



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2013년11월11일
(11) 등록번호 10-1326316
(24) 등록일자 2013년10월31일

- (51) 국제특허분류(Int. Cl.)
F02D 41/22 (2006.01) F02D 45/00 (2006.01)
F02D 41/26 (2006.01)
- (21) 출원번호 10-2009-7007279
(22) 출원일자(국제) 2007년09월19일
심사청구일자 2012년08월22일
- (85) 번역문제출일자 2009년04월09일
(65) 공개번호 10-2009-0077773
(43) 공개일자 2009년07월15일
(86) 국제출원번호 PCT/EP2007/059904
(87) 국제공개번호 WO 2008/043650
국제공개일자 2008년04월17일
- (30) 우선권주장
10 2006 048 169.0 2006년10월10일 독일(DE)
- (56) 선행기술조사문헌
KR100441576 B1
US06705286 B1
- (73) 특허권자
로베르트 보쉬 게엠베하
독일 데-70442 스투트가르트 포스트파흐 30 02 20
- (72) 발명자
필러 베른트
독일 70839 게일링엔 슈탈러 슈트라세 38
팻찰 폴커
독일 73550 발트슈테텐/비쓰골딩엔 올란트슈트라
세 21
그멜리히 라이너
독일 71254 딘칭엔 회엔백 2
- (74) 대리인
안국찬, 양영준

전체 청구항 수 : 총 11 항

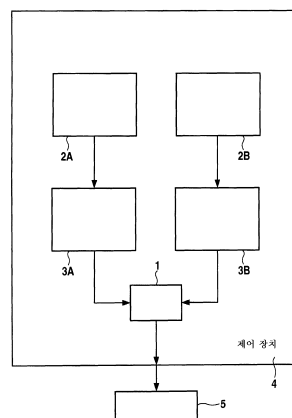
심사관 : 김길남

(54) 발명의 명칭 내연 기관의 엔진 제어 장치의 기능을 모니터링하기 위한 방법 및 장치

(57) 요약

본 발명은 내연 기관의, 복수의 실행 유닛(2A, 2B)을 구비한 엔진 제어 장치의 기능을 모니터링하기 위한 방법에 관한 것이다. 본 발명에서, 비교 작동 모드(VM)일 때 토크 모니터링 프로그램이 본 시스템의 복수의 실행 유닛(2)에서 실행되고, 모니터링 프로그램이 실행될 때 상기 실행 유닛(2)으로부터 송출되는 신호들은 에러의 인식을 위해 서로 비교된다.

대표도 - 도3



특허청구의 범위

청구항 1

복수의 실행 유닛들(2)을 구비한 시스템에서 실행되는, 제어 장치의 기능을 모니터링하기 위한 방법이며,

실행 유닛들(2)은 CPU, 코-프로세서, 디지털 신호 프로세서(DSP), 부동 소수점 계산 유닛(FPU), 또는 산술 논리 계산 유닛(ALU)에 의해 형성되며,

비교 작동 모드(VM)에서 동일한 모니터링 프로그램은 상기 시스템의 복수의 실행 유닛들(2)에서 중복 실행되고, 모니터링 프로그램이 실행될 때 상기 실행 유닛들(2)로부터 송출되는 신호들은 에러의 인식을 위해 서로 비교되고, 시스템은 모니터링 프로그램의 성공적인 실행 이후, 실행 유닛들(2)이 각종 프로그램들을 실행하는 성능 작동 모드(PM)로 전환되며, 성능 작동 모드(PM)에서 실행되는 프로그램들은 제어 장치를 실행시키는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 2

제1항에 있어서, 모니터링 프로그램은 엔진에 의해 발생하는 토크를 모니터링하는 토크 모니터링 프로그램으로 구성되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 3

제1항에 있어서, 제어 장치는 엔진 제어 장치에 의해 형성되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 4

제1항에 있어서, 모니터링 프로그램은 실행 유닛들(2)에서 동기식으로 실행되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 5

제1항에 있어서, 모니터링 프로그램은 실행 유닛들(2)에서 비동기식으로 실행되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 6

제1항에 있어서, 모니터링 프로그램은 주기적으로 실행되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 7

제1항에 있어서, 모니터링 프로그램이 실행될 때 실행 유닛들(2)로부터 송출되는 신호들이 서로 다른 경우, 모니터링 프로그램의 실행 시 에러가 인식되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 8

제7항에 있어서, 모니터링 프로그램의 실행 시 에러가 인식된 후, 제어 장치(4)에 의해 제어되는 유닛(5)이 스위치 오프되는, 제어 장치의 기능을 모니터링하기 위한 방법.

청구항 9

복수의 실행 유닛을 구비한 제어 장치(4)이며,

실행 유닛들(2)은 CPU, 코-프로세서, 디지털 신호 프로세서(DSP), 부동 소수점 계산 유닛(FPU), 또는 산술 논리 계산 유닛(ALU)에 의해 형성되며,

비교 작동 모드(VM)에서 동일한 모니터링 프로그램은 복수의 실행 유닛들(2)에서 중복 실행되고, 모니터링 프로그램이 실행될 때 상기 실행 유닛들(2)로부터 송출되는 신호들은 에러의 인식을 위해 서로 비교되고, 시스템은 모니터링 프로그램의 성공적인 실행 이후, 실행 유닛들(2)이 각종 프로그램들을 실행하는 성능 작동 모드(PM)로 전환되며, 성능 작동 모드(PM)에서 실행되는 프로그램들은 제어 장치를 실행시키는, 복수의 실행 유닛을 구비한 제어 장치.

