(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년07월09일
(11) 등록번호 10-1998469
(24) 등록일자 2019년07월03일

(51) 국제특허분류(Int. Cl.)
G06F 11/07 (2006.01)
(52) CPC특허분류
G06F 11/0724 (2013.01)
G06F 11/0757 (2013.01)
(21) 출원번호 10-2017-0098148
(22) 출원일자 2017년08월02일
심사청구일자 2017년08월02일
(65) 공개번호 10-2019-0014388
(43) 공개일자 2019년02월12일
(56) 선행기술조사문헌
JP2015103052 A*
KR101534974 B1*
KR100648490 B1*
JP2011159136 A
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
현대오트론 주식회사
서울특별시 강남구 테헤란로113길 12(삼성동)
(72) 발명자
정해성
경기도 용인시 수지구 대지로 49 202동 102호(죽전동, 죽전퍼스트하임)
(74) 대리인
특허법인(유한) 대아

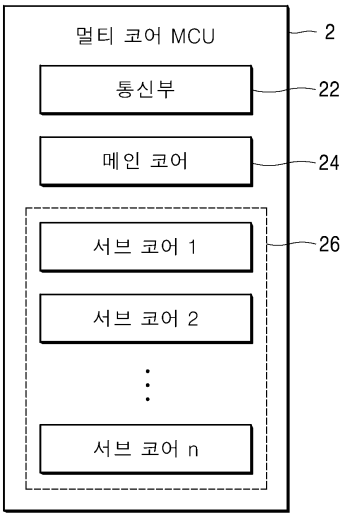
전체 청구항 수 : 총 10 항 심사관 : 김계준

(54) 발명의 명칭 **멀티 코어 MCU 및 그 동작 방법**

(57) 요 약

본 발명은 멀티 코어 MCU 및 그 동작 방법에 관한 것으로, 보다 상세하게는 메인 코어가 워치독으로부터 정상 동작 여부를 확인하기 위한 질문을 수신하는 단계, 상기 메인 코어가 상기 질문을 복수의 서브 코어에 전달하는 단계, 상기 복수의 서브 코어로부터 상기 질문에 대한 서브 코어 응답 데이터를 수신하는 단계, 상기 메인 코어가 상기 질문에 대한 메인 코어 응답 데이터를 생성하는 단계 및 상기 메인 코어가 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 기초로 상기 서브 코어의 동작 상태를 판단하는 단계를 포함한다. 전술한 바와 같은 본 발명에 의하면, 메인 코어를 통해 워치독 타이머의 동작 감시 대상이 아닌 서브 코어의 동작 상태를 판단함으로써, 소프트웨어 동작의 신뢰성을 높일 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

대 표 도 - 도2



명 세 서

청구범위

청구항 1

메인 코어가 워치독 타이머로부터 정상 동작 여부를 확인하기 위한 질문을 수신하는 단계;

상기 메인 코어가 상기 질문을 복수의 서브 코어에 전달하는 단계;

상기 복수의 서브 코어가 상기 질문에 대한 서브 코어 응답 데이터를 생성하는 단계;

상기 서브 코어가 상기 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시키는 단계;

상기 메인 코어가 복수의 서브 코어로부터 상기 서브 코어 응답 데이터를 수신하는 단계;

상기 메인 코어가 상기 질문에 대한 메인 코어 응답 데이터를 생성하는 단계; 및

상기 메인 코어가 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 기초로 논리 연산을 수행하여 상기 서브 코어의 동작 상태를 판단하는 단계를 포함하는

멀티 코어 MCU의 동작 방법.

청구항 2

삭제

청구항 3

제1항에 있어서,

상기 메인 코어가 상기 서브 코어의 동작 상태를 판단하는 단계는

상기 메인 코어가 상기 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 상기 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행하는 단계; 및

상기 메인 코어가 상기 논리 연산의 결과에 기초하여 상기 서브 코어의 동작 상태를 판단하는 단계를 포함하는

멀티 코어 MCU의 동작 방법.

청구항 4

제3항에 있어서,

상기 논리 연산은

배타적 논리합 연산인

멀티 코어 MCU의 동작 방법.

청구항 5

제3항에 있어서,

상기 논리 연산 결과에 기초하여 상기 메인 코어가 상기 서브 코어의 동작 상태를 판단하는 단계는

상기 논리 연산 결과가 제1 비트값일 경우 상기 복수의 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단하는 단계; 및

상기 논리 연산 결과가 제2 비트값일 경우 상기 복수의 서브 코어가 정상인 것으로 판단하는 단계를 포함하는

멀티 코어 MCU의 동작 방법.

청구항 6

제1항에 있어서,

상기 메인 코어가 상기 서브 코어의 동작 상태를 판단한 결과 상기 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단되면 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 비교하여 이상이 발생한 서브 코어를 검출하는 단계를 더 포함하는

멀티 코어 MCU의 동작 방법.

청구항 7

워치독 타이머로부터 정상 동작 여부를 확인하기 위한 질문을 수신하여 메인 코어에 전달하는 통신부;

상기 질문을 복수의 서브 코어에 전달하는 메인 코어; 및

상기 질문에 대한 서브 코어 응답 데이터를 생성하고, 상기 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시키는 서브 코어를 포함하고,

상기 메인 코어는 상기 복수의 서브 코어로부터 상기 서브 코어 응답 데이터를 수신하며, 상기 질문에 대한 메인 코어 응답 데이터를 생성하고, 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 기초로 논리 연산을 수행하여 상기 서브 코어의 동작 상태를 판단하는

멀티 코어 MCU.

청구항 8

삭제

청구항 9

제7항에 있어서,

상기 메인 코어는

상기 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 상기 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행하고, 상기 논리 연산의 결과에 기초하여 상기 서브 코어의 동작 상태를 판단하는

멀티 코어 MCU.

청구항 10

제9항에 있어서,

상기 논리 연산은

배타적 논리합 연산인

멀티 코어 MCU.

청구항 11

제9항에 있어서,

상기 메인 코어는

상기 논리 연산 결과가 제1 비트값일 경우 상기 복수의 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단하고, 상기 논리 연산 결과가 제2 비트값일 경우 상기 복수의 서브 코어가 정상인 것으로 판단하는

멀티 코어 MCU.

청구항 12

제7항에 있어서,

상기 메인 코어는

상기 서브 코어의 동작 상태를 판단한 결과 상기 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단되면 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 비교하여 이상이 발생한 서브 코어를 검출하는

멀티 코어 MCU.

발명의 설명

기 술 분 야

[0001] 본 발명은 멀티 코어 MCU 및 그 동작 방법에 관한 것으로, 상세하게는 워치독 타이머에 의해 모니터링 되고 있는 메인 코어를 통해 서브 코어의 동작 상태를 판단하는 멀티 코어 MCU 및 그 동작 방법에 관한 것이다.

