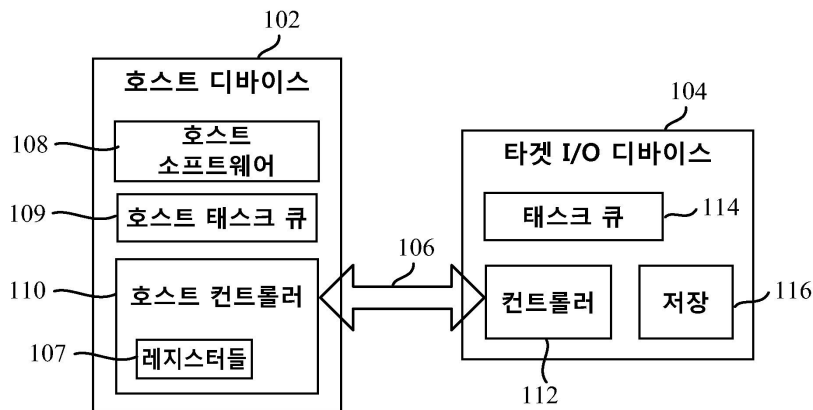
[0041] 또한, 적어도 일부 구현들은 플로우차트, 플로우 다이어그램, 구조 다이어그램, 또는 블록 다이어그램으로 나타내진 프로세스로서 기술되었다는 점이 유의된다. 플로우차트가 동작들을 일련의 프로세스로서 기술할 수 있더라도, 많은 동작들은 병렬적으로 또는 동시에 수행될 수 있다. 게다가, 동작들의 순서는 재배열될 수 있다. 프로세스는 그 동작들이 완료될 때 종료된다. 프로세스는 방법, 기능, 프로시저, 서브루틴, 서브프로그램 등에 대응할 수 있다. 프로세스가 기능에 대응할 때, 그 종단은 그 기능이 호출 기능 또는 메인 기능으로 리턴하는 것에 대응한다. 본원에 기술된 여러 방법들은 비-일시적 머신-판독가능, 컴퓨터-판독가능, 및/또는 프로세서-판독가능 저장 매체 내 저장될 수 있는 프로그래밍 (예컨대, 명령들 및/또는 데이터)에 의해 부분적으로 또는 완전히 구현될 수 있으며, 하나 이상의 프로세서들, 머신들, 및/또는 디바이스들에 의해 실행될 수 있다.

[0042] 본원에 개시된 실시예들에 관련하여 기술된 다양한 구체적 논리 블록들, 모듈들, 회로들, 및 알고리즘 단계들은 하드웨어, 소프트웨어, 펌웨어, 미들웨어, 마이크로코드, 또는 그들의 임의의 조합으로써 구현될 수 있음이 당업자에게 추가로 이해될 것이다. 이 호환성을 분명히 도시하기 위하여, 다양한 구체적 구성요소들, 블록들, 모듈들, 회로들, 및 단계들은 일반적으로 그들의 기능성에 관하여 상기에 기술되었다. 이러한 기능성이 하드웨어 또는 소프트웨어로써 구현될 수 있는지의 여부는 전체 시스템상에 부과된 특정 용도 및 설계 구속들에 의존한다.

