



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2013-0118708
(43) 공개일자 2013년10월30일

(51) 국제특허분류(Int. Cl.)
H03K 19/173 (2006.01) G06F 11/07 (2006.01)
(21) 출원번호 10-2012-0059143
(22) 출원일자 2012년06월01일
심사청구일자 2012년06월01일
(30) 우선권주장
1020120041626 2012년04월20일 대한민국(KR)

(71) 출원인
조선대학교산학협력단
광주광역시 동구 서석동 375 조선대학교 내
(72) 발명자
이정아
서울특별시 마포구 상암동 휴먼시아아파트 203동 102호
하산베이그
파키스탄 75400 카라치 피.이.씨.에이치.에스, 블
록 2 하우스 # 3-제이
(74) 대리인
특허법인아이엠

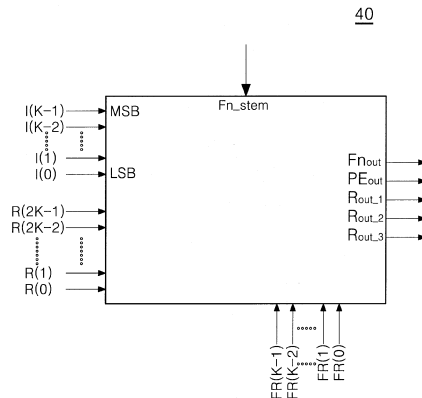
전체 청구항 수 : 총 11 항

(54) 발명의 명칭 자가 치유 생체 모사형 오류허용 에프피지에이

(57) 요약

본 발명은 오류허용이 가능하며, 스템셀을 통해 부분 재구성성이 가능한 자가 치유 생체 모사형 오류허용 FPGA에 관한 것이다. 본 발명에 따른 FPGA는 제1 함수 및 제2 함수를 입력받아 미리 저장된 에러검출코드와 생성된 에러검출코드 신호를 비교하여 내부의 일시오류 또는 영구오류를 검출하는 복수의 연산 유닛, 연산 유닛과 연결되며, 연산 유닛에 영구오류가 발생할 경우 영구오류가 발생된 연산 유닛의 기능을 대체하며, 부분적으로 재구성성이 가능한 적어도 하나의 스템셀, 각각의 스템셀과 복수의 연산 유닛이 일렬로 연결되어 배치되는 연산블록, 복수의 연산블록이 가로방향 또는 세로방향으로 배열되는 복수의 연산타일 및 복수의 연산타일 중 적어도 2개 이상의 연산타일에서 영구오류가 발생되면 영구오류가 검출된 연산타일의 우선순위를 설정하여 우선순위별로 영구오류가 치유되도록 제어하는 오류허용코어를 포함할 수 있다.

대표도 - 도1



이 발명을 지원한 국가연구개발사업

과제고유번호 2011-0017914

부처명 교과부

연구사업명 중견연구자지원사업(핵심연구)

연구과제명 오류 모듈에 안정적으로 대처하는 재구성 가능한 가상실험장치 구조 동시설계 연구

주관기관 조선대학교

연구기간 2010.05.01 ~ 2013.04.30

특허청구의 범위

청구항 1

제1 함수 및 제2 함수를 입력받아 미리 저장된 에러검출코드와 생성된 에러검출코드 신호를 비교하여 내부의 일시오류 또는 영구오류를 검출하는 복수의 연산 유닛;

상기 연산 유닛과 연결되며, 상기 연산 유닛에 영구오류가 발생할 경우 상기 영구오류가 발생된 연산 유닛의 기능을 대체하며, 부분적으로 재구성 가능한 적어도 하나의 스템셀;

각각의 상기 스템셀과 복수의 상기 연산 유닛이 일렬로 연결되어 배치되는 연산블록;

상기 복수의 연산블록이 가로방향 또는 세로방향으로 배열되는 복수의 연산타일; 및

상기 복수의 연산타일 중 적어도 2개 이상의 연산타일에서 상기 영구오류가 발생되면 상기 영구오류가 검출된 연산타일의 우선순위를 설정하여 상기 우선순위에 따라 상기 영구오류가 치유되도록 제어하는 오류허용코어를 포함하는 FPGA.

청구항 2

제 1 항에 있어서,

상기 오류허용코어는

상기 영구 에러의 유무를 확인하는 상태신호를 생성하는 영구 에러 핸들러; 및

상기 상태신호 및 상기 영구 에러가 발생된 타일의 타일아이디를 이용하여 프레임 생성하는 프레임 생성부를 포함하는 FPGA.

청구항 3

제 2 항에 있어서,

상기 오류허용코어는

상기 영구 에러 핸들러가 상기 타일아이디 정보를 생성하여 상기 프레임 생성부에 제공하는 것을 특징으로 하는 FPGA

청구항 4

제 2 항에 있어서,

상기 오류허용 코어는

복수의 상기 영구 에러 핸들러; 및

미리 설정된 상기 연산타일의 타일아이디를 정보를 상기 프레임 생성부에 제공하는 우선순위 제어부를 더 포함하는 FPGA.

청구항 5

제 2 항에 있어서,

상기 프레임 생성부는

스타트프레임 필드, 타일아이디 필드 및 상태 필드를 포함하는 프레임을 생성하되,

상기 스타트 프레임 필드가 0이 아닌 값이 존재할 경우에만 프레임이 처리되고, 나머지는 프레임이 폐기되는 것을 특징으로 하는 FPGA.

청구항 6

제 2 항 내지 제 5 항 중 어느 한 항에 있어서,

상기 영구 에러 핸들러는 적어도 하나의 우선순위 인코더를 더 포함하는 FPGA.

청구항 7

제 1 항에 있어서,

상기 스템셀은 상기 연산블록의 최상부, 최하부 또는 중간 중 어느 하나의 영역에 배치되는 것을 특징으로 하는 FPGA.

청구항 8

제 7 항에 있어서,

상기 연산블록의 크기는

$B = N + W$ 로 정의되는 것을 특징으로 하는 FPGA.

(여기서, B는 연산블록의 크기, N은 재구성 가능한 연산셀 내의 특업테이블 개수, W는 스템셀의 모든 파티션 핀을 수용할 수 있는 크기임)

청구항 9

제 7 항에 있어서,

상기 제1 함수 또는 제2 함수의 개수는

상기 연산블록 내의 연산셀들의 개수에서 2를 차감하는 것을 특징으로 하는 FPGA.

청구항 10

제 8 항에 있어서,

상기 연산타일의 크기는

상기 연산블록의 크기와 상기 재구성 가능한 연산셀들의 길이를 곱한 것을 특징으로하는 FPGA.

청구항 11

제 10 항에 있어서,

상기 연산타일 내의 셀의 총 개수는 상기 연산타일의 크기에서 상기 스템셀의 개수를 차감한 것을 특징으로 하는 FPGA.

명세서

기술분야

[0001] 본 발명은 FPGA에 관한 것으로, 특히 자가치유 생체 모사형 오류허용 FPGA에 관한 것이다.

배경 기술

[0002] FPGA(Field Programmable Gate Array)는 프로그램이 가능한 논리 블록들의 어레이로 구성되는 집적회로이다.

[0003] 종래 SRAM 기반의 FPGA는 고성능, 저가격, 재프로그래밍이 가능한 이유로 각광받는 집적회로로 특히 원격 제어 시스템들에 많이 사용되고 있다.