배 경 기 술

[0003] 임베디드 시스템은 제어를 수행하기 위한 MCU(Micro Controller Unit) 및 MCU의 동작을 감시하는 워치독 타이머(Watchdog Timer, WDT)를 포함하는 것이 일반적이다. 워치독 타이머는 컴퓨터 또는 임베디드 시스템의 오작동을 탐지하고 복구하기 위해 쓰이는 전자 타이머이다.

[0004] 정상 작동 중인 시스템은 워치독 타이머의 에러 카운트 증가로 인한 타임 아웃이 발생하는 것을 막기 위해 미리 설정된 주기에 따라 워치독 타이머를 리셋시킨다. 즉, 의도치 않은 오류로 인해 시스템이 비정상적으로 동작할 경우, 워치독 타이머의 카운트는 리셋되지 않고 미리 설정된 카운트에 도달하여 타임 아웃이 발생하게 된다. 이때 워치독 타이머는 시스템의 오작동이 발생한 것으로 판단하여 해당 시스템을 정지시키거나 리셋(Reset) 시킬 수 있다.

[0005] 도 1은 질문/응답 방식을 사용하는 기존의 워치독 타이머의 감시 동작 과정을 나타낸 것이다.

[0006] 도 1을 참조하면, 질문/응답 방식을 사용하는 기존의 워치독 타이머(12)는 에러 카운트를 초기화한 후(101), 질문(Query)을 순차적으로 생성한다(102). 이때 질문은 질문에 대한 응답과 서로 대응되며, 워치독 타이머(12)는 질문에 대한 MCU(10)의 응답의 정답 여부를 판단하여 MCU의 정상 동작 여부를 확인할 수 있다.

[0007] 워치독 타이머(12)는 생성한 질문을 MCU(10)에게 전송한다(103). MCU(10)는 워치독 타이머(12)로부터 제1 질문을 수신한뒤(104), 제1 질문에 대한 제1 응답을 생성한다(105). 이와 같은 제1 응답 생성은 MCU에 미리 포함된 질문/응답 시스템을 통해 이루어질 수 있다. MCU(10)는 생성한 제1 응답을 워치독 타이머(12)에게 전송한다(106).

[0008] MCU(10)로부터 제1 응답을 수신(107)한 워치독 타이머(12)는 제1 응답의 정답 여부를 확인한다(108). 확인(108) 결과, 제1 응답이 정답인 경우 워치독 타이머(12)는 다음 질문을 위한 제2 질문을 생성한다(110). 확인(108) 결과, 만약 제1 응답이 정답이 아닌 경우 워치독 타이머(12)의 에러 카운트는 증가한다(109). 이처럼 질문/응답 방식을 사용하는 기존의 워치독 타이머(12)는 질문에 대한 응답의 정답 여부를 확인하여 에러 카운트의 증가 여부를 결정한다.

[0009] 최근 높은 성능을 가지는 임베디드 시스템에 대한 요구에 따라 두 개 이상의 독립 코어를 포함하는 멀티 코어

MCU의 개발 및 사용이 증가하고 있다. 이와 같은 멀티 코어 MCU의 통신 채널은 통신 채널과 연결된 하나의 코어에 의해 제어되므로, 멀티 코어 MCU에 포함된 코어 중 워치독 타이머와 연결된 하나의 코어만이 워치독 타이머에 의해 감시된다. 즉, 워치독 타이머는 통신 채널과 연결된 하나의 코어와 질문/응답을 주고 받으며 연결되지 않은 나머지 코어는 와치독 타이머에 의한 감시의 대상에서 제외된다.

[0010] 보통 멀티 코어 MCU는 각 코어에 소프트웨어를 파티셔닝(partitioning)하여 사용하므로, 어느 하나의 코어의 동작에 문제가 발생하면 전체 소프트웨어의 동작에 문제가 생긴다. 즉, 멀티 코어 MCU는 높은 연산 처리 능력을 가질 수 있으나, 워치독 타이머가 모든 코어의 동작을 감시하지 못하므로 소프트웨어 동작의 신뢰성이 떨어지는 문제가 있다.

발명의 내용

해결하려는 과제

[0012] 본 발명은 메인 코어를 통해 워치독 타이머의 동작 감시 대상이 아닌 서브 코어의 동작 상태를 판단함으로써, 소프트웨어 동작의 신뢰성을 높일 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공하는 것을 목적으로 한다.

[0013] 또한 본 발명은 워치독 타이머의 질문에 대한 메인 코어의 응답 및 서브 코어의 응답에 대한 논리 연산을 수행함으로써, 서브 코어의 이상 발생 여부를 간단히 검출할 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공하는 것을 목적으로 한다.

[0014] 본 발명의 목적들은 이상에서 언급한 목적으로 제한되지 않으며, 언급되지 않은 본 발명의 다른 목적 및 장점들은 하기의 설명에 의해서 이해될 수 있고, 본 발명의 실시예에 의해 보다 분명하게 이해될 것이다. 또한, 본 발명의 목적 및 장점들은 특허 청구 범위에 나타낸 수단 및 그 조합에 의해 실현될 수 있음을 쉽게 알 수 있을 것이다.

과제의 해결 수단

[0016] 이러한 목적을 달성하기 위한 본 발명의 일 측면은, 메인 코어가 워치독 타이머로부터 정상 동작 여부를 확인하기 위한 질문을 수신하는 단계, 상기 메인 코어가 상기 질문을 복수의 서브 코어에 전달하는 단계, 상기 복수의 서브 코어로부터 상기 질문에 대한 서브 코어 응답 데이터를 수신하는 단계, 상기 메인 코어가 상기 질문에 대한 메인 코어 응답 데이터를 생성하는 단계 및 상기 메인 코어가 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 기초로 상기 서브 코어의 동작 상태를 판단하는 단계를 포함하는 멀티 코어 MCU의 동작 방법을 제공할 수 있다.

[0017] 상기 멀티 코어 MCU의 동작 방법은 상기 복수의 서브 코어가 상기 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시키는 단계 및 상기 복수의 서브 코어가 상기 서브 코어 응답 데이터를 상기 메인 코어로 송신하는 단계를 더 포함할 수 있다.

[0018] 상기 서브 코어의 동작 상태를 판단하는 단계는 상기 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 상기 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행하는 단계 및 상기 논리 연산의 결과에 기초하여 상기 서브 코어의 동작 상태를 판단하는 단계를 포함할 수 있다.

[0019] 상기 논리 연산은 배타적 논리합 연산일 수 있다.