청구항 10

제9항에 있어서, 모니터링 프로그램은 엔진에 의해 발생하는 토크를 모니터링하는 토크 모니터링 프로그램인, 복수의 실행 유닛을 구비한 제어 장치.

청구항 11

제9항에 있어서, 제어 장치(4)는 엔진 제어 장치인, 복수의 실행 유닛을 구비한 제어 장치.

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

명세서

기술분야

[0001] 본 발명은 복수의 실행 유닛을 구비한 시스템에서 실행되는 제어 장치의 기능을 모니터링하기 위한 방법에 관한 것이다.

배경기술

[0002] 임베디드 시스템의 영역에서는, 특히 자동차 기술 또는 자동화 기술에서는 하드웨어의 에러가 잠재적으로 안전성과 관련된 결과를 갖는 복수의 어플리케이션 또는 어플리케이션 프로그램들이 존재한다. 따라서, 상기 결과를 방지하거나 이의 영향을 감소시키기 위해, 상기 유형의 에러를 검출하기 위한 모니터링 수단이 사용된다. 상기 모니터링이 거의 영구적으로 필요한 어플리케이션이 존재한다. 다른 어플리케이션에서는 데이터 처리 시스템 또는 기타 하드웨어 부품들이 여전히 정확하게 기능하는지를 규칙적으로, 예를 들어 주기적으로 또는 특정 요건에 따라 검사하는 모니터링 기능들이 사용된다.

[0003] 도 1에는 엔진 제어 장치 영역에서의 종래의 모니터링 방법의 구조가 도시되어 있다. 엔진 제어 시 분사 시스템을 통해 연료가 연소실 내로 분사된다. 안전성의 관점으로부터 엔진 제어 장치 영역에서 상기 예시적인 어플리케이션은 3단계(E1, E2, E3)로 구조화된다. 분사 제어 장치의 어플리케이션 프로그램들은 원래 실행될 기능들을 포함하는 기초 또는 기본 단계(E1)를 형성한다. 분사 제어 장치는 얼마만큼의 연료가 정확히 어느 시점에 연소실에 분사되어야 하는지를 결정한다. 분사 제어 장치의 고장 시에는, 분사 제어 장치가 연료를 연소실에 너무 많이 또는 끊임없이 분사함으로써 자동차가 매우 빠르게 가속되어 사고가 일어날 수 있는 경우가 발생할 수 있다. 따라서 종래의 시스템에서는 분사 제어 장치가 단계(E1)에서 에러없이 작동하는지를 모니터링하는 모니터링 단계(E2)가 제공된다. 모니터링 단계(E2)는 필요한 경우 추가 센서들을 액세스하는 추가 소프트웨어 코드 또는 추가 프로그램에 의해 형성된다. 종래의 엔진 제어 장치에서 통상적으로 모니터링 단계(E2)는, 엔진을 통해 현재 발생하는 토크가 특정 한계값을 초과하지는 않는지 연속적인 토크 모니터링을 통해 형성된다. 종래의 엔진 제어 장치의 경우, 분사 제어 단계(E1)와 모니터링 단계(E2)의 프로그램들은 동일한 하드웨어 또는 동일한 실행 유닛에서 실행된다. 단계(E1)에서의 분사 제어 장치의 어플리케이션 프로그램과 단계(E2)에서의 토크 모니터링의 어플리케이션 프로그램이 동일한 실행 유닛 또는 CPU에서 실행되므로, 실행 유닛 내의 하드웨어 에러는 분사 제어 장치뿐만 아니라 토크 모니터링도 동시에 고장나는 결과를 초래할 수 있다. 따라서 안전성을 이유로 종래의 엔진 제어 장치에는 모니터링 단계(E2)가 흠결없이 기능하는지 검사하는 추가의 안전도 검사 단계(E3)가 제공된다. 안전도 검사 단계(E3)는 외부 하드웨어 부품, 예를 들어 ASIC를 구비한 실행 유닛의 질의 응답 커뮤니케이션을 실행하며, 이 경우, 기본적으로 실행 유닛 또는 마이크로컨트롤러의 기능, 특히 어플리케이션 프로그램의 기능이 모니터링 단계(E2) 내에서 검사된다. 모니터링 단계(E2)의 어플리케이션 프로그램은 타당성 검사를 실행한다. 예를 들어 모니터링 단계(E2)의 모니터링 프로그램은 가속 페달의 각도 위치(α)를 관독 입력한다. 분사 제어 단계(E1)의 어플리케이션 프로그램에 의해 주어지는 연료량이, 센서에 의해 모니터