[0004] 이러한 SRAM 기반의 FPGA는 환경적으로 방사 또는 복사 등의 영향으로 인하여 회로 기능에 심각한 영향을 줄 수 있다. 예를 들면, SSE(Single Event Effect)는 대기에 존재하는 대전입자가 실리콘에 충돌하여 FPGA 시스템 내의 오류를 유발할 수 있다. 또한, SSE에 의하여 FPGA의 CMOS 등으로 구현된 메모리 소자들은 대전된 입자들에 의해 전달되는 에너지의 양에 따라 일시적오류 또는 고착오류가 발생할 수 있다. 일시적오류는 메모리 소자들 내에서 비트 반전이 발생하는 수준인데 반하여, 고착오류(stuck-at faults)는 영구적 결함으로 나타난다. 이러한 환경에서 프로그래밍이 가능한 장치들은 계속적으로 동작하기 위하여 오류허용 또는 자가 치유형 논리 블록들이 필요하다.

발명의 내용

해결하려는 과제

[0005] 본 발명의 해결하고자 하는 과제는 연산셀에 발생한 오류를 치유하며, 오류허용이 가능한 자가 치유 생체 모사형 오류허용 FPGA를 제공하는 데 있다.

과제의 해결 수단

[0006] 본 발명에 따른 자가 치유 생체 모사형 오류허용 FPGA는 제1 함수 및 제2 함수를 입력받아 미리 저장된 에러검출코드와 생성된 에러검출코드 신호를 비교하여 내부의 일시오류 또는 영구오류를 검출하는 복수의 연산 유닛; 상기 연산 유닛과 연결되며, 상기 연산 유닛에 영구오류가 발생할 경우 상기 영구오류가 발생한 연산 유닛의 기능을 대체하며, 부분적으로 재구성이 가능한 적어도 하나의 스템셀; 각각의 상기 스템셀과 복수의 상기 연산 유닛이 일렬로 연결되어 배치되는 연산블록; 상기 복수의 연산블록이 가로방향 또는 세로방향으로 배열되는 복수의 연산타일; 및 상기 복수의 연산타일 중 적어도 2개 이상의 연산타일에서 상기 영구오류가 발생되면 상기 영구오류가 검출된 연산타일의 우선순위를 설정하여 상기 우선순위별로 상기 영구오류가 치유되도록 제어하는 오류허용코어를 포함할 수 있다.

[0007] 상기 오류허용코어는 상기 영구 에러의 유무를 확인하는 상태신호를 생성하는 영구 에러 핸들러; 및 상기 상태 신호 및 상기 영구 에러가 발생한 타일의 타일아이디를 이용하여 프레임을 생성하는 프레임 생성부를 포함할 수 있다.

[0008] 상기 오류허용코어는 상기 영구 에러 핸들러가 상기 타일아이디 정보를 생성하여 상기 프레임 생성부에 제공할 수 있다.

[0009] 상기 오류허용 코어는 복수의 상기 영구 에러 핸들러 및 미리 설정된 상기 연산타일의 타일아이디를 정보를 상기 프레임 생성부에 제공하는 우선순위 제어부를 더 포함할 수 있다.

[0010] 상기 프레임 생성부는 스타트프레임 필드, 타일아이디 필드 및 상태 필드를 포함하는 프레임을 생성하되, 상기 스타트 프레임 필드가 0이 아닌 값이 존재할 경우에만 프레임이 처리되고, 나머지는 프레임이 폐기될 수 있다.

[0011] 상기 영구 에러 핸들러는 적어도 하나의 우선순위 인코더를 더 포함할 수 있다.

[0012] 상기 스템셀은 상기 연산블록의 최상부, 최하부 또는 중간 중 어느 하나의 영역에 배치될 수 있다.

[0013] 상기 연산블록의 크기는 $B = N + W$ 로 정의될 수 있다. (여기서, B는 연산블록의 크기, N은 재구성 가능한 연산셀 내의 특업테이블 개수, W는 스템셀의 모든 파티션 핀을 수용할 수 있는 크기임)

- [0014] 상기 제1 함수 또는 제2 함수의 개수는 상기 연산블록 내의 연산셀들의 개수에서 2를 차감하여 계산될 수 있다.
- [0015] 상기 연산타일의 크기는 상기 연산블록의 크기와 상기 재구성 가능한 연산셀의 길이를 곱하여 계산될 수 있다.
- [0016] 상기 연산타일 내의 셀의 총 개수는 상기 연산타일의 크기에서 상기 스템셀의 개수를 차감하여 계산될 수 있다.

발명의 효과

- [0017] 본 발명의 일 실시 예에 따른 자가 치유 생태모사형 오류허용 FPGA는 스템셀 및 연산유닛에 포함된 룩업테이블의 재구성이 가능하고, 연산셀에 일시적 오류 또는 영구적 오류가 발생하여도 정상적인 출력신호를 출력하여, 해당 연산셀 및 연산타일이 정상적으로 동작할 수 있다.
- [0018] 또한, 본 발명의 일 실시 예에 따른 자가 치유 생태모사형 오류허용 FPGA는 코어에서 연산타일의 영역별로 구분하여 제어함으로써 효율을 높일 수 있다.

도면의 간단한 설명

- [0019] 도 1은 본 발명의 실시 예에 따른 FPGA에 포함된 연산셀의 신호 입출력 관계를 도시한 도면.
- 도 2는 도 1에 도시된 연산셀의 기능부를 개략적으로 도시한 블록도.
- 도 3(a) 및 도 3(b)는 본 발명의 실시 예에 따른 FPGA의 스템셀과 연산셀의 접속을 도시한 도면들.
- 도 4(a) 및 4(b)는 연산블록 내의 영구적 에러 라우팅의 실시 예들을 도시한 블록도들.
- 도 5(a) 및 도 5(b)는 도 3(a), 도 3(b), 도4(a) 및 도4(b)에 도시된 연산블록의 스템 셀의 두 개의 서로 다른 영역의 두 개의 연산 타일을 도시한 도면.
- 도 6은 연산타일로부터 단일의 영구적 에러 플래그를 생성하는 타일내(intra-tile) 라우팅 아키텍처를 도시한 도면
- 도 7(a) 및 도 7(b)는 본 발명의 실시 예에 따른 자가 치유 생체 모사형 오류허용 FPGA를 도시한 도면.
- 도 8(a) 및 도8(b)는 도 7(a)와 도7(b)에 도시된 오류허용코어의 내부 구성요소들을 도시한 도면
- 도 9는 우선순위 제어기의 내부 구조를 도시한 블록도.
- 도 10(a)와 도 10(b)는 도 7(a)와 도7(b)에 도시된 프레임 생성기를 통해 생성되는 프레임을 도시한 도면.

발명을 실시하기 위한 구체적인 내용

- [0020] 이하, 도면을 참조한 본 발명의 설명은 특정한 실시 형태에 대해 한정되지 않으며, 다양한 변환을 가할 수 있고 여러 가지 실시예를 가질 수 있다. 또한, 이하에서 설명하는 내용은 본 발명의 사상 및 기술 범위에 포함되는 모든 변환, 균등물 내지 대체물을 포함하는 것으로 이해되어야 한다.
- [0021] 이하의 설명에서 제1, 제2 등의 용어는 다양한 구성요소들을 설명하는데 사용되는 용어로서, 그 자체에 의미가 한정되지 아니하며, 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다.
- [0022] 본 명세서 전체에 걸쳐 사용되는 동일한 참조번호는 동일한 구성요소를 나타낸다.
- [0023] 본 발명에서 사용되는 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 또한, 이하에서 기재되는 "포함하다", "구비하다" 또는 "가지다" 등의 용어는 명세서상에 기재된 특징, 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것이 존재함을 지정하려는 것으로 해석되어야 하며, 하나 또는 그 이상의 다른 특징들이나, 숫자, 단계, 동작, 구성요소, 부품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0024] 이하, 본 발명의 실시 예를 첨부한 도 1 내지 도 10을 참조하여 상세히 설명하기로 한다.
- [0025] 본 발명의 실시 예에 따른 오류허용이 가능한 생태모방형 FPGA는 오류허용 재구성이 가능하고, 연산셀에 오류가 발생하여도 정상적인 출력신호를 출력하여, 해당 연산셀, 연산블록 및 연산타일이 정상적으로 동작할 수 있다.