[0020] 상기 논리 연산 결과에 기초하여 상기 서브 코어의 동작 상태를 판단하는 단계는 상기 논리 연산 결과가 제1 비트값일 경우 상기 복수의 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단하는 단계 및 상기 논리 연산 결과가 제2 비트값일 경우 상기 복수의 서브 코어가 정상인 것으로 판단하는 단계를 포함할 수 있다.

[0021] 상기 멀티 코어 MCU의 동작 방법은 상기 서브 코어의 동작 상태를 판단한 결과 상기 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단되면 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 비교하여 이상이 발생한 서브 코어를 검출하는 단계를 더 포함할 수 있다.

[0022] 한편, 이러한 목적을 달성하기 위한 본 발명의 다른 측면은, 워치독 타이머로부터 정상 동작 여부를 확인하기 위한 질문을 수신하여 메인 코어에 전달하는 통신부 및 상기 질문을 복수의 서브 코어에 전달하고 상기 복수의 서브 코어로부터 상기 질문에 대한 서브 코어 응답 데이터를 수신하며, 상기 질문에 대한 메인 코어 응답 데이터를 생성하고 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 기초로 상기 서브 코어의 동작 상

태를 판단하는 메인 코어를 포함하는 멀티 코어 MCU를 제공할 수 있다.

[0023] 상기 멀티 코어 MCU는 상기 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시키고, 상기 서브 코어 응답 데이터를 상기 메인 코어로 송신하는 서브 코어를 더 포함할 수 있다.

[0024] 상기 메인 코어는 상기 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 상기 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행하고, 상기 논리 연산의 결과에 기초하여 상기 서브 코어의 동작 상태를 판단할 수 있다.

[0025] 상기 논리 연산은 배타적 논리합 연산일 수 있다.

[0026] 상기 메인 코어는 상기 논리 연산 결과가 제1 비트값일 경우 상기 복수의 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단하고, 상기 논리 연산 결과가 제2 비트값일 경우 상기 복수의 서브 코어가 정상인 것으로 판단할 수 있다.

[0027] 상기 메인 코어는 상기 서브 코어의 동작 상태를 판단한 결과 상기 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단되면 상기 메인 코어 응답 데이터와 상기 서브 코어 응답 데이터를 비교하여 이상이 발생한 서브 코어를 검출할 수 있다.

발명의 효과

[0029] 전술한 바와 같은 본 발명에 의하면, 메인 코어를 통해 워치독 타이머의 동작 감시 대상이 아닌 서브 코어의 동작 상태를 판단함으로써, 소프트웨어 동작의 신뢰성을 높일 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

[0030] 또한 본 발명에 의하면, 워치독 타이머의 질문에 대한 메인 코어의 응답 및 서브 코어의 응답에 대한 논리 연산을 수행함으로써, 서브 코어의 이상 발생 여부를 간단히 검출할 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

도면의 간단한 설명

[0032] 도 1은 질문/응답 방식을 사용하는 기존의 워치독 타이머의 감시 동작 과정을 나타낸 것이다.

도 2는 본 발명의 일 실시예에 따른 멀티 코어 MCU의 구성을 개략적으로 나타낸 개념도이다.

도 3은 본 발명의 일 실시예에 따른 멀티 코어 MCU의 동작 방법의 흐름도를 나타낸 것이다.

도 4는 본 발명의 일 실시예에 따른 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산을 나타낸 것이다.

도 5는 본 발명의 일 실시예에 따른 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과의 진리표를 나타낸 것이다.

도 6은 복수의 서브코어가 정상 동작할 경우 메인 코어 응답 데이터(M) 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과를 나타낸 것이다.

도 7은 복수의 서브 코어 중 어느 하나에 이상이 발생한 경우 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과를 나타낸 것이다.

발명을 실시하기 위한 구체적인 내용

[0033] 전술한 목적, 특징 및 장점은 첨부된 도면을 참조하여 상세하게 후술되며, 이에 따라 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명의 기술적 사상을 용이하게 실시할 수 있을 것이다. 본 발명을 설명함에 있어서 본 발명과 관련된 공지 기술에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 상세한 설명을 생략한다. 이하, 첨부된 도면을 참조하여 본 발명에 따른 바람직한 실시예를 상세히 설명하기로 한다. 도면에서 동일한 참조부호는 동일 또는 유사한 구성요소를 가리키는 것으로 사용된다.

[0034] 도 2는 본 발명의 일 실시예에 따른 멀티 코어 MCU의 구성을 개략적으로 나타낸 개념도이다.

[0035] 도 2를 참조하면, 본 발명의 일 실시예에 따른 멀티 코어 MCU(2)는 통신부(22), 메인 코어(24) 및 서브 코어(26)를 포함할 수 있다.

[0036] 통신부(22)는 워치독 타이머로부터 정상 동작 여부를 확인하기 위한 질문을 수신하여 메인 코어(24)에 전달한다.

[0037] 본 발명의 일 실시예에서, 워치독 타이머는 메인 코어(24)의 동작 상태를 감시할 수 있다. 즉, 워치독 타이머는 질문/응답 방식을 통해 본 발명의 멀티 코어 MCU(2)에 포함된 메인 코어(24)의 동작 상태를 감시할 수 있다. 워치독 타이머와 메인 코어(24)의 질문/응답 송수신은 통신부(22)를 통해 이루어질 수 있으며, 통신부(22)는 메인 코어(24)에 의해 제어될 수 있다.

[0038] 메인 코어(24)는 워치독으로부터 수신한 질문에 대한 메인 코어 응답 데이터를 생성한다.

[0039] 본 발명의 일 실시예에서, 메인 코어(24)는 미리 설정된 알고리즘을 사용하여 워치독 타이머의 질문에 대한 메인 코어 응답 데이터를 생성할 수 있다. 이때 미리 설정된 알고리즘은 후술할 서브 코어(26)의 서브 코어 응답 데이터의 생성에 동일하게 사용될 수 있다.

[0040] 본 발명의 일 실시예에서, 메인 코어(24)는 워치독 타이머로부터 수신한 질문에 대한 메인 코어 응답 데이터를 생성하여 다시 워치독 타이머에게 전송할 수 있다. 즉, 워치독 타이머는 질문에 대한 메인 코어 응답 데이터의 정답 여부를 판단하여 메인 코어(24)의 정상 동작 여부를 확인할 수 있다.

[0041] 본 발명의 일 실시예에서, 메인 코어(24)는 서브 코어(26)의 동작 상태와 무관하게 워치독 타이머의 동작 감시 대상이 될 수 있다.

[0042] 메인 코어(24)는 워치독 타이머로부터 수신한 질문을 복수의 서브 코어(26)에 전달한다.