링된 가속 페달 위치에 좌우되는 특정 한계값을 초과하는 경우, 단계(E2)에서 실행되는 모니터링 프로그램은 분사 제어 장치 내에 에러가 발생하는 것을 인식하고, 안전성을 이유로 대개 엔진이 스위치 오프되도록 지시한다. 모니터링 단계(E2)는, 예를 들어 엔진에서 발생하는 토크를 모니터링하여 한계값 초과시 엔진을 비활성화하는 토크 모니터링 프로그램을 추가로 포함한다. 모니터링 기능을 구현하기 위해 모니터링 프로그램의 코드는 단계(E2')로 복사되어 저장된다. 이 경우, 단계(E2')의 알고리즘 또는 프로그램은 디폴트 데이터 또는 테스트 데이터에 의해 완전히 검사된다. 예를 들어 ASIC, 즉 주문형 집적 회로에서 실행되는 안전도 검사 단계(E3) 프로그램은 질의로서 특정 비트 패턴을 실행 유닛 또는 CPU에 제공하며, 상기 실행 유닛 또는 CPU는 단계(E2')에 따른 카피된 모니터링 프로그램을 상기 디폴트값으로 완전히 검사하고 응답 비트 패턴을 주문형 집적 회로(ASIC)에서 단계(E3)의 안전성 프로그램에 송출한다. 안전성 프로그램은 CPU내의 모니터링 프로그램이 아직 에러없이 기능하는지 확인하기 위해 상기 응답 비트 패턴을 기준 비트 패턴과 비교한다. 주문형 집적 회로 내의 안전성 프로그램은 다른 하드웨어, 즉 ASIC에서, 실행 유닛 또는 CPU에서 실행되는 모니터링 프로그램으로서 실행된다. 따라서 상기의 종래 처리 방식은 CPU 내의 하드웨어 에러에 대해 일정 정도의 안전성을 형성한다.

[0004] 그러나 종래의 안전성 개념의 단점은 도 1에 도시된 바와 같이, 디폴트값 또는 테스트값에 의한 완전 검사에 대한 명령 테스트를 위한 모니터링 프로그램이 카피로써 존재해야 한다는 데 있다. 따라서 카피된 프로그램 명령을 저장하기 위한 메모리 공간이 모니터링 단계(E2')에서 필요하다.

[0005] 카피된 모니터링 단계(E2')를 위한 입력 데이터로서 디폴트 데이터 또는 테스트 데이터가 사용되는 종래의 명령 테스트의 다른 단점은 오퍼랜드(operand)에 따른 에러가 검출되지 않는다는 것이다.

발명의 상세한 설명

[0006] 따라서 본 발명의 목적은 제어 장치의 기능을 모니터링하기 위한 방법을 제공하는 것이며, 상기 방법은 오퍼랜드에 따른 에러도 검출한다.

[0007] 본 발명은 복수의 실행 유닛을 구비한 시스템에서 실행되는 제어 장치의 기능을 모니터링 하기 위한 방법을 제공하며, 비교 작동 모드(VM)에서 모니터링 프로그램은 시스템의 복수의 실행 유닛에서 실행되고, 모니터링 프로그램이 실행될 때 상기 실행 유닛으로부터 송출되는 신호들을 서로 비교하여 에러를 인식한다.

[0008] 본 발명에 따른 방법의 장점은 모니터링 프로그램의 카피된 프로그램 지시를 위해 메모리 공간이 낭비되지 않는다는 것이다.

[0009] 본 발명에 따른 방법의 실시예에서 모니터링 프로그램은 엔진에 의해 발생하는 토크를 모니터링하는 토크 모니터링 프로그램으로 구성된다.

[0010] 본 발명에 따른 방법의 실시예에서 제어 장치는 엔진 제어 장치를 통해 형성된다.

[0011] 본 발명에 따른 방법의 실시예에서 모니터링 프로그램은 동기식으로 실행 유닛에서 실행된다.

[0012] 본 발명에 따른 방법의 대안적인 실시예에서 모니터링 프로그램은 비동기식으로 실행 유닛에서 실행된다.

[0013] 본 발명에 따른 방법의 실시예에서 시스템은 모니터링 프로그램의 성공적인 실행 이후, 실행 유닛이 상이한 프로그램들을 실행하는 성능 작동 모드로 전환된다.

[0014] 본 발명에 따른 방법의 실시예에서 성능 작동 모드에서 실행되는 프로그램은 제어 장치를 실행시킨다.

[0015] 본 발명에 따른 방법의 실시예에서 모니터링 프로그램은 주기적으로 실행된다.

[0016] 본 발명에 따른 방법의 실시예에서 모니터링 프로그램이 실행될 때 실행 유닛으로부터 송출되는 신호들이 서로 다른 경우, 모니터링 프로그램의 실행 시 에러가 인식된다.

[0017] 본 발명에 따른 방법의 실시예에서는 모니터링 프로그램의 실행 시 에러가 인식된 후, 제어 장치에 의해 제어되는 유닛이 스위치 오프된다.

[0018] 또한 본 발명은 복수의 실행 유닛들을 구비한 제어 장치를 제공하며, 비교 작동 모드(VM)에서 모니터링 프로그램은 복수의 실행 유닛에서 실행되고, 모니터링 프로그램이 실행될 때 실행 유닛으로부터 송출되는 신호들은 에러를 인식하기 위해 서로 비교된다.

[0019] 본 발명에 따른 제어 장치의 실시예에서 모니터링 프로그램은 엔진에 의해 발생하는 토크를 모니터링하는 토크 모니터링 프로그램이다.

- [0020] 본 발명에 따른 제어 장치의 실시예에서 제어 장치는 엔진 제어 장치이다.
- [0021] 본 발명에 따른 제어 장치의 실시예에서 실행 유닛은 마이크로프로세서, 코-프로세서(co-processor), 디지털 신호 프로세서(DSP), 부동 소수점 계산 유닛(FPU), 또는 산술 논리 계산 유닛(ALU)에 의해 형성된다.
- [0022] 본 발명에 따른 방법과 본 발명에 따른 제어 장치의 바람직한 실시예는 본 발명에서 중요한 특징을 설명하기 위해 첨부된 도면들을 참조하여 이후 기술된다.