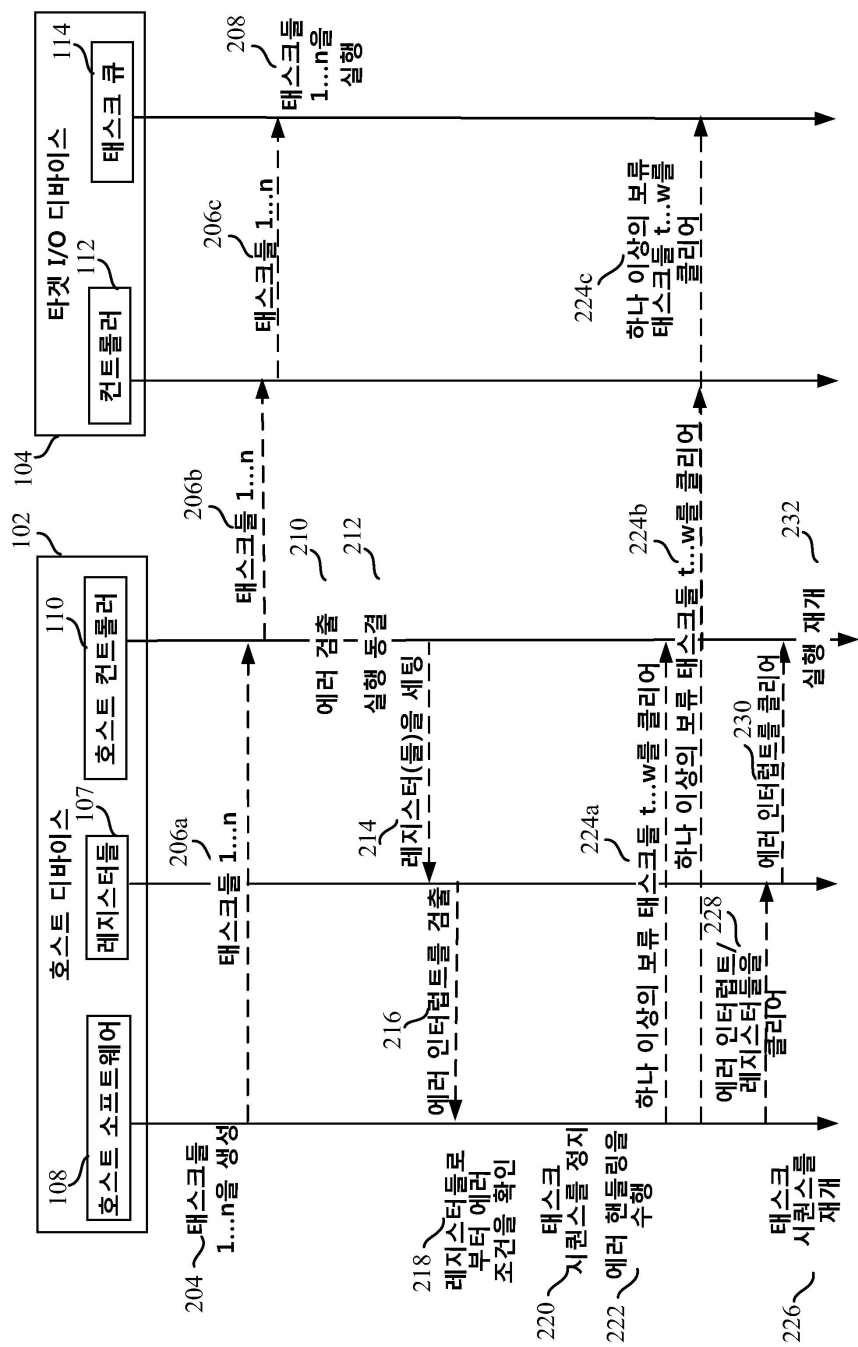
[0043] 본원에 기술되고 참조 도면들 상에서 보여진 예시들에 관한 다양한 특징들은 본 개시물의 범위를 벗어나지 않고 상이한 예시들 및 작업들 상에서 구현될 수 있다. 그러므로, 임의의 특정 구조물들 및 배열들이 기술되고 참조 도면들 상에서 보여졌더라도, 기술된 실시예들에 다양한 다른 추가들 및 변형들, 및 실시예들로부터의 삭제들이 통상의 당업자에게 자명할 것이므로, 이러한 실시예들은 단지 실례가 되는 것이며 본 개시물의 범위를 제한하는 것이 아니다. 따라서, 본 개시물의 범위는 오로지 다음과 같은 청구항들의 문자 언어, 및 법적 등가물들에 의하여 결정된다.