[0043] 본 발명의 일 실시예에서, 메인 코어(24)는 워치독 타이머로부터 수신한 질문을 복수의 서브 코어(26) 각각에 전달할 수 있으며, 각 서브 코어(26)는 메인 코어(24)로부터 전달받은 워치독 타이머의 질문에 대한 서브 코어 응답 데이터를 생성할 수 있다.

[0044] 메인 코어(24)는 복수의 서브 코어(26)로부터 서브 코어 응답 데이터를 수신한다.

[0045] 본 발명의 일 실시예에서, 메인 코어(24)는 워치독 타이머의 감시 대상이 되지 못하는 서브 코어(26)에게 메인 코어(24)가 전송받은 워치독 타이머의 질문을 전달할 수 있다. 서브 코어(26)가 워치독 타이머의 질문에 대한 서브 코어 응답 데이터를 생성하면, 메인 코어(24)는 서브 코어(26)로부터 서브 코어 응답 데이터를 수신할 수 있다.

[0046] 메인 코어(24)는 메인 코어 응답 데이터와 서브 코어 응답 데이터를 기초로 서브 코어(26)의 동작 상태를 판단한다.

[0047] 본 발명의 일 실시예에서, 메인 코어(24)는 메인 코어 응답 데이터와 복수의 서브 코어 응답 데이터를 기초로 서브 코어(26)의 동작 상태를 판단할 수 있다.

[0048] 이처럼 본 발명의 멀티 코어 MCU는 워치독 타이머에 의해 모니터링 되고 있는 메인 코어를 통해 서브 코어의 동작 상태를 판단함으로써, 워치독 타이머의 동작 감시 대상이 아닌 복수의 서브 코어에 대한 모니터링을 수행할 수 있다.

[0049] 본 발명의 일 실시예에서, 메인 코어(24)는 서브 코어(26)의 동작 상태를 판단하기 위해 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어(26)로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행할 수 있다. 여기서 메인 코어(24)가 수행하는 논리 연산은 배타적 논리합 연산(Exclusive OR operation)으로 이루어질 수 있다.

[0050] 본 발명의 일 실시예에서, 메인 코어(24)는 서브 코어 응답 데이터와 메인 코어 응답 데이터를 사용하여 복수의 서브 코어(26) 중 어느 하나에 이상이 발생했는지 여부를 판단할 수 있다.

[0051] 메인 코어(24)는 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어(26)로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대한 논리 연산을 수행할 수 있다.

[0052] 논리 연산의 결과가 제1 비트값일 경우, 복수의 서브 코어(26) 중 어느 하나에 이상이 발생한 것으로 판단할 수 있다. 반대로 논리 연산 결과가 제2 비트값일 경우, 메인 코어(24)는 복수의 서브 코어(26)가 정상인 것으로 판단할 수 있다.

[0053] 이때 제1 비트값 및 제2 비트값은 각각 참(1) 또는 거짓(0)의 서로 다른 비트값으로 이루어질 수 있다.

[0055] *예를 들어, 제1 비트값이 참(1)으로 설정될 경우 제2 비트값은 거짓(0)으로 설정될 수 있으며, 반대로 제1 비트값이 거짓(0)으로 설정될 경우 제2 비트값은 참(1)으로 설정될 수 있다.

[0056] 본 발명의 일 실시예에서, 제1 비트값 및 제2 비트값은 메인 코어(24)가 사용하는 논리 연산식, 서브 코어(26)의 개수, 워치독 타이머의 질문 및 메인 코어(24)와 서브 코어(26)가 응답을 생성하기 위해 사용하는 알고리즘에 따라 달라질 수 있다.

[0057] 본 발명의 일 실시예에서, 메인 코어(24)가 수행하는 논리 연산은 배타적 논리합 연산으로 이루어질 수 있다.

[0058] 배타적 논리 회로 게이트는 두 입력 중 하나가 참(1)이고 다른 하나가 거짓(0)인 경우에만 참(1)을 출력하고, 두 입력이 모두 참(1)이거나 모두 거짓(0)인 경우, 거짓(0)을 출력한다.

[0059] 이와 같은 배타적 논리합 연산을 사용하여 메인 코어(24)가 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산을 수행하는 과정은 도 4 및 도 5를 통해 상세히 설명한다.

[0060] 메인 코어(24)는 서브 코어(26)의 동작 상태를 판단한 결과 서브 코어(26) 중 어느 하나에 이상이 발생한 것으로 판단되면, 메인 코어 응답 데이터와 서브 코어 응답 데이터를 비교하여 이상이 발생한 서브 코어(26)를 검출한다.

[0061] 이처럼 본 발명은 워치독 타이머의 질문에 대한 메인 코어의 응답 및 서브 코어의 응답에 대한 논리 연산을 수행함으로써, 서브 코어의 이상 발생 여부를 간단히 검출할 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

[0062] 서브 코어(26)는 메인 코어(24)로부터 전달받은 워치독 타이머 질문을 기초로 서브 코어 응답 데이터를 생성한다.

[0063] 전술한 것과 같이, 메인 코어(24)는 워치독 타이머로부터 수신한 질문을 복수의 서브 코어(26) 각각에 전달할 수 있으며, 각 서브 코어(26)는 메인 코어(24)로부터 전달 받은 워치독 타이머의 질문에 대한 서브 코어 응답 데이터를 생성할 수 있다.

[0064] 본 발명의 일 실시예에서, 서브 코어(26)는 미리 설정된 알고리즘을 사용하여 서브 코어 응답 데이터를 생성할 수 있다. 이때 서브 코어(26)에서 사용되는 미리 설정된 알고리즘은 메인 코어(24)가 메인 코어 응답 데이터를 생성하는 과정에서 동일하게 사용될 수 있다.

[0065] 서브 코어(26)는 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시킨다. 즉, 서브 코어(26)는 메인 코어(24)로부터 전달받은 워치독 타이머 질문을 기초로 서브 코어 응답 데이터를 생성하고, 생성된 서브 코어 응답 데이터를 메인 코어(24)에 전송하기 전에 미리 설정된 비트 수만큼 시프트 시킨다.

[0066] 본 발명의 일 실시예에서, 서브 코어(26)는 미리 설정된 시프트 연산자에 따라 서브 코어 응답 데이터를 시프트 시킬 수 있다. 이때 미리 설정된 시프트 연산자는 서브 코어 응답 데이터를 미리 설정된 방향으로 미리 설정된 비트 수만큼 시프트 시킬 수 있다.

[0067] 본 발명의 일 실시예에서, 복수의 서브 코어(26)가 서브 코어 응답 데이터를 시프트하기 위해 사용하는 미리 설정된 시프트 연산자는 각 서브 코어(26)에 따라 다르게 설정될 수 있다.