실시예

- [0028] 도 2에 도시된 바와 같이, 전환 유닛 및 비교 유닛(1)은 입력 측에서, N+1 실행 유닛(2)으로 이어지고, 실행 유닛(2-i)으로부터 입력 로직 신호들($E_0, E_1, E_2, E_3 \dots E_N$)을 수신한다. 전환 유닛 및 비교 유닛(1)은 비교 로직(1A)과 전환 로직(1B)을 포함한다.
- [0029] 도 2에 도시되어 있는 시스템은 2개의 이상의 작동 모드로 작동될 수 있다. 성능 작동 모드(PM)로도 불리는, 출력 상승을 위한 제1 작동 모드에서 실행 유닛(1-i) 또는 코어는 이에 평행하게 상이한 프로그램 또는 태스크를 처리한다. 실행 유닛(2-i)은, 계산 명령을 실행하기 위한 임의의 실행 유닛(2-i), 예를 들어 프로세서, 부동 소수점 계산 유닛(FPU), 디지털 신호 프로세서(DSP), 코-프로세서 또는 산술 논리 계산 유닛(ALU)일 수 있다. 성능 모드(PM)에서, 상이한 실행 유닛들(2-i)을 통한 프로그램의 진행은 동기식 또는 비동기식으로 실행될 수 있다. 출력 모드에서는 중복 처리가 실행되지 않고, 실행 유닛(2-i)은 여러 가지 계산 또는 프로그램을 평행하게 실행한다. 순수 성능 작동 모드(PM)에서 모든 입력 신호들(E_i)은 상응한 출력 신호들(A_i)로 전환되거나 안내된다.
- [0030] 슈퍼스칼라(superscalar) 계산 시스템의 사용과 더불어, 멀티 코어 아키텍처(multi core architecture)를 위한 제2 작동 모드는 복수의 실행 유닛들(2-i)이 동일한 프로그램을 중복 진행함으로써, 신호 처리의 안전성을 높이는 것이다. 안전성 모드나 안전 모드 또는 비교 작동 모드(VM)로도 불리는 제2 작동 모드에서 실행 유닛의 로직 출력 신호 또는 결과값이 전환 및 비교 회로(1)를 통해 서로 비교됨으로써, 발생된 에러 또는 신호 편차가 일치됨이 비교를 통해 인식될 수 있다. 따라서, 순수 비교 작동 모드(VM)에서는 모든 입력 신호(E_i)가 정확히 단 하나의 출력 신호(A_i)로 안내되거나 맵핑(mapping)된다. 혼합 형태들도 가능하다. 구성 가능한 전환 로직(1B)에서는 몇 개의 출력 라인 또는 출력 신호(A_i)가 제공되는지가 주어진다. 계속해서 전환 로직(1B)에서는, 어느 입력 신호들(E_i)이 출력 신호들(A_i) 중 어느 신호에 더해지는지가 저장된다. 따라서 전환 로직(1B)에는 입력 신호(E_i)를 여러 가지 출력 신호들(A_i)에 할당하는 맵핑 기능이 저장되어 있다.
- [0031] 처리 로직(1A)은 입력 신호가 어떤 형태로 각각의 출력 신호에 더해지는지를 각각의 출력 신호(A_i)에서 설정한다. 예를 들어 출력 신호(A_0)는 입력 신호들(E_1, \dots, E_N)을 통해 발생된다. " $m=1$ "일 때, 이는 간단히 입력 신호의 상호 연결 회로에 상응한다. " $m=2$ "일 때, 2개의 입력 신호들(E_1, E_2)은 서로 비교된다. 상기 비교는 회로(1)를 통해 동기식으로 또는 비동기식으로 실행될 수 있다. 이 경우, 상기 비교는 비트 단위로 실행되거나, 대안적으로는 두드러진 비트들만이 서로 비교된다. $M \geq 3$ 일 때, 여러 가지 가능성들이 존재한다. 제1 가능성은 모든 신호들이 서로 비교되고, 둘 이상의 여러 가지 값들이 존재할 때, 전환 및 비교 회로(1)를 통해 최적으로 신호화되는 에러가 검출되는 것이다. $K > M/2$ 일 때, K가 "m-선택"으로부터 실행될 추가의 가능성이 있다. 이는 한 실시예에서 비교와 보상을 통해 구현된다. 이 경우, 입력 신호들 중 하나가 다른 입력 신호들과 다르게 인식될 때, 선택적으로 제1 에러가 발생된다. 제1 에러 신호로부터의 여러 가지 제2 에러 신호에서는 3개의 모든 입력 신호가 서로 다를 수 있다. 추가 실시예에서는 입력 신호값이 예를 들어 평균값 또는 중앙값이나 에러 허용 알고리즘(FTA)을 실행하는 추가 계산 유닛에 공급된다. 에러 허용 알고리즘에서는 입력 신호값의 극값이 삭제되거나 무시되고, 나머지 신호값을 통한 평균값이 실행된다. 한 실시예에서 상기 평균값은 나머지 신호값의 전체량을 통해 달성된다. 한 대안적인 실시예에서 평균값은 하드웨어에서 쉽게 형성될 남아있는 신호값의 부분량을 통해 달성된다. 중간값 형성 시 덧셈과 나눗셈만이 실행되어야 하는 반면, FTM, FTA 또는 중앙값 형성은 부분적으로 입력 신호값의 분류를 요구한다. 한 실시예에서는 충분히 큰 신호 편차 또는 극값에서 선택적으로 에러 신호가 발생하거나 표시된다. 한 신호에 대한 신호 처리를 하기 위한 여러 가지로 언급되는 가능성들은 비교 연산을 나타낸다. 처리 로직(1A)은 실행될 비교 연산의 정확한 형태를 각각의 출력 신호(A_i)와, 이에 따라 마찬가지로 입력 신호(E_i)에 대해 설정한다. 전환 로직(1B) 내의 정보들의 조합은, 즉