이를 위하여, FPGA는 입력함수들에 대하여 자가 검사를 통해 단방향 오류를 식별하여 오류를 허용할 수 있도록 한다.

[0026] 이하의 설명에서 사용하는 입력함수는 임의의 불리언 함수(Boolean function)으로 정의할 수 있다. 또한, 제1 함수는 임의의 불리언 함수로 정의되며, 제2 함수는 제1 함수의 오류 발생 시 이를 대체할 함수로 제1 함수의 룩업테이블과 동일한 룩업테이블로 표현되는 함수이다.

[0027] 도 1은 본 발명의 실시 예에 따른 FPGA에 포함된 연산셀의 신호 입출력 관계를 도시한 도면이고, 도 2는 도 1에 도시된 연산셀의 기능부를 개략적으로 도시한 블록도이다.

[0028] 도 1 및 도 2를 참조하면, 연산셀은 제1 함수/에러검출코드저장부(100), 제2 함수/에러검출코드저장부(200), 제1 저장부(300), 제2 저장부(400), 자가검사부(500), 함수라우터/영구에러발생표시부(600), 라우터부(700) 및 스위치 박스(800)를 포함할 수 있다.

[0029] 구체적으로, 제1 함수/에러검출코드저장부(100)는 복수의 입력단을 통해 제1 함수가 입력되면 제1 함수를 라우팅시키는 제1 함수신호(Fn_0_out) 또는 제1 함수의 에러 검출을 위한 미리 저장된 제1 에러검출신호(PS_EDC_1)를 출력할 수 있다.

[0030] 이를 위하여, 제1 함수/에러검출코드저장부(100)는 적어도 하나의 룩업테이블(LUT; Look Up Table)로 구성되며, 룩업테이블은 $2^{k-1} \times 1$ 의 함수를 저장할 수 있다.

[0031] 제1 함수/에러검출코드저장부(100)는 제1 함수가 입력되면 저장된 제1 함수 정보와 비교하여 제1 함수신호(Fn_0_out)를 출력할 수 있다. 또한, 제1 함수/에러검출코드저장부(100)는 제1 함수의 에러 검출을 위한 미리 저장된 제1 에러검출코드(PS_EDC-1)를 출력할 수 있다.

[0032] 제1 함수/에러검출코드저장부(100)는 입력된 제1 함수가 k-1 비트일 경우 제1 함수신호(Fn_0_out) 및 제1 에러검출코드(PS_EDC_1)를 자가검사부(500)에 제공할 수 있다. 여기서, k 비트는 제1 함수의 비트 개수로 정의할 수 있다.

[0033] 여기서, 제1 함수신호(Fn_0_out)는 제1 함수를 다른 연산셀(40)로 라우팅시키는 신호이며, 제1 에러검출코드(PS_EDC_1)는 제1 함수의 에러검출을 위한 신호이다.

[0034] 상기의 실시 예는 제1 함수로 k-1 비트 신호가 입력되는 경우를 예를 들어 설명하고 있다. 그러나 제1 함수/에러검출코드저장부(100)는 제1 함수가 k 비트일 경우 그 자체로 제1 함수를 저장하는 저장부로 사용될 수 있다. 이러한 경우는 추후 다시 설명하기로 한다.

[0035] 제2 함수/에러검출코드저장부(200)는 제2 함수가 입력되면 제2 함수를 라우팅시키는 제2 함수신호(Fn_sp_out) 또는 제2 함수의 에러검출을 위한 제2 에러검출코드(PS_EDC_2)를 출력할 수 있다.

[0036] 제2 함수/에러검출코드저장부(200)는 적어도 하나의 룩업테이블(LUT; Look Up Table)로 구성되며, 룩업테이블은 $2^{k-1} \times 1$ 의 함수를 저장할 수 있다. 여기서, k는 제2 함수의 입력 비트 개수이다.

[0037] 제2 함수/에러검출코드저장부(200)는 제2 함수가 입력되면 저장된 제2 함수와 비교하여 제2 함수신호(Fn_sp_out)를 출력할 수 있다. 또한, 제2 함수/에러검출코드저장부(200)는 제2 함수가 입력되면 미리 저장된 제2 에러검출코드(PS_EDC_2)를 출력할 수 있다.

[0038] 제2 함수/에러검출코드저장부(200)는 입력된 제2 함수가 k-1 비트일 경우 제2 함수신호(Fn_sp_out) 및 제2 에러검출코드(PS_EDC_2)를 자가검사부(500)에 제공할 수 있다.

[0039] 여기서, 제2 함수신호(Fn_sp_out)는 제2 함수를 다른 연산셀(40)로 라우팅시키는 신호이며, 제2 에러검출코드(PS_EDC_2)는 제2 함수의 에러검출을 위한 신호이다.

[0040] 상기의 실시 예는 제2 함수로 k-1 비트 신호가 입력되는 경우를 예를 들어 설명하고 있다. 그러나 제2 함수/에러검출코드저장부(200)는 제2 함수가 k 비트일 경우 그 자체로 제2 함수를 저장하는 저장부로 사용될 수 있다. 이러한 경우는 추후 다시 설명하기로 한다.

[0041] 제1 저장부(300)는 내부 라우터로 이용될 수 있다. 제1 저장부(300)는 제1 함수에 따라 그 기능이 가변될 수 있다. 예를 들면, 제1 함수가 k-1 비트 이하일 경우 내부 라우터 기능을 수행한다. 그러나, 제1 함수가 k 비트일

경우 제1 저장부(300)는 제1 함수에 상응하는 제1 에러검출코드(PS_EDC_1)를 저장한다.