[0068] 도 3은 본 발명의 일 실시예에 따른 멀티 코어 MCU의 동작 방법의 흐름도를 나타낸 것이다.

[0069] 도 3을 참조하면, 먼저 메인 코어(24)는 워치독 타이머로부터 질문을 수신한다(301). 메인 코어(24)는 워치독 타이머로부터 수신한 질문을 복수의 서브 코어(26)에 전달한다(302).

[0070] 메인 코어(24)로부터 질문을 전달받은 서브 코어(26)는 전달받은 질문에 대한 서브 코어 응답 데이터를 생성한다(303).

[0071] 본 발명의 일 실시예에서, 서브 코어(26)는 미리 설정된 알고리즘을 사용하여 서브 코어 응답 데이터를 생성할 수 있다. 이때 서브 코어(26)에서 사용되는 미리 설정된 알고리즘은 메인 코어(24)가 메인 코어 응답 데이터를 생성하는 과정에서 사용될 수 있다.

[0072] 즉, 메인 코어(24)는 워치독 타이머로부터 수신한 질문을 서브 코어(26)에 전달하고, 메인 코어(24) 및 서브 코어(26)는 각각 동일한 질문에 대해 미리 설정된 동일한 알고리즘을 사용하여 메인 코어 응답 데이터 및 서브 코어 응답 데이터를 생성할 수 있다.

[0073] 다음으로, 서브 코어(26)는 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트 시킨다(304).

[0074] 본 발명의 일 실시예에서, 서브 코어(26)는 미리 설정된 시프트 연산자에 따라 서브 코어 응답 데이터를 시프트 시킬 수 있다.

[0075] 본 발명의 일 실시예에서, 복수의 서브 코어(26)가 서브 코어 응답 데이터를 시프트하기 위해 사용하는 미리 설정된 시프트 연산자는 각 서브 코어(26)에 따라 다르게 설정될 수 있다.

[0076] 각 서브 코어(26)가 각 서브 코어 응답 데이터에 대해 미리 설정된 비트 수만큼 시프트를 수행하는 과정은 도 6을 통해 후술한다.

[0077] 이후, 메인 코어(24)는 서브 코어(26)로부터 서브 코어 응답 데이터를 수신한다(305).

[0078] 본 발명의 일 실시예에서, 메인 코어(24)는 복수의 서브 코어(26)로부터 서브 코어 응답 데이터를 수신할 수 있다. 메인 코어(24)가 수신하는 각 서브 코어 응답 데이터는 각 서브 코어(26)에서 미리 설정된 비트 수만큼 시프트가 수행된 서브 코어 데이터로 이루어질 수 있다.

[0079] 다음으로, 메인 코어(24)는 워치독 타이머로부터 수신한 질문에 대한 메인 코어 응답 데이터를 생성한다(306).

[0080] 마지막으로, 메인 코어(24)는 메인 코어 응답 데이터와 서브 코어 응답 데이터를 기초로 서브 코어(26)의 동작 상태를 판단한다(307).

[0081] 본 발명의 일 실시예에서, 메인 코어(24)는 메인 코어 응답 데이터와 복수의 서브 코어 응답 데이터를 기초로 서브 코어(26)의 동작 상태를 판단할 수 있다.

[0082] 즉, 발명의 멀티 코어 MCU는 워치독 타이머에 의해 모니터링 되고 있는 메인 코어를 통해 서브 코어의 동작 상태를 판단함으로써, 워치독 타이머의 동작 감시 대상이 아닌 복수의 서브 코어에 대한 모니터링을 수행할 수 있다.

[0083] 따라서, 본 발명에 의하면 메인 코어를 통해 복수의 서브 코어의 동작 상태를 감시함으로써 멀티 코어 MCU에서 구동되는 소프트웨어의 동작 신뢰성을 높일 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

[0084] 본 발명의 일 실시예에서, 메인 코어(24)는 서브 코어(26)의 동작 상태를 판단하기 위해 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어(26)로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산을 수행할 수 있다. 여기서 메인 코어(24)가 수행하는 논리 연산은 배타적 논리합 연산으로 이루어질 수 있다.

[0085] 본 발명의 일 실시예에서, 메인 코어(24)는 서브 코어 응답 데이터와 메인 코어 응답 데이터를 사용하여 복수의 서브 코어(26) 중 어느 하나에 이상이 발생했는지 여부를 판단할 수 있다.

[0086] 이처럼 본 발명은 워치독 타이머의 질문에 대한 메인 코어의 응답 및 서브 코어의 응답에 대한 논리 연산을 수행함으로써, 서브 코어의 이상 발생 여부를 간단히 검출할 수 있는 멀티 코어 MCU 및 그 동작 방법을 제공할 수 있는 장점이 있다.

[0087] 메인 코어(24)는 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어(26)로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대한 논리 연산 결과가 제1 비트값일 경우 복수의 서브 코어(26) 중 어느 하나에 이상이 발생한 것으로 판단할 수 있다. 반대로, 메인 코어(24)는 논리 연산 결과가 제2 비트값일 경우 복수의 서브 코어(26)가 정상인 것으로 판단할 수 있다.

[0088] 본 발명의 일 실시예에서, 제1 비트값 및 제2 비트값은 메인 코어(24)가 사용하는 논리 연산식, 서브 코어(26)의 개수, 워치독 타이머의 질문 및 메인 코어(24)와 복수의 서브 코어(26)가 응답을 생성하기 위해 사용하는 미리 설정된 알고리즘에 따라 달라질 수 있다.

[0089] 이하에서는 도 4 및 도 5를 통해, 메인 코어가 배타적 논리합 연산을 통해 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산을 수행하는 과정을 상세히 설명한다.

[0090] 도 4는 본 발명의 일 실시예에 따른 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산을 나타낸 것이다.

[0091] 본 발명의 일 실시예에서, 메인 코어는 메인 코어 응답 데이터 및 서브 코어 응답 데이터에 대한 배타적 논리합

연산을 수행할 수 있다.

[0092] 도 4를 참조하면, 메인 코어는 3개의 배타적 논리합 게이트(41, 42, 43)를 사용하여 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 내지 제3 서브 코어 응답 데이터(S3)에 대한 배타적 논리합 연산을 수행할 수 있다.

[0093] 제1 배타적 논리합 게이트(41)는 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1)에 대한 논리 연산을 수행하며, 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 중 하나가 참(1)이고 다른 하나가 거짓(0)인 경우에만 참(1)을 출력하고, 두 데이터가 모두 참(1)이거나 모두 거짓(0)인 경우 거짓(0)을 출력한다.