처리 로직(1A)에서 주어지는 출력 신호 당 비교 연산 또는 기능값 당 비교 연산의 할당 기능은 작동 모드 정보를 나타내고, 상기 작동 모드를 설정한다. 상기 정보는 통상적으로 다가(multi-valued)로, 하나 이상의 로직 비트를 통해 나타난다. 2개의 실행 유닛들(2-i)만이 제공됨에 따라 단 하나의 비교 모드가 존재하는 경우, 작동 모드에서 전체 정보는 단 하나의 논리 비트로 보상될 수 있다.

[0032] 일반적으로 성능 작동 모드(PM)에서 비교 작동 모드(TM)으로의 시스템 전환은, 성능 작동 모드(PM)일 때 여러 가지 신호 출력들로 맵핑되거나 이에 상호 연결되는 실행 유닛들(2-i)이 비교 작동 모드(TM)일 때 동일한 신호 출력으로 맵핑되거나 이와 상호 연결됨으로써 이루어진다. 바람직하게 이는, 부분 량에서 고려될 모든 입력 신호들(E_i)이 성능 작동 모드(PM)일 때 상응하는 출력 신호(A_i)로 직접 전환되는 실행 유닛들(2-i)의 부분 량이 제공되는 반면, 비교 작동 모드(TM)일 때 입력 신호들이 모두 단 하나의 신호 출력으로 맵핑되거나 이와 상호 연결됨으로써 구현된다. 대안적으로, 전환 회로는 결합이 변화됨으로써 구현될 수 있다.

[0033] 여러 가지 작동 모드들 사이에서는 소프트웨어에 의해 제어되어, 작동 중에 다이내믹하게 전환될 수 있다. 상기 전환은 한 실시예에서 특수한 전환 지시 또는 전환 명령, 특수한 명령 시퀀스, 명백하게 표시되는 명령의 실행을 통해 개시되거나 시스템의 실행 유닛들(2-i) 중 하나 이상의 실행 유닛에 의한 특정 어드레스로의 액세스를 통해 개시된다.

[0034] 진행 및 검사가 중복 실행되는 안전성 모드(TM)와, 별도의 프로그램 진행을 통한 출력 상승이 달성되는 출력 또는 성능 작동 모드(PM) 간의 전환은 전환 장치(1)에 의해 실행된다. 한 실시예에서는 전환을 위해 프로그램, 어플리케이션 프로그램, 프로그램 부분 또는 마찬가지로 프로그램 지시의 표시가 실행되며, 이는 무엇을 통해 인식 가능한지, 상기 프로그램 지시들이 안전성에 관련된 것인지, 즉 안전성 작동 모드 또는 비교 작동 모드(TM)으로 진행되어야 하는지, 또는 출력 작동 모드 또는 쉽게 성능 작동 모드(PM)가 될 수 있는지의 특성값에 의해 이루어진다. 상기 표시는 프로그램 지시에서의 하나의 비트를 통해 실행될 수 있다. 대안적으로 특수한 프로그램 지시를 통해서도 이에 계속되는 시퀀스가 표시될 수 있다.

[0035] 안전성 작동 모드 또는 안전 모드(TM)에서 결과값의 계산 또는 실행 유닛(2-i)의 출력 신호는 동기식으로 진행될 때, 여러 가지 실행 유닛(2-i)으로 동일한 길이로 지속된다. 이 경우, 상기 결과값은 안전성 작동 모드(TM)에서 전환 회로(1)가 동기식으로 진행될 때, 동시적으로 제공된다. 상기 결과값이 일치하는 경우, 상응하는 데이터가 개방된다. 신호가 편차를 보일 때, 사전 설정된 예러 반작용이 실행된다.

[0036] 상기 시스템이 성능 작동 모드(PM)일 때, 프로그램은 병행 처리되고 전환 및 비교 회로(1) 내의 비교부 또는 비교기는 제어되지 않는다.

[0037] 복수의 실행 유닛(2)을 구비한 시스템에서 실행되는 제어 장치의 기능을 모니터링하기 위한 본 발명에 따른 방법에서 하나 이상의 모니터링 프로그램은 비교 작동 모드(TM)일 때, 복수 또는 심지어 모든 시스템 실행 유닛에서 실행된다. 모니터링 프로그램이 실행될 때 상기 실행 유닛(2)에서 송출되는 신호는 예러의 인식을 위해 서로 비교된다. 본 발명에 따른 제어 장치의 바람직한 실시예에서 이는 3개 이상의 실행 유닛(2)을 포함한다. 다른 신호들과 비교하여 가장 큰 신호 편차를 포함하는 상기 신호는 예를 들어 다수결에 의해 예러로써 인식된다. 한 실시예에서 상기 신호는 디지털 로직 신호, 특히 2진수 신호이다. 한 바람직한 실시예에서 본 발명에 따른 제어 장치(4)는 내연 기관의 제어를 위한 엔진 제어 장치이다. 대안적인 실시예에서 제어 장치(4)는 전동기의 제어를 위한 제어 장치이다. 모니터링 프로그램은 예를 들어 내연 기관이나 전동기를 통해 발생하는 토크를 모니터링하는 토크 모니터링 프로그램을 통해 형성된다. 이 경우, 상기 모니터링 프로그램은 동기식으로 또는 비동기식으로 실행 유닛(2)에 실행될 수 있다.