- [0042] 제1 저장부(300)가 제1 에러검출코드(PS_EDC_1)를 저장하는 경우에는 제1 함수/에러검출코드저장부(100)가 제1 함수를 저장하는 저장부로 사용되는 경우이며 이때, 제1 함수에 상응하는 제1 에러검출코드(PS_EDC_1)가 저장되어야만 하므로 제1 저장부(300)의 기능이 변환될 수 있다.
- [0043] 제2 저장부(400)는 내부 라우터로 이용될 수 있다. 제2 저장부(400)의 기능은 제2 함수에 따라 가변될 수 있다. 예를 들면, 제2 함수가 k-1 비트 이하일 경우 내부 라우터 기능을 수행할 수 있다. 그러나, 제2 함수가 k 비트 일 경우 제2 저장부(400)는 제2 함수에 상응하는 제2 에러검출코드(PS_EDC_2)를 저장한다.
- [0044] 제2 저장부(400)가 제2 에러검출코드(PS_EDC_2)를 저장하는 경우에는 제2 함수/에러검출코드저장부(200)가 제2 함수를 저장하는 저장부로 사용되는 경우이며 이때, 제2 함수에 상응하는 제2 에러검출코드(PS_EDC_2)가 저장되어야만 하므로 제2 저장부(400)의 기능이 변환될 수 있다.
- [0045] 자가검사부(500)는 제1 함수와 제2 함수의 동작 중 에러검출코드를 생성하며, 기 저장된 에러검출코드와 생성된 에러검출코드를 비교하여 셀의 오류 여부를 확인할 수 있다. 자가검사부(500)는 해당 연산셀의 일시오류 또는 영구오류를 검출할 수 있다. 이를 위하여, 자가검사부(500)는 에러검출코드생성부(510) 및 오류검사부(550)를 포함할 수 있다.
- [0046] 에러검출코드생성부(510)는 제1 함수신호(Fn_O_out)와 제2 함수신호(Fn_sp_out)가 입력되면, 각각의 함수신호들을 내부 로직에 의해 연산하여 제1 에러발생코드(EDC_G_O) 및 제2 에러발생코드(EDC_G_Sp)를 출력할 수 있다. 에러검출코드생성부(510)는 출력된 제1 에러발생코드(EDC_G_O) 및 제2 에러발생코드(EDC_G_Sp)를 오류검사부(550)에 제공한다.
- [0047] 오류검사부(550)는 제1 에러발생코드(EDC_G_O), 제2 에러발생코드(EDC_G_Sp), 제1 에러검출코드(PS_EDC_1) 및 제2 에러검출코드(PS_EDC_2)를 입력받아 내부 연산을 수행하여 제1 오류신호(Faulty_O) 및 제2 오류신호(Faulty_Sp)를 출력할 수 있다.
- [0048] 자가검사부(500)는 오류신호들을 함수라우터/영구에러발생표시부(600)에 제공할 수 있다.
- [0049] 함수라우터/영구에러발생표시부(600)는 오류신호가 있을 경우와 오류신호가 없을 경우 각각에 대하여 연산 결과를 출력할 수 있다. 예를 들면, 함수라우터/영구에러발생표시부(600)는 비오류신호가 검출될 경우 라우터로 동작하여 함수를 라우팅할 수 있다. 또한, 함수라우터/영구에러발생표시부(600)는 오류신호가 있을 경우 에러 플래그를 생성한다.
- [0050] 라우터부(700)는 연산셀 또는 자신과 인접한 연산셀로부터의 영구에러플래그를 라우팅할 수 있다.
- [0051] 스위치 박스(800)는 제1 함수/에러검출코드저장부(100), 제2 함수/에러검출코드저장부(200), 제1 저장부(300), 제2 저장부(400)에서 출력된 신호를 적절하게 자가검사부(500), 함수라우터/영구에러발생표시부(600) 및 라우터부(700)에 제공할 수 있다.
- [0052] 스위치 박스(800)는 CMOS 레벨에서 제조될 수 있으며, 자가검사부(500), 함수라우터/영구에러발생표시부(600) 및 라우터부(700)에 적절한 신호들을 분배할 수 있다. 스위치 박스(800)는 자가검사부(500), 함수라우터/영구에러발생표시부(600) 및 라우터부(700)가 CMOS 레벨에서 제조되는 경우에 필수적으로 사용될 수 있다.
- [0053] 본 발명의 실시 예에 따른 연산셀은 도 2 도시된 제1 함수/에러검출코드저장부(100), 제2 함수/에러검출코드저장부(200), 제1 저장부(300) 및 제2 저장부(400)가 재구성이 가능한 k-1 입력 룩업테이블을 포함할 수 있다. 자가검사부(500), 라우팅함수/영구에러표시부(600) 및 라우터부(700)는 CMOS 레벨에서 경화되거나 기존에 사용되는 FPGA의 k-1 입력 룩업테이블에 구현될 수 있다.
- [0054] 도 3(a) 및 도 3(b)는 본 발명의 실시 예에 따른 FPGA의 스템셀과 연산셀의 연결 관계를 도시한 도면이다. 도 3(a)의 경우 스템셀 상하에 연산셀들이 배치된 것을 도시한 도면이고, 도 3(b)는 스템셀의 일측에 연산셀들이 배치된 것을 도시한 도면이다.
- [0055] 본 발명의 실시 예에서는 각 스템셀이 8개의 연산셀과 접속되는 것을 예를 들어 설명하고 있으나 이에 한정되지 않는다.
- [0056] 도 3(a) 및 도 3(b)를 참조하면, 스템셀(20)은 대응하는 각각의 연산셀들(30)과 접속되며, 연산셀(40)의 개수만

를 서로 상이한 스템함수(F_n _stem)를 구비한다. 본 실시 예에서는 8개의 상이한 스템함수를 포함할 수 있다. 도 3(a)에서 스템셀(20)과 접속되는 연산셀들(40) 각각에 대응하는 스템함수들을 가지며, 연산셀(40)이 영구적 오류를 갖게 될 때마다, 이러한 스템함수들은 그들의 대응하는 연산셀들의 기능을 넘겨받기 위해 부분적으로 재구성될 수 있다..

[0057] 여기서, 스템셀(20) 및 스템셀(20)과 접속되는 연산셀들(40)을 포함하는 연산블록(50)의 폭은 연산셀(40)이 가질 수 있는 룩업 테이블의 개수에 의존한다. 이러한 오류 허용 아키텍처를 기존 FPGA에 포팅하기 위해, 연산블록의 폭은 CLB(Configurable Logic Block)가 가질 수 있는 룩업 테이블의 개수에 의존한다. 연산셀(40)의 모든 구성요소를 단일 재구성 가능한 로직 블록(Configurable Logic Block; 이하, CLB라 함) 내에 포팅하기 위해서는 적어도 8개의 룩업 테이블을 가져야 한다.

[0058] 본 실시 예에서는 각각의 연산셀(40)이 8개의 룩업 테이블을 포함하므로, 각각의 연산셀에 대한 8개의 함수가 수용될 수 있다. 따라서, 연산블록 크기는 $8+1$ (스템 셀을 위한 1개) = 9셀이다. 본 발명의 실시 예에 따른 FPGA는 CMOS 레벨에서 제조되는 경우, 연산블록의 크기는 룩업 테이블 단위로 표현되는 스템셀(20)의 크기에 따라 결정된다. 본 발명의 FPGA 구조를 종래의 FPGA 내에 포팅하기 위하여 하나의 연산셀(40)은 하나의 CLB로 간주할 수 있다. 연산블록 크기를 CLB 단위로 표현한 일반식은 수학식 1과 같다.

[0059] [수학식 1]

$$B = N + W$$

[0061] 여기서, B는 연산블록 크기, N : CLB 내의 룩업 테이블의 수($N=8$), W는 구성 프레임의 폭(스템셀의 모드 파티션 핀을 수용할 수 있는 폭)임.

[0062] 스템셀의 크기는 구성 프레임을 적절하게 부분적으로 재구성하기 위해 구성 프레임의 폭과 동일할 수 있다. 부분재구성 유형에 따라 스템셀의 폭을 선택할 수 있다. 모듈러 기반의 부분재구성 유형인 경우, 소프트웨어는 프록시 로직을 자동으로 삽입한다. 프록시 로직은 단일 룩업테이블1(LUT1)로 구현되며, 정적 및 재구성 영역 사이의 고정 인터페이스를 구현한다. 이 경우, 스템셀의 폭이 스템셀의 모든 파티션 핀을 수용할 수 있을 만큼 커야 한다. 이에 반하여, 차이에 근거한 부분 재구성 유형이 사용되는 경우, 이러한 추가적인 프록시 로직이 필요하지 않게 된다.