[0095] *마찬가지로, 제2 배타적 논리합 게이트(42)는 제2 서브 코어 응답 데이터(S2) 및 제3 서브 코어 응답 데이터(S3)에 대한 논리 연산을 수행하며, 제2 서브 코어 응답 데이터(S2) 및 제3 서브 코어 응답 데이터(S3) 중 어느 하나가 참(1)이고 다른 하나가 거짓(0)인 경우에만 참(1)을 출력하고, 두 데이터가 모두 참(1)이거나 모두 거짓(0)인 경우 거짓(0)을 출력한다.

[0096] 다시 도 4를 참조하면, 제3 배타적 논리합 게이트(43)는 제1 배타적 논리합 게이트(41) 및 제2 배타적 논리합 게이트(42)의 출력에 대해 논리 연산을 수행하며, 제3 배타적 논리합 게이트(43)의 출력은 논리식 R=(MS1)(S2S3)의 결과값으로 표현될 수 있다(45).

[0097] 도 5는 본 발명의 일 실시예에 따른 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과의 진리표를 나타낸 것이다.

[0098] 도 5를 참조하면, 제1 배타적 논리합 게이트(41)가 수행한 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1)의 배타적 논리합 연산 결과(52), 제2 배타적 논리합 게이트(42)가 수행한 제2 서브 코어 응답 데이터(S2)와 제3 서브 코어 응답 데이터(S3)의 배타적 논리합 연산 결과(53) 및 제3 배타적 논리합 게이트(43)가 수행한 제1 배타적 논리합 게이트 출력 및 제2 배타적 논리합 게이트 출력의 배타적 논리합 연산 결과(56)가 나타나있다.

[0099] 제3 배타적 논리합 게이트(43)가 수행한 배타적 논리합 연산 결과를 참조하면, 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 내지 제3 서브 코어 응답 데이터(S3)에 포함된 참(1)의 개수가 홀수인 경우 제3 배타적 논리합 게이트(43)의 출력은 참(1)이 된다.

[0100] 반대로, 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 내지 제3 서브 코어 응답 데이터(S3)에 포함된 참(1)의 개수가 0개 이거나 짝수인 경우 제3 배타적 논리합 게이트(43)의 출력은 거짓(0)이 된다.

[0101] 즉, 메인 코어는 메인 코어 응답 데이터(M) 및 복수의 서브 코어 응답 데이터에 대한 배타적 논리합 연산을 수행하여 코어 응답 데이터 및 복수의 서브 코어 응답 데이터 중 참(1)의 비트 값을 가지는 데이터의 수가 홀수인지 또는 짝수인지 여부를 판단할 수 있다.

[0102] 이하에서는 도 6 및 도 7을 통해 메인 코어가 메인 코어 응답 데이터 및 복수의 서브 코어 데이터를 사용하여 배타적 논리합 연산을 통해 복수의 서브 코어의 동작 상태를 감시하는 과정을 상세히 설명한다.

[0103] 도 6은 복수의 서브코어가 정상 동작할 경우 메인 코어 응답 데이터(M) 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과를 나타낸 것이다.

[0104] 도 6을 참조하면, 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 내지 제3 서브 코어 응답 데이터(S3)는 각각 32비트 데이터로 이루어질 수 있다.

[0105] 즉, 메인 코어는 미리 설정된 알고리즘을 사용하여 워치독 타이머의 질문에 대해 32비트의 데이터로 구성되는 메인 코어 응답 데이터(M)를 생성할 수 있다. 도 6에 도시된 것과 같이, 메인 코어 응답 데이터(M)는 1111 1111 0000 1111 1111 0000 0000 0000의 비트 행으로 구성될 수 있다.

[0106] 서브 코어는 메인 코어로부터 전달받은 워치독 타이머의 질문에 대한 서브 코어 응답 데이터를 생성할 수 있다.

[0107] 서브 코어가 서브 코어 응답 데이터를 생성하기 위해 사용하는 알고리즘은 메인 코어가 메인 코어 응답 데이터(M)를 생성하기 위해 사용하는 알고리즘과 동일하다.

[0108] 전술한 것과 같이, 서브 코어는 서브 코어 응답 데이터를 미리 설정된 비트 수만큼 시프트할 수 있다.

[0109] 예를 들어, 메인 코어 응답 데이터(M)가 1111 1111 0000 1111 1111 0000 0000 0000의 비트 행으로 이루어질 경

우, 제1 서브 코어가 처음 생성한 제1 서브 코어 응답 데이터는 1111 1111 0000 1111 1111 0000 0000 0000의 비트 행으로 이루어질 수 있다.

[0110] 이때 제1 서브 코어는 제1 서브 코어 응답 데이터를 좌측으로 8비트만큼 시프트 시킬 수 있다. 제1 서브 코어의 데이터 시프트 수행으로 인해 제1 서브 코어 응답 데이터(S1)는 0000 1111 1111 0000 0000 0000 1111 1111의 비트 행으로 이루어질 수 있다.

[0111] 이와 유사하게, 제2 서브 코어는 제2 서브 코어 응답 데이터를 좌측으로 16비트 만큼 시프트 시킬 수 있으며, 제3 서브 코어는 제3 서브 코어 응답 데이터를 좌측으로 24비트 만큼 시프트 시킬 수 있다.

[0112] 다시 도 6을 참조하면, 제2 서브 코어 및 제3 서브 코어가 데이터 시프트를 수행한 결과, 제2 서브 코어 응답 데이터(S2)는 1111 0000 0000 0000 1111 1111 0000 1111의 비트 행으로 이루어질 수 있으며, 제3 서브 코어 응답 데이터(S3)는 0000 0000 1111 1111 0000 1111 1111 0000의 비트 행으로 이루어질 수 있다.

[0113] 메인 코어는 메인 코어 응답 데이터(M) 및 서브 코어 응답 데이터(S1, S2, S3)에 대한 배타적 논리합 연산을 수행할 수 있다.

[0114] 다시 도 6을 참조하면, 메인 코어 응답 데이터(M)의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터(S1, S2, S3)의 n번째 비트 열에 저장된 데이터에 대하여 논리 연산 수행 결과가 나타나있다.

[0115] 예를 들어, 모든 서브 코어가 정상 동작할 경우 메인 코어 응답 데이터(M) 및 제2 서브 코어 응답 데이터(S2)의 24번째 비트 열에 저장된 데이터는 거짓(0)이 될 수 있으며, 제1 서브 코어 응답 데이터(S1) 및 제3 서브 코어 응답 데이터(S3)의 24번째 비트 열에 저장된 데이터는 참(1)이 될 수 있다.