도면

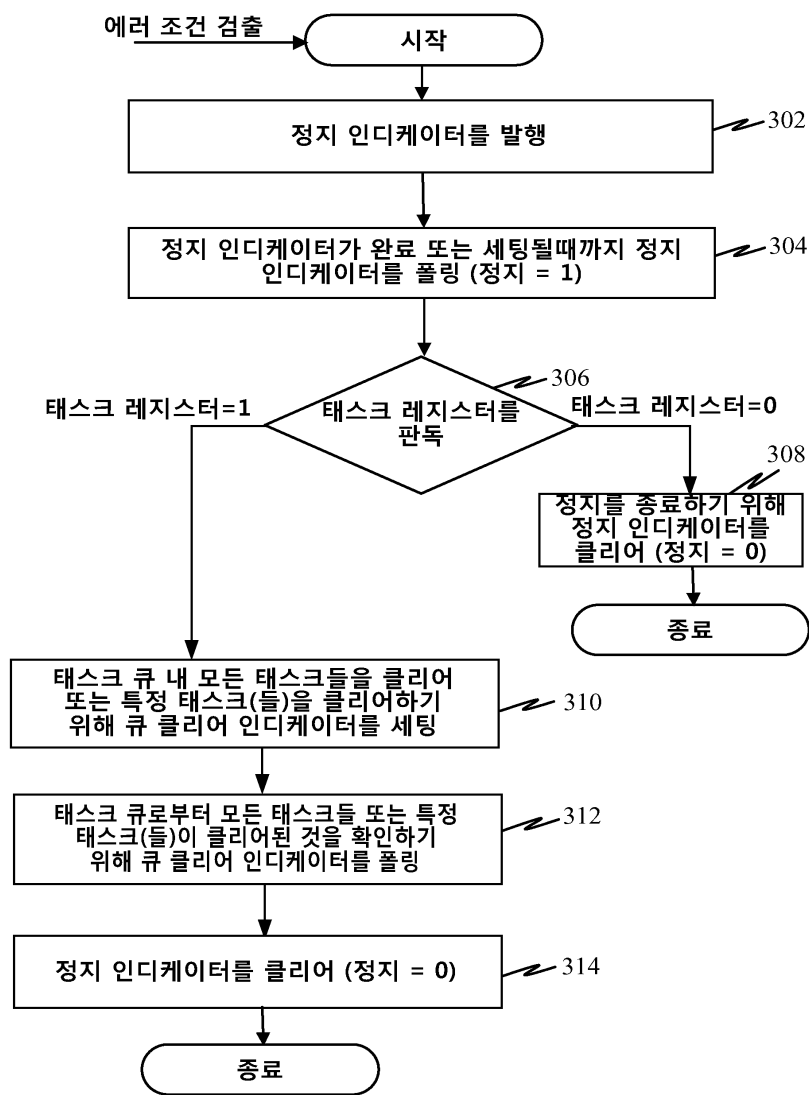
도면1



도면2

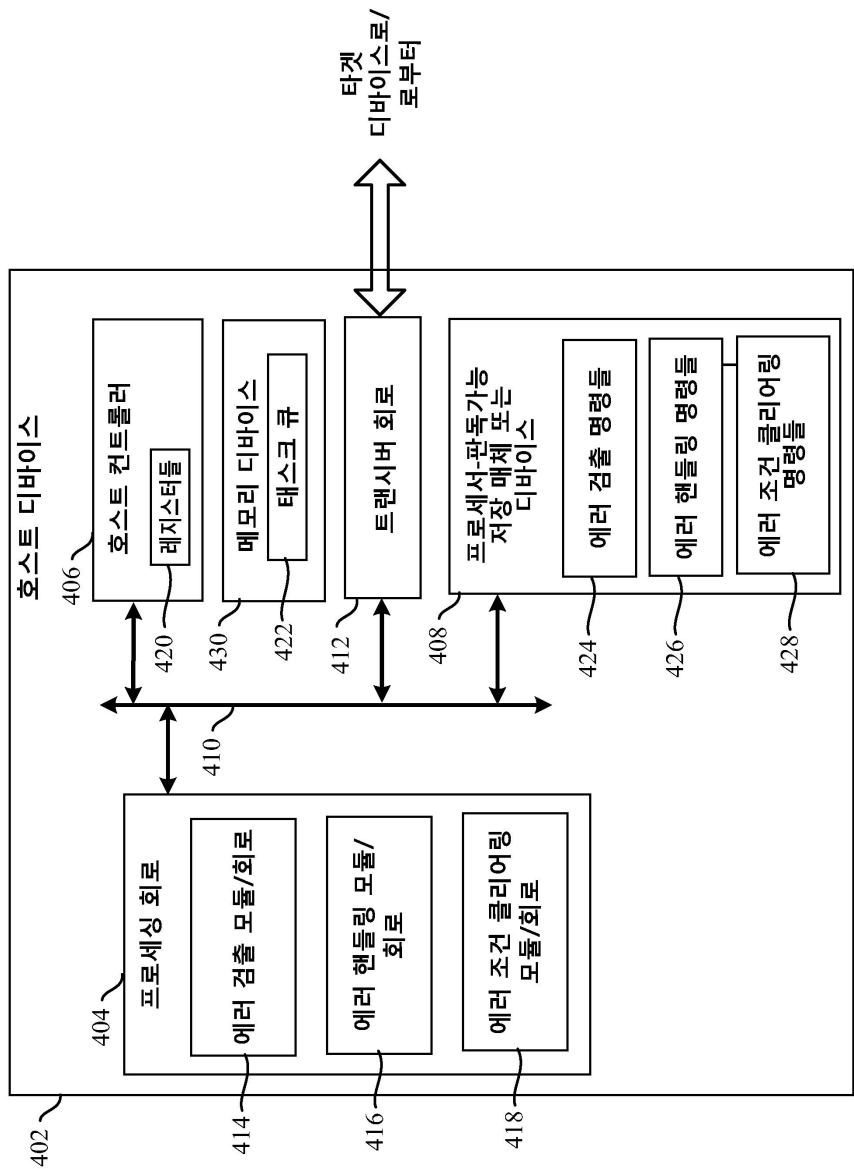


도면3