[0063] 도 4(a) 및 4(b)는 연산블록 내의 영구적 에러 라우팅의 실시 예들을 도시한 블록도이다.

[0064] 도 4(a) 및 도 4(b)는 각각의 연산셀은 도 2에 도시된 바와 같이 함수라우터/영구에러표시부(86)를 포함할 수 있다. 도 4(a) 및 도 4(b)는 도 2에 도시된 제1 저장부(300) 또는 제2 저장부(400)나, 라우터부(700)의 입력들로 라우팅되는 경로들을 도시하고 있다. 이때, 제1 저장부(300) 또는 제2 저장부(400)는 라우터로 동작하는 기능을 수행하고 있을 경우이다. 도 4(a) 및 도 4(b)에서 도시된 바와 같이, 연산 블록의 다른 절반부에서도 동일한 라우팅 경로가 포함될 수 있다. 이것은 도 2에 도시된 제1 저장부(300) 및 제2 저장부(400)가 라우터로서 이용하기 위해서는, 적어도 단일 연산블록 중의 2개의 연산셀이 K 입력 미만의 함수를 가져야 한다는 것을 나타낸다.

[0065] 따라서, 연산 블록 내의 8개의 연산셀에 대하여, $8-2=6$ 개의 K_입력 함수가 그것의 내부에서 구현될 수 있다. 연산 블록 내의 10개의 연산셀에 대해서는, $10-2=8$ 개의 K_입력 함수가 그것의 내부에서 구현될 수 있다.

[0066] K_입력 함수의 개수는 수학식 2와 같다.

[0067] [수학식 2]

$$K_입력\ 함수의\ 개수 = M - 2$$

[0069] 여기서, M은 연산블록 내의 연산셀들의 개수임.

- [0070] 도 5(a) 및 도 5(b)는 도 3(a), 도 3(b), 도4(a) 및 도4(b)에 도시된 연산블록의 스템 셀의 두 개의 서로 다른 영역의 두 개의 연산 타일을 도시한 도면이다.
- [0071] 연산 타일의 길이는 구성 프레임 길이에 비례한다. 20×1 (20 CLB 길이 곱하기 1 CLB 폭)의 구성 프레임 크기 및 8 룩업 테이블의 CLB 크기에 대하여, 연산타일의 크기는 수학식 3과 같이 가정할 수 있다.
- [0072] [수학식 3]
- [0073] 타일 크기 = $20 \times (8+1) = 180$
- [0074] 수학식 3에서와 같이, 180개의 CLB 중에서, 20개의 수직 CLB가 스템 셀을 위해 이용될 수 있으며, 나머지 160개는 연산셀의 역할을 할 것이다. 구성 프레임은 부분 재구성의 그레인ularity(granularity), 즉 부분 재구성 동안 얼마나 많은 최소 개수의 CLB가 재구성되는지를 정의한다. 여기에서는 구성 프레임 크기가 장치의 일부분을 재구성하는 동안 그것을 나타내는 20×1 CLB이므로, 20×1 CLB가 적어도 재구성된다. 따라서, 160개의 연산셀이 있을 경우 그들 전부에서 동시에 발생하는 영구적 에러들을 순간적으로 치유할 수 있다.
- [0075] 40×2 (40 CLB 길이 곱하기 2 CLB 폭)의 구성 프레임 크기 및 8 룩업 테이블의 CLB 크기에 대하여, 연산타일의 크기는 수학식 4와 같이 계산된다.
- [0076] [수학식 4]
- [0077] 연산타일 크기 = $40 \times (8+2) = 400$
- [0078] 수학식 4에서와 같이, 400개의 CLB 중에서, 80개의 수직 CLB가 스템셀을 위해 이용될 것이고, 나머지 320개는 연산셀의 역할을 할 것이다. 연산타일의 크기는 수학식 5와 같이 정의될 수 있다.
- [0079] [수학식 5]
- [0080] 연산타일 크기 = $L \times (N + W)$
- [0081] 이때, 연산타일의 크기는 수학식 1과 수학식 5를 통해 수학식 6과 같이 변환될 수 있다.
- [0082] [수학식 6]
- [0083] 연산타일 크기 = $L \times B$
- [0084] 여기서, L은 구성 프레임의 길이(재구성 가능한 연산셀들의 길이), B는 연산블록 크기이다.
- [0085] 연산타일 내의 스템 셀들(SC)의 개수는 수학식 7과 같이 계산될 수 있다.
- [0086] [수학식 7]
- [0087] $SC = L \times W$
- [0088] 여기서, L은 구성 프레임의 길이, W는 구성 프레임의 폭이다.
- [0089] 연산타일 내의 하나의 연산셀 내에서 에러가 발생하는 경우, 장치의 구성 입도로 인해 그 타일 내의 스템셀 전

부가 재구성된다. 여기서, 오류를 갖지 않는 연산셀에 대응하는 스템셀들의 재구성은 그들의 기능성에 영향을 주지 않는데, 왜냐하면 그들은 재구성이 수행되는 동안 실행 중이기 때문이다.

[0090] 본 발명에 따른 스템셀은 영구적 에러가 단일 연산셀에서 발생하든 다수의 셀에서 발생하든 간에, 영구적 에러가 특정 연산타일에서 발생함을 나타내는 단일 플래그 신호가 연산셀로부터 추출되어야 한다. 임의의 연산타일 내에서 얼마나 많은 수의 영구적 에러가 존재하는지에 상관없이, 오류가 발생된 모든 연산타일은 한 번에 치유될 수 있다.

[0091] 그러므로, 영구적 에러로부터 동시에 함께 치유될 수 있는 연산타일 내의 셀의 총 개수 C는 수학식 8과 같다.

[0092] [수학식 8]

[0093] $C = \text{타일 크기} - SC$

[0094] 여기서, SC는 스템 셀의 개수이고 SC 개의 스템셀은 연산타일에서 발생된 모든 오류를 치유할 수 있도록 충분한 개수의 록업테이블을 가지고 있다.

[0095]

[0096] 도 6은 연산타일로부터 단일 영구적 에러 플래그를 생성하는 타일내(Intra-Tile) 라우팅 아키텍처를 도시한 도면이다.

[0097] 도 4(a)에 도시된 바와 같이, 단일의 영구적 에러 신호는 연산블록의 각 절반부로부터 생성된다. 이러한 신호들 전부는 타일의 경계에 위치된 연산셀의 라우터부 내로 라우팅된다. 이때, 20개의 연산블록이 존재한다면, 모든 연산블록의 각각의 절반부로부터 하나씩, 2개의 그러한 신호가 존재할 것이고, 따라서 타일로부터 모두 합쳐서 40개의 신호가 존재할 것이다. 이러한 신호들은 5개의 신호의 그룹으로 더 분할된다. 도 6에 도시된 영역A(Blk-A)는 이러한 5개의 신호가 도 2에 도시된 제2 저장부(400)에 입력되는 것을 나타낸다. 제2 저장부(400)로부터 나오는 이러한 신호들의 개수는 연산타일의 각각의 절반부로부터 4개일 것이다. 이러한 신호들은 영역B(Blk-B) 내에 도시된 함수라우터/영구에러표시부(600) 내로 다시 라우팅된다. 연산타일의 각각의 절반부에서, 연산타일로부터의 단일의 영구적 에러 플래그를 생성하기 위해 함수라우터/영구에러표시부(600)에 입력되는 단일 신호가 생성된다.