[0038] 본 발명에 따른 방법에서 엔진 제어 장치를 위한 정상적인 어플리케이션 프로그램은 성능 작동 모드(PM)일 때 실행되며, 즉 시스템의 각각의 실행 유닛(2)이 출력 상승을 위해 제어 장치를 위한 프로그램을 실행하는 반면, 다른 실행 유닛들(2)은 이와 상이한 어플리케이션 프로그램을 실행한다. 단계(E2)에서 실행되는 모니터링 프로그램은 본 발명에 따른 방법의 가능한 실시예에서 주기적으로 검색된다. 본 발명에 따른 방법에서 상기 모니터링 프로그램은 비교 작동 모드(TM)일 때, 시스템의 복수 실행 유닛들(2)에서 실행된다. 비교 작동 모드(TM)에서 시스템의 복수 또는 모든 실행 유닛들(2)은 동일한 모니터링 프로그램을 실행하며, 예러를 인식하기 위해 실행과 더불어 발생하는 출력 신호들은 서로 비교된다. 한 가능한 실시예에서 예를 들어 주기적으로 검색되는 복수의 모니터링 프로그램은 단계(E2)에서 실행된다. 검색되는 모든 모니터링 프로그램은 비교 작동 모드(TM)에서 실행된다. 한 대안적인 실시예에서 모니터링 프로그램은 특정 요건 또는 요건 지시로 검색되고, 이어서 비교 작동 모드(TM)에서 시스템의 복수의 또는 2개 이상의 실행 유닛에 의해 실행된다. 모니터링 프로그램을 실행

행하기 위한 상기 유형의 요건 지시는 예를 들어 인터럽트를 통해 개시될 수 있다.

- [0039] 모니터링 프로그램의 실행 이후, 시스템은 실행 유닛들(2)이 바람직하게 상이한 제1 단계(E1) 프로그램들, 예를 들어 제어 장치 프로그램을 실행하는 성능 작동 모드(PM)로 재전환된다.
- [0040] 본 발명에 따른 방법의 한 실시예에서는 비교 작동 모드(TM)에서 모니터링 프로그램이 실행될 때 실행 유닛(2)으로부터 송출되는 신호들이 서로 다른 경우, 단계(E2)에서 모니터링 프로그램이 실행될 때 에러가 인식된다. 이 경우, 바람직하게 모니터링 프로그램이 실행될 때 오류 인식 이후, 제어 장치(4)에 의해 제어된 유닛(5), 예를 들어 엔진은 스위치 오프된다.
- [0041] 도 3에는 본 발명에 따른 제어 장치 시스템의 가능한 실시예의 블록 회로도(도 3)가 도시되어 있다. 도 3에 도시된 실시예에서 본 발명에 따른 제어 장치(4)는 2개의 실행 유닛들(2A, 2B)을 포함한다. 실행 유닛들(2A, 2B)은 완전한 마이크로프로세서 또는 CPU나, 코-프로세서, 디지털 신호 프로세서(DSP), 부동 소수점 계산 유닛(FPU)이나, 산술 논리 계산 유닛(ALU)일 수 있다. 본 발명에 따른 제어 장치(4)의 추가의 실시예에서는 2개 이상의 실행 유닛들(2)이 제공된다. 도 3에 도시된 간단한 실시예에서는 실행 유닛들(2A, 2B)에 의해 발생된 신호들이 각각 버퍼 메모리(3A, 3B)에 일시 저장된다. 바람직하게 각각의 실행 유닛(2)은 하나의 고유한 버퍼 메모리(3)를 출력측에 포함한다. 상기 실행 유닛들(2A, 2B)의 일시 저장된 결과값 또는 출력 신호는 비교 유닛(1)에 공급된다. 예를 들어 비교 유닛(4)은 도 2에 도시된 바와 같이, 전환 및 비교 회로(1)를 통해 형성될 수 있다. 일시 저장된 출력 신호의 비교는 상응하는 비교 프로그램 또는 비교 소프트웨어의 진행을 통해서 또는 고정 배선으로 하드웨어에서 실행될 수 있다.
- [0042] 도 4에는 제어 장치의 기능을 모니터링하기 위한 본 발명에 따른 방법의 가능한 실시예의 흐름도가 도시되어 있다.
- [0043] 제2 단계(E2)에서 모니터링 프로그램이 검색을 실시한 이후, 단계(S1)에서는 성능 작동 모드(PM)에서 비교 작동 모드(TM)으로의 시스템 전환이 실행된다. 이어서 단계들(S2, S3)에서 2개의 실행 유닛들(2A, 2B)이 도 3에 도시된 바와 같이 동일한 모니터링 프로그램을 실행하기 위해 활성화되고, 동일한 모니터링 프로그램, 예를 들어 토크 모니터링 프로그램을 실행한다. 도 4에 도시된 실시예에서 2개의 실행 유닛들(2A, 2B)은 단계들(S4, S5)에서 해당 버퍼 메모리(3A, 3B)에 일시 저장되는 상응하는 결과 신호를 비동기식으로 단계들(S2, S3)에서 계산한다. 한 대안적인 실시예에서 2개의 실행 유닛들(2A, 2B)은 단계들(S2, S3)에서 해당 출력 신호 또는 결과값을 동기식으로 서로 계산한다. 2개의 결과값 또는 출력 신호의 존재 이후, 단계(S6)에서는 바람직하게 전환 및 비교 회로(1)를 통해 2개의 출력 신호들 간의 비교가 실행된다. 2개의 신호들이 서로 다른 경우, 에러가 인식된 후, 상응하는 에러 처리가 실행된다. 안전성에 관련된 어플리케이션에서, 제어 장치(4)에 의해 제어되는 유닛(5), 예를 들어 엔진은 스위치 오프된다. 단계(S6)에서의 비교는 이어지는 버퍼 메모리(3A, 3B) 판독 이후 상응하는 소프트웨어 당 비교 연산에 의해 실행될 수 있거나, 한 대안적인 실시예에서 상기 비교가 고정 배선된 회로를 통해 실행된다.
- [0044] 도 5에는 본 발명에 따른 방법의 가능한 실시예를 설명하기 위한 시간 흐름도가 도시되어 있다. 상기 실시예에서 모니터링 프로그램은 단계(E2)에서 주기적으로 검색되고, 비교 작동 모드(TM)일 때 복수의 실행 유닛(2)을 통해 동시에 실행된다. 모니터링 프로그램이 실행된 후 성능 작동 모드(PM)에서 시스템은 귀환되고, 단계(E1)에서 원래의 제어 장치 프로그램이 실행된다.
- [0045] 본 발명에 따른 제어 장치(4)의 한 대안적인 실시예에서 제어 장치(4)는 항상 성능 작동 모드(PM)로 작동되며, 2개 이상의 실행 유닛(2)을 구비한 모니터링 프로그램은 동기식으로 계산된다. 이 경우, 실행 유닛으로부터 발생하는 결과값 또는 출력 신호는 에러 인식을 위해 서로 비교된다. 그러나 상기 실시예에서 결과값들은 각각 일시 저장되어야 하고, 이어서 결과값들은 2차례 서로 비교되는데 즉, 혹시 일어날지 모르는 2개 실행 유닛(2)의 하드웨어 에러를 고려하기 위해, 한번은 제1 실행 유닛(2A)에서 또 한번은 제2 실행 유닛(2B)에서 서로 비교된다. 따라서 상기 실시예는 모니터링 프로그램이 비교 작동 모드(TM)로 실행되는 실시예 보다 더 복잡하다.
- [0046] 본 발명에 따른 방법은 오퍼랜드에 따른 에러의 검출도 가능하게 한다. 또한 본 발명에 따른 상기 방법은 도 1에 도시된 종래의 안전성 컨셉과 비교하여 확실히 메모리 공간을 절약할 수 있게 해준다.
- [0047] 본 발명에 따른 제어 장치(4)의 한 가능한 실시예에서 이 제어 장치는 3개 이상의 실행 유닛들(2)을 포함하며, 신호에 편차가 있을 때 다수결을 이용하여 어느 실행 유닛(2)이 에러 작동하는지 추정, 확인될 수 있다. 이어서 상기 실행 유닛(2)은 이 실행 유닛(2)이 실제로 고장 났는지를 확인하기 위해 바람직하게 자체 검사를 실행한다. 한 실시예에서는 자체 검사 결과, 실행 유닛(2)이 실제로 고장났다고 판단되면 실행 유닛(2)은 비활성화