[0116] 이때 메인 코어 응답 데이터(M) 및 제1 서브 코어 응답 데이터(S1) 내지 제3 서브 코어 응답 데이터(S3)의 24번째 비트 열에 저장된 데이터에 대한 배타적 논리합 연산 결과는 거짓(0)이 될 수 있다.

[0117] 도 6에 도시된 것과 같이, 모든 서브 코어가 정상 동작할 경우 메인 코어 응답 데이터(M)의 모든 비트 열에 저장된 데이터 및 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 모든 비트 열에 저장된 데이터에 대한 배타적 논리합 연산 결과는 모두 거짓(0)이 될 수 있다.

[0118] 도 7은 복수의 서브 코어 중 어느 하나에 이상이 발생한 경우 메인 코어 응답 데이터 및 복수의 서브 코어 응답 데이터에 대한 논리 연산 결과를 나타낸 것이다.

[0119] 전술한 것과 같이, 메인 코어는 메인 코어 응답 데이터의 n번째 비트 열에 저장된 데이터 및 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 n번째 비트 열에 저장된 데이터에 대한 논리 연산 결과가 제1 비트값일 경우 복수의 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단하고, 반대로 논리 연산 결과가 제2 비트값일 경우 복수의 서브 코어가 정상인 것으로 판단할 수 있다.

[0120] 도 7을 참조하면, 이상이 발생한 제1 서브 코어의 제1 서브 코어 응답 데이터(S1)는 도 6의 제1 서브 코어 응답 데이터(S1)와 달리 21번째 내지 24번째 비트 열 데이터가 거짓(0)으로 이루어질 수 있다.

[0121] 이때 메인 코어 응답 데이터(M)의 21번째 내지 24번째 비트 열에 저장된 데이터 및 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터의 21번째 내지 24번째 비트 열에 저장된 데이터(7)에 대한 배타적 논리합 연산 결과는 참(1)이 될 수 있다.

[0122] 즉, 모든 서브 코어가 정상 동작할 경우 메인 코어가 수행한 메인 코어 응답 데이터(M) 및 서브 코어 응답 데이터에 대한 배타적 논리합 연산 결과는 모든 비트 열에 대해 거짓(0)이 될 수 있다.

[0123] 반면, 복수의 서브 코어 중 어느 하나에 이상이 발생한 경우 메인 코어가 수행한 메인 코어 응답 데이터(M) 및 서브 코어 응답 데이터에 대한 배타적 논리합 연산 결과는 일부 비트 열에 대해 참(1)이 될 수 있고, 나머지 비트 열에 대해 거짓(0)이 될 수 있다.

[0124] 본 발명의 일 실시예에서, 메인 코어가 사용하는 배타적 논리합 연산의 제1 비트값이 참(1), 제2 비트값이 거짓(0)으로 설정될 수 있다. 이때 메인 코어가 메인 코어 응답 데이터 및 서브 코어 응답 데이터의 모든 비트 열에 대해 논리 연산을 수행한 결과 모든 비트 열 중에서 제1 비트값에 해당하는 배타적 논리합 연산 결과를 가지는 비트 열이 검출될 경우, 메인 코어는 제1 서브 코어 내지 제3 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단할 수 있다.

[0125] 다시 도 7을 참조하면, 메인 코어가 사용하는 배타적 논리합 연산의 제1 비트값은 참(1)으로 설정되며, 메인 코어가 수행한 메인 코어 응답 데이터(M) 및 복수의 서브 코어로부터 수신된 서브 코어 응답 데이터(S1, S2, S3)에 대한 배타적 논리합 연산 결과의 일부 비트 열이 참(1)이다. 따라서 메인 코어는 제1 서브 코어 내지 제3 서브 코어 중 어느 하나에 이상이 발생한 것으로 판단할 수 있다.

[0126] 이처럼 본 발명의 멀티 코어 MCU는 메인 코어를 통해 메인 코어의 응답 및 복수의 서브 코어의 응답에 대한 논리 연산을 수행함으로써, 복수의 서브 코어 중 어느 하나의 서브 코어에 이상이 발생하였는지 여부를 신속하고 간단하게 판단할 수 있는 장점이 있다.

[0127] 전술한 본 발명은, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에게 있어 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 여러 가지 치환, 변형 및 변경이 가능하므로 전술한 실시예 및 첨부된 도면에 의해 한정되는 것이 아니다.