[0098] 연산타일내 영구적 에러 라우팅의 이러한 방법을 따르면, 상이한 타일 크기들에 대해 쉽게 이용될 수 있는 라우터부를 여전히 이용할 수 있으므로, 각각의 연산셀로부터 생성되는 영구적 에러 신호는 동일한 라우팅 지연을 가진다.

[0099] 도 7(a) 및 도 7(b)는 본 발명의 실시 예에 따른 자가 치유 생체 모사형 오류허용 FPGA를 도시한 도면이다.

[0100] 도 7(a) 및 도 7(b)에 도시된 바와 같이, 이웃하는 연산타일들 내에 스템셀들은 서로 다른 라인에 배치될 수 있다.

[0101] 스템셀들의 영역은 부분적으로 재구성 가능하므로, 또한 네트들은 그 부분적으로 재구성 가능한 영역을 통과할 수 없다는 사실로 인해, 이것은 이웃하는 타일들이 이웃하는 타일들 내의 스템 셀들이 상이한 배치를 가져야만 하는 이유이다. 그들이 동일한 행을 따라 배열된 경우, 각각의 타일의 우측 및 좌측 절반부는 그 자신과만 통신할 수 있을 뿐 아니라, 이웃하는 타일과는 통신할 수 없다.

[0102] 도 7(a)은 각각의 타일이 어느 타일에서 영구적 오류를 갖는지를 식별하기 위해 FPGA의 절반부들 둘 다에 대하여 그것의 고유 ID CT0, CT1, ..., CT21에 의해 표시된다. 모든 타일에서 CT0 또는 CT43 중 하나는 영구 오류 치료에 대한 높은 선택이 부여될 수 있다. 높은 선호도가 CT0에 주어지고, 최하 선호도가 CT43에 주어질 때, 이러한 경우, 예를 들어 CT[0:7]에서 동시에 영구 오류 발생을 가정하면, CT4에서 발생한 에러는 CT[0:3]의 영구 에러가 치유될 때까지 기다려야 한다. 이러한 상황을 방지하기 위해 FPAG는 그림 7a에 도시된 바와 같이, 두 개의 절반으로 나눌 수 있으며, 영구 오류를 가지는 타일을 식별하기 위하여 FPAG의 양 반쪽은 각각의 타일이 자기의 고유 ID CT0, CT1, CT21로 표시된다. 상기의 잘못된 시나리오에 따라 좌측 절반이 높은 우선 순위가 할당되면 좌측 절반의 CT0가 먼저 치유되고 우측의 CT0(도 7(a)에 CT4)가 치유될 것이다. 순차적으로 좌측의 CT1이 치유되고, 우측의 CT1이 치유된다. 마찬가지로 FPAG는 영구 에러 치유를 향상시키기 위해 4개 영역으로 나눌 수 있

다.

- [0103] 이러한 오류 허용 FPGA는 오류 허용이 가능한 코어에 의해 제어된다. 기존 아키텍처를 오류 허용 능력과 함께 사용하기 위해, 이 코어는 FPGA 중심에서 초기화되고 배치될 수 있다.
- [0104] 도 8(a)와 도 8(b)는 도 7(a)와 도 7(b)에 도시된 오류허용코어의 내부 구성요소들을 도시한 도면이고, 도 9는 도 8(b)에 도시된 우선순위 제어부를 도시한 블록도이고, 도 10(a) 및 도 10(b)는 도 8(a)와 도 8(b)에 도시된 프레임 생성부에서 생성되는 프레임을 각각 도시한 도면들이다.
- [0105] 도 8(a) 내지 도 10(b)를 참조하면, 오류허용코어(30)는 영구 에러 핸들러(910), 코어부(920), 프레임 생성부(930) 및 인터페이스부(940)를 포함할 수 있다.
- [0106] 도 8(a)의 경우 도 7(a)에 도시된 FPGA와 같이 오류허용코어가 담당하는 영역이 구분되지 않을 경우의 예를 도시한 도면이고, 도 8(b)는 도 7(b)에 도시된 FPAG와 같이 오류허용코어(30)가 2개의 구역으로 구획된 경우의 오류허용코어를 도시한 도면이다.
- [0107] 먼저, 도 8(a)에 도시된 바와 같이, 오류허용코어(30)는 영구 에러 핸들러(910), 코어부(920), 프레임 생성부(930) 및 인터페이스부(940)를 포함할 수 있다.
- [0108] 구체적으로, 영구 에러 핸들러(910)는 단일 연산셀 또는 다수의 연산셀에 영구적 에러를 포함하는 연산타일들의 타일ID들을 생성한다. 또한, 영구 에러 핸들러(910)는 상태비트를 생성하도록 한다. 영구 에러 핸들러(910)는 적어도 하나의 우선 순위 인코더를 포함할 수 있다. 우선 순위 인코더는 타일들 중 2개 이상에서 동시에 영구에러가 발생할 때, 어느 타일이 가장 먼저 치유되어야 하는지를 결정하여 신호를 출력한다.
- [0109] 프레임 생성부(930)는 동적인 부분적 재구성을 통해 영구적 에러들을 치유하기 위해 온라인 소프트웨어에 의해 이용되는 충분한 정보를 포함하는 프레임을 생성한다. 예를 들면, 프레임 생성부(930)는 도 10(a)에 도시된 바와 같이, 3개의 필드로 구성된 프레임을 생성할 수 있다. 도 10(a)에서와 같이, 프레임에 포함되는 연산타일 ID들은 6비트 폭의 영구 에러 핸들러(910)로부터 인터페이스를 통해 출력되는 결합이 있는 연산타일의 타일ID들을 보유할 수 있다.
- [0110] 여기서, 상태비트는 영구 에러 핸들러(910)로부터 취해진 0 또는 1을 갖는다. 여기서, 상태비트는 "0"은 영구에러 없음을 나타내며, 상태비트 "1"은 영구에러를 나타낼 수 있다.
- [0111] 코어부(920)는 오류허용코어(30)의 전체 제어를 담당한다. 코어부(920)는 생성된 프레임을 인터페이스부(940)를 통해 온라인 소프트웨어에 제공한다.
- [0112] 오류허용코어(30)의 다른 실시 예로서, 도 8(b)와 같이 구성될 수 있다.
- [0113] 도 8(b)를 참조하면, 오류허용코어(30)는 도 8(a)의 영구 에러 핸들러(910)가 2개로 구분된 것을 제외하고는 동일한 구성요소를 구비한다. 각각의 영구 에러 핸들러(912, 914)는 우선 순위 인코더를 포함할 수 있다.
- [0114] 제1 영구 에러 핸들러(912)는 FPGA의 중앙의 가상의 선을 기준으로 우측 영역을 담당하고, 제2 영구 에러 핸들러(914)는 좌측 영역을 담당한다.
- [0115] 제1 영구 에러 핸들러(912)와 제2 영구 에러 핸들러(914)는 입력된 신호들을 통해 상태비트를 생성하도록 한다.
- [0116] 제1 영구 에러 핸들러(912)와 제2 영구 에러 핸들러(914) 각각에는 우선순위 인코더가 포함될 수 있다. 제1 영구 에러 핸들러(912)에 포함된 우선 순위 인코더는 FPAG의 좌측 타일들 중 둘 이상에서 영구적 에러가 한 번에 발생할 때 어느 타일이 가장 먼저 치유되어야 하는지를 결정한다. 마찬가지로, 제2 영구 에러 핸들러(914)는 우측 절반부에 대해 동일한 결정을 내린다.
- [0117] 우선 순위 제어부(950)는 단일 연산셀 또는 다수의 연산셀에 영구적 에러를 포함하는 연산타일들의 연산타일ID들을 생성하도록 할 수 있다. 또한, 우선 순위 제어부(950)는 에러가 FPAG의 두 영역에서 동시에 발생할 때, 좌측 영역에 우선권을 줄 수 있다. 그러나 우선 순위 제어부(950)는 사용자의 설정에 따라 우측 영역에 우선권을 줄 수도 있다. 도 9에서와 같이, 우선 순위 제어부(950)는 2개의 OR 게이트와 2-1 우선순위 인코더(952), 제어모듈(956) 등을 구비할 수 있다. 여기서, FPGA의 4개의 영역을 각각 제어할 경우에는 4개의 OR 게이트, 4-1 인코더, 제어모듈 등을 구비할 수 있음은 당연하다 할 것이다.
- [0118] 예를 들면, 우선 순위 인코더로부터 출력되는 신호를 수신하여 오류가 FPAG의 절반부들 둘 다에 존재하는지, 아