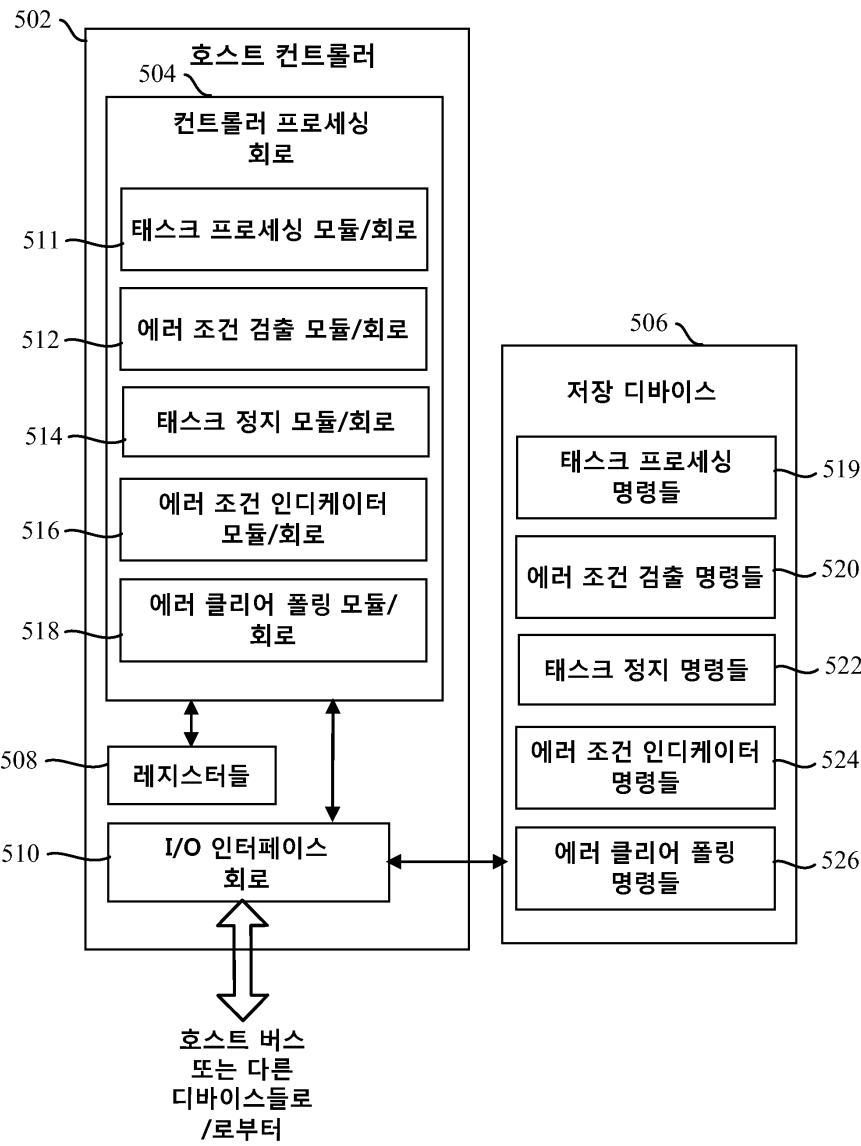


태스크 시퀀스의 정지 및
폐기 방법

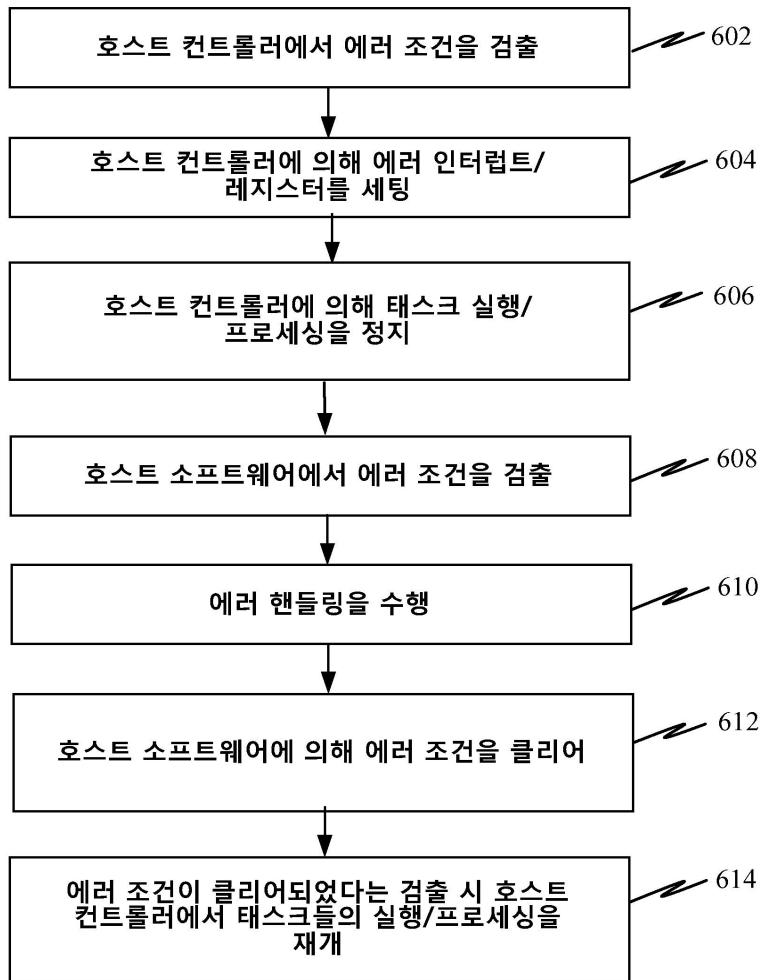