부호의 설명

[0129] 2: 멀티 코어 MCU

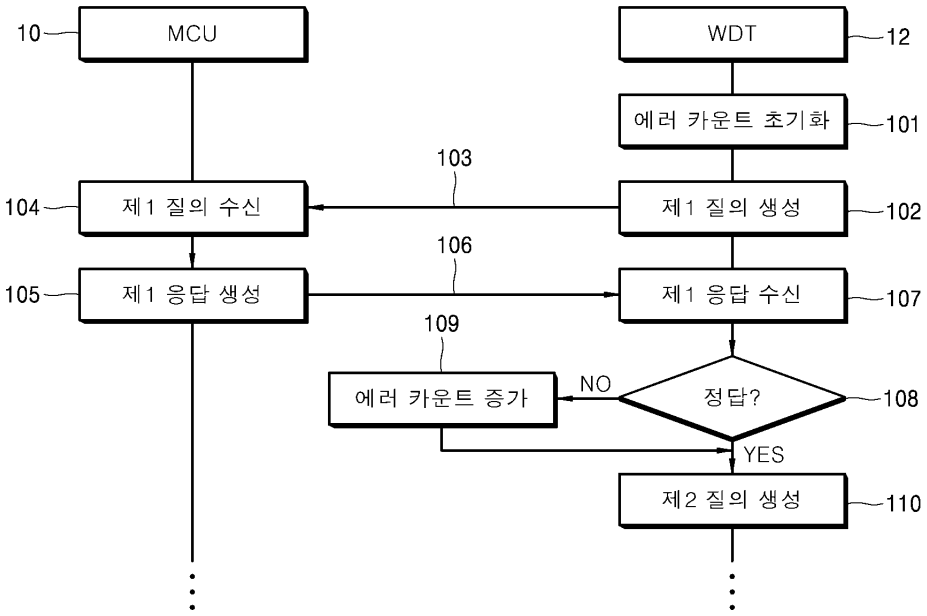
22: 통신부

24: 메인 코어

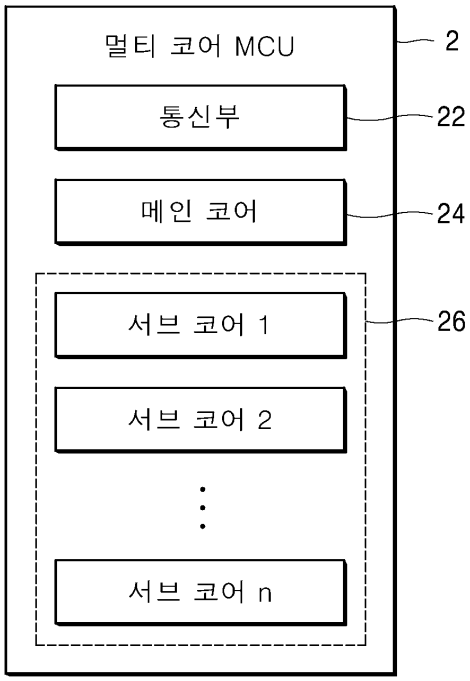
26: 서브 코어

도면

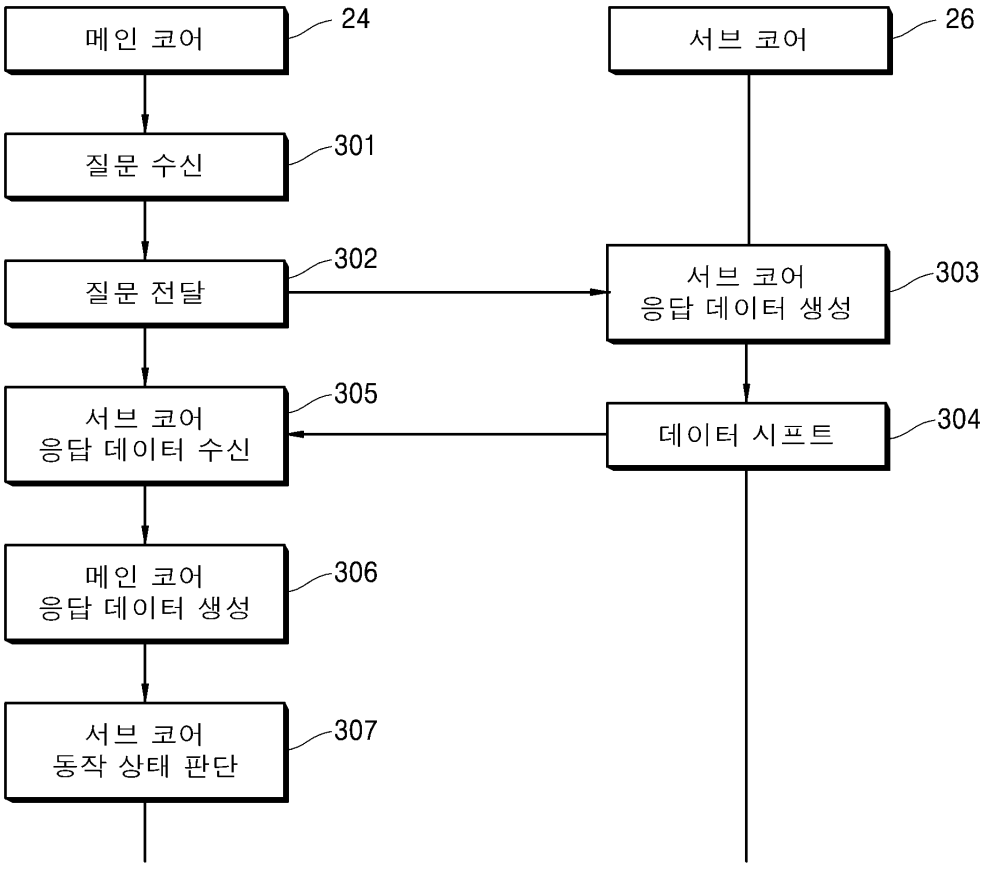
도면1



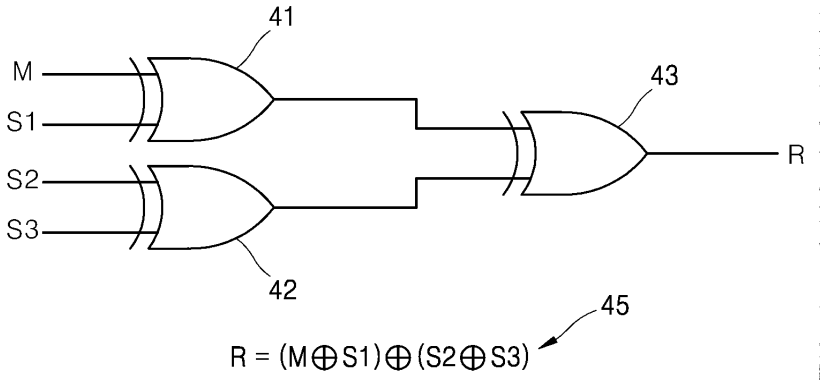
도면2


멀티 코어 MCU
2
통신부
22
메인 코어
24
서브 코어 1
26
서브 코어 2
서브 코어 n

도면3


메인 코어
24
서브 코어
26
질문 수신
301
질문 전달
302
서브 코어
응답 데이터 생성
303
서브 코어
응답 데이터 수신
305
데이터 시프트
304
메인 코어
응답 데이터 생성
306
서브 코어
동작 상태 판단
307

도면4



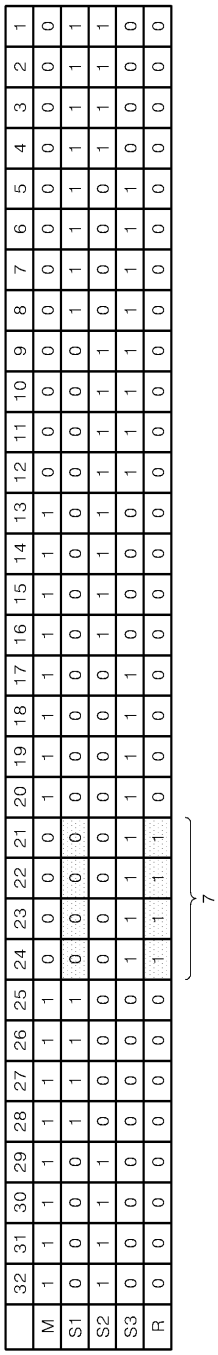
도면5

M	S1	S2	S3	M⊕S1 (52)	S2⊕S3 (54)	R (56)
0	0	0	0	0	0	0
0	0	0	1	0	1	1
0	0	1	0	0	1	1
0	0	1	1	0	0	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	0
0	1	1	1	1	0	1
1	0	0	0	1	0	1
1	0	0	1	1	1	0
1	0	1	0	1	1	0
1	0	1	1	1	0	1
1	1	0	0	0	0	0
1	1	0	1	0	1	1
1	1	1	0	0	1	1
1	1	1	1	0	0	0

도면6

	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
M	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
S1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
S2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1
S3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

도면7


32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
M 1 1 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
S1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0
S2 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 0
S3 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0
R 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0
7