된다. 따라서 상기 실시예에서 시스템은 에러를 허용하면서도 작동된다.

[0048] 영구적인 에러로부터 보호하기 위해 2개의 코어들 또는 실행 유닛(2)에서는, 예를 들어 이들이 가공 에러를 통해 발생될 때와 같이, 본 발명에 따른 방법의 가능한 한 실시예에서 각각 하나씩의 실행 유닛(2) 자체 검사가 실행된다.

[0049] 본 발명에 따른 방법의 가능한 실시예에서 모니터링 프로그램의 실행은 단계(E2)에서 비교 작동 모드(VM)로 실행되며, 모니터링 프로그램의 기능을 모니터링하기 위한 명령 테스트를 계속 실행하는 안전도 검사 단계(E3)가 추가의 보호를 위해 추가 제공된다. 상기 유형의 실시예는 특히 안전성과 관련된 어플리케이션에서 가능하다.

도면의 간단한 설명

[0023] 도 1은 종래의 안전성 컨셉을 3단계로 도시하는 흐름도이다.

[0024] 도 2는 본 발명에 따른 방법에서 사용된 전환 유닛 및 비교 유닛의 블록 회로도이다.

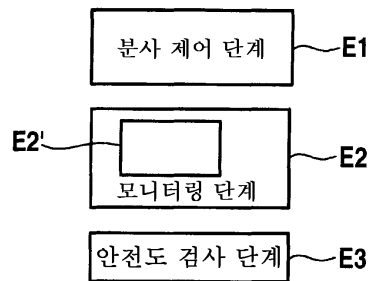
[0025] 도 3은 본 발명에 따른 제어 장치의 가능한 실시예를 도시하기 위한 블록 회로도이다.

[0026] 도 4는 본 발명에 따른 방법을 설명하기 위한 흐름도이다.