니면 FPAG의 어느 한 절반부에 존재하는지를 나타낸다. 제어 모듈(956)은 단순히 한 입력(954)이 디폴트로 1인 AND 게이트이다. 제어 모듈로부터 출력되는 신호는 타일ID들이 FPAG의 어느 절반부에 속하는지를 나타낸다.

[0119] 에러가 FPAG의 절반부들 둘 다에 존재한다고 가정하면 (우선순위 인코더(952)의 값이 11일 것임), 제어 모듈(952)의 출력은 1일 것이고, 온라인 소프트웨어는 좌측 절반부에 존재하는 에러들을 먼저 치유할 것이다. 좌측 절반부 상의 모든 에러를 치유한 후, 온라인 치유 소프트웨어는 FPAG의 우측 절반부를 선택하기 위해 외부 신호를 디어써트(deassert)한다.

[0120] 프레임 생성부(930)는 동적인 부분적 재구성을 통해 영구적 에러들을 치유하기 위해 온라인 소프트웨어에 의해 이용되는 충분한 정보를 포함하는 프레임을 생성할 수 있다. 이때, 코어부(920)는 이 프레임을 직렬 인터페이스부(940)를 통해 온라인 소프트웨어에 제공한다.

[0121] 프레임 생성부(930)는 도 10(b)에 도시된 바와 같이, 타일ID, 상태비트, 제어신호, L/R 칩 선택 비트 등을 생성한다. 예를 들면, 도 10(b)에 도시된 바와 같이, 프레임 생성부(930)에서 생성되는 프레임은 4개의 필드로 구성된다.

[0122] 타일 ID들은 우선순위 제어기로부터 인터페이스를 통해 입력하는 오류 타일들의 타일 ID들을 보유할 수 있다.

[0123] L/R 칩 선택 비트들은 FPAG의 어느 절반부가 에러를 갖는지에 관한 정보를 저장하고, 우선 순위 제어부(950)로부터 인터페이스를 통해 피딩할 수 있다. 이때, "10"은 좌측에 배치된 타일들 중 어느 하나가 에러를 가진 것을 나타내고, "01"은 우측에 배치된 타일들 중 어느 하나가 에러를 가진 것을 나타내며, "11"은 양측에 에러를 가진 것을 나타낸다.

[0124] 제어신호는 타일ID들 내에 존재하는 데이터가 FPAG의 어느 영역에 속하는지를 나타내고, 우선 순위 제어부(950)로부터 인터페이스를 통해 피딩된다. 여기서, "0"은 FPAG의 우측 영역, 1은 FPAG의 좌측 영역을 나타낸다.

[0125] 상태 비트들은 인터페이스를 통해 양 영역에 대해 제1 및 제2 영구 에러 핸들러(914)로부터 취해진 값이다. 이때, "00"은 영구적 에러 없음, "01"은 우측 영역에 영구적 에러가 존재, "10"은 좌측 영역에 영구적 에러가 존재함을 나타내며, "11"은 양 영역들 둘 다에 영구적 에러가 존재함을 나타낸다.

[0126] 이러한 상태 비트들은 SOF(Start of Frame) 필드의 목적에 알맞다. 온라인 소프트웨어는 이 필드 내에 "0"이 아닌 값이 존재할 때에만 프레임을 처리하고, 그렇지 않으면 프레임이 폐기된다.

[0127] 상기에서 설명한 바와 같이, 본 발명의 일 실시 예에 따른 FPGA는 스템셀 및 연산유닛에 포함된 록업테이블의 재구성이 가능하고, 연산셀에 일시적 오류 또는 영구적 오류가 발생하여도 정상적인 출력신호를 출력하여, 해당 연산셀 및 연산타일이 정상적으로 동작할 수 있다.

[0128] 또한, 본 발명의 일 실시 예에 따른 FPGA는 코어에서 연산타일의 영역별로 구분하여 제어함으로써 효율을 높일 수 있다.