도면4



도면5

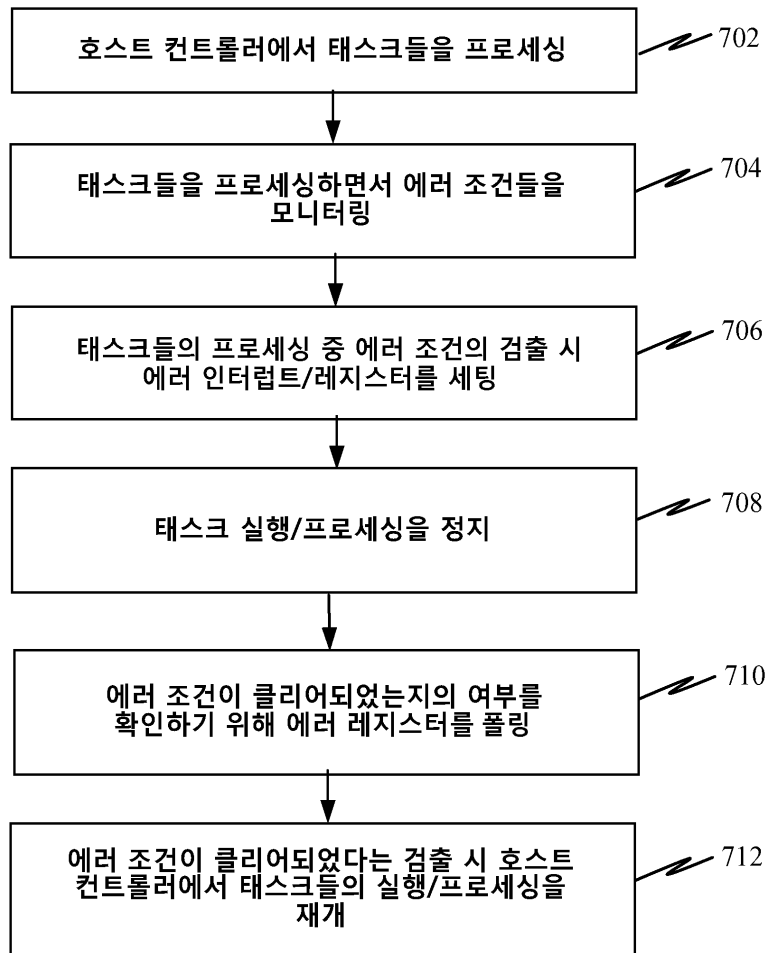


도면6



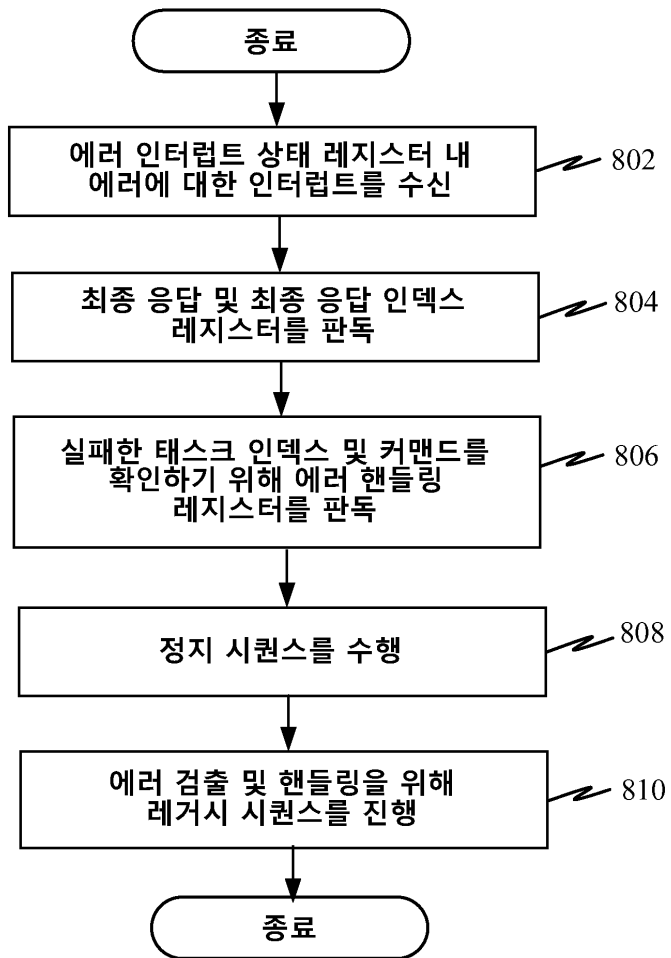
호스트 컨트롤러 및 소프트웨어
- 에러 검출 및 복구 방법

도면7



호스트 컨트롤러 - 에러 검출 및 복구 방법

도면8



호스트 소프트웨어 - 에러 검출 및 복구 방법