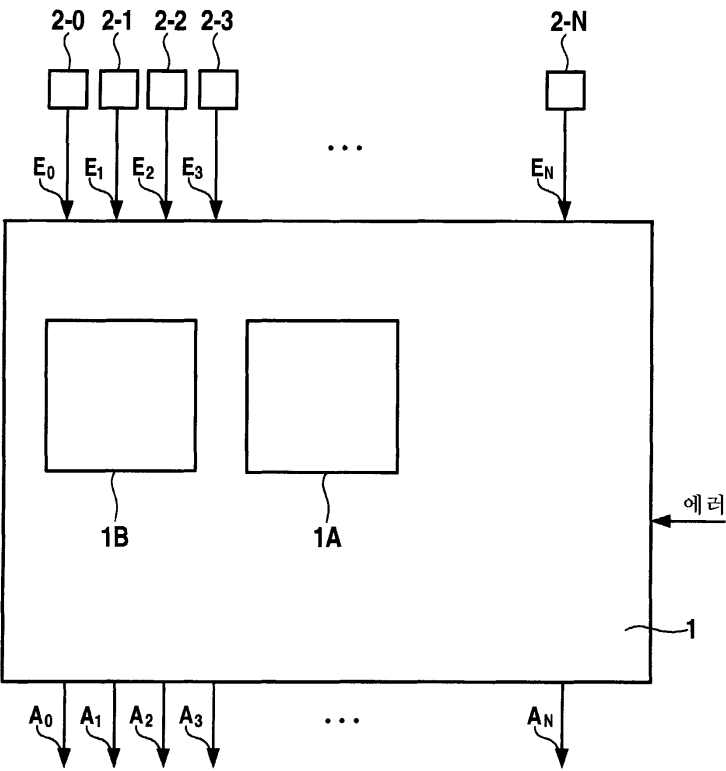
[0027] 도 5는 본 발명에 따른 방법의 가능한 실시예를 설명하기 위한 시간 흐름도이다.

도면

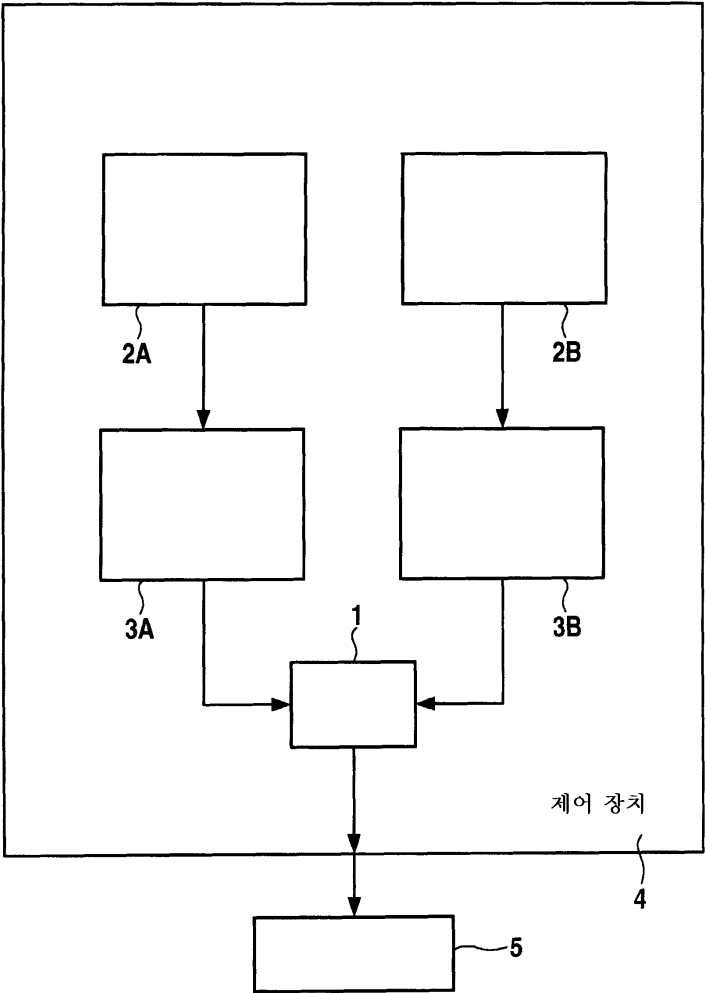
도면1



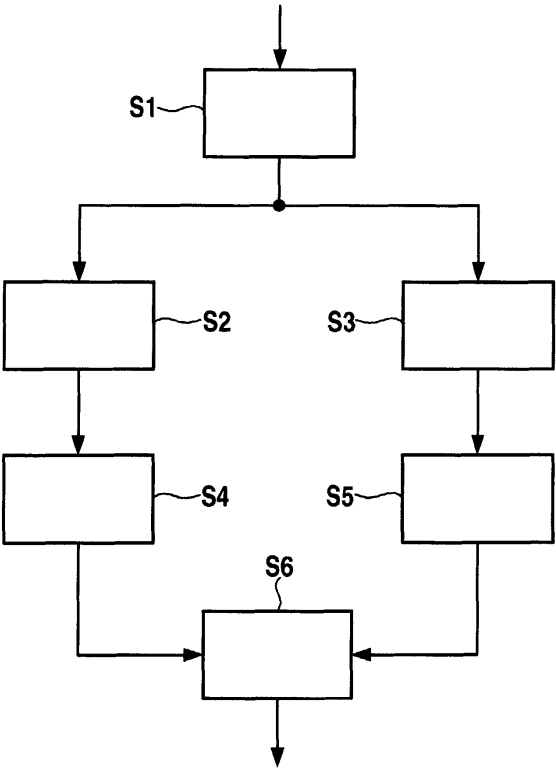
도면2



도면3



도면4



도면5