부호의 설명

[0129] 20: 스템셀

30: 오류허용코어

40: 연산셀

50: 연산블록

100: 제1 함수/에러검출코드저장부

100: 제1 함수/에러검출코드저장부

200: 제2 함수/에러검출코드저장부

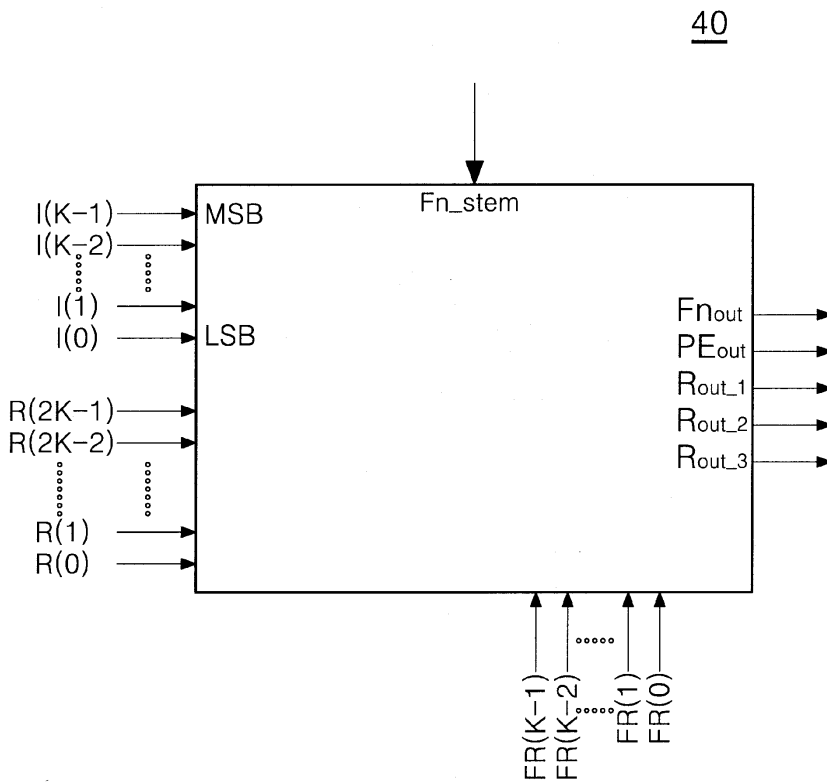
300: 제1 저장부

400: 제2 저장부

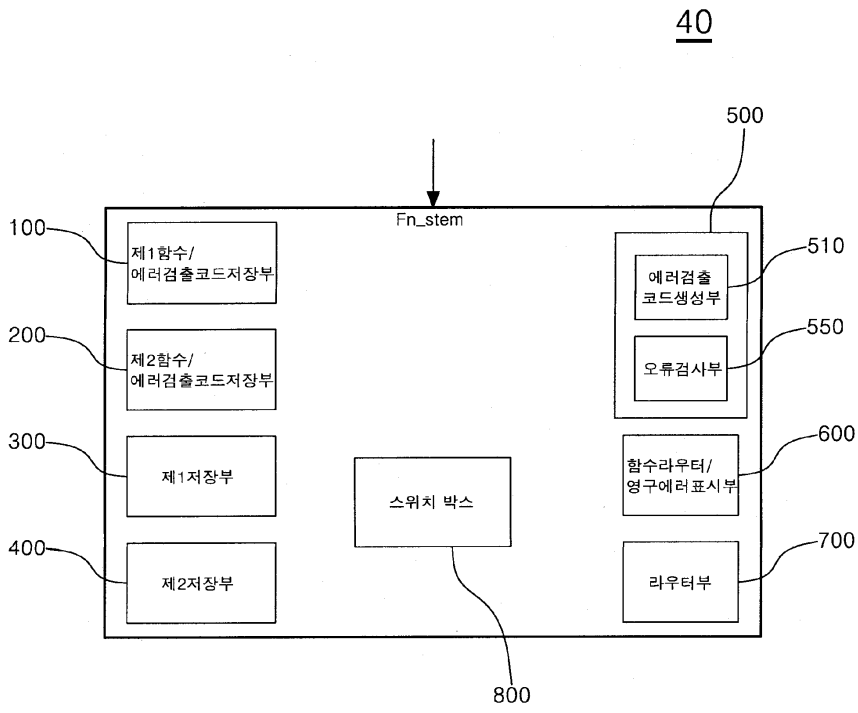
- 500: 자가검사부
- 600: 함수라우터/영구에러발생표시부
- 700: 라우터부
- 800: 스위치박스
- 910: 영구 에러 핸들러
- 920: 코어부
- 930: 프레임 생성부
- 940: 인터페이스부
- 950: 우선순위 제어부

도면

도면1

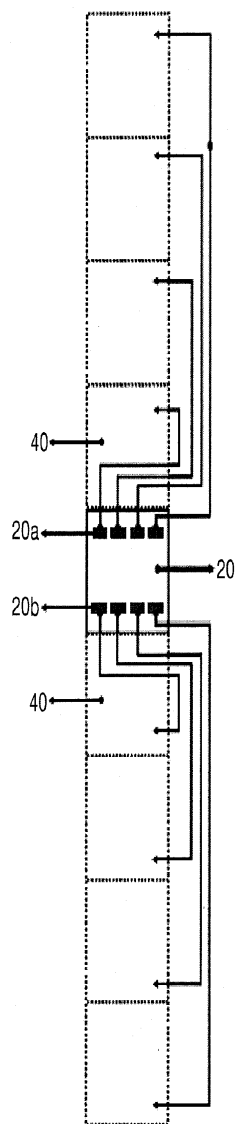


도면2



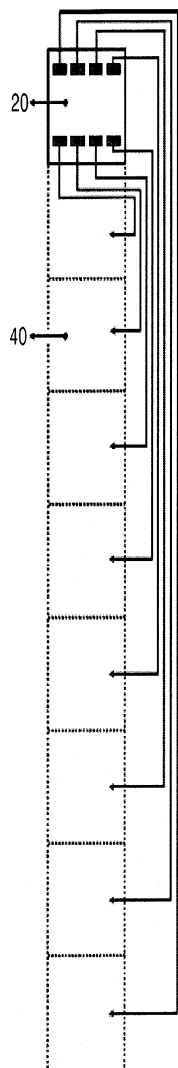
도면3a

50



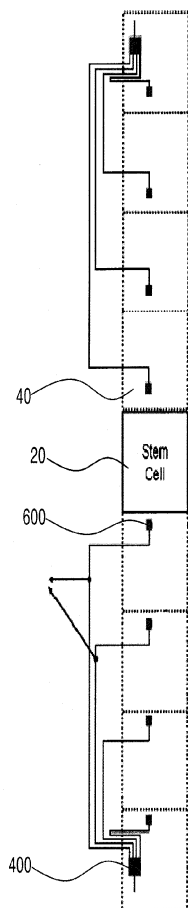
도면3b

50

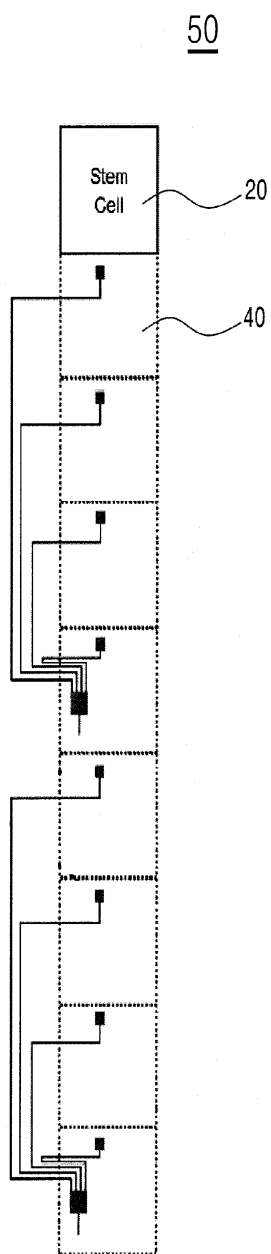


도면4a

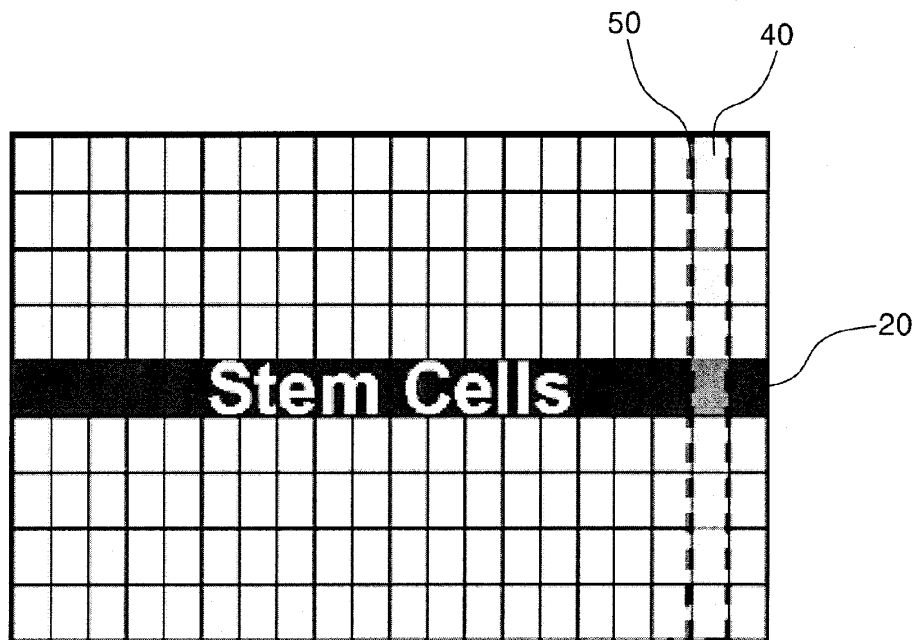
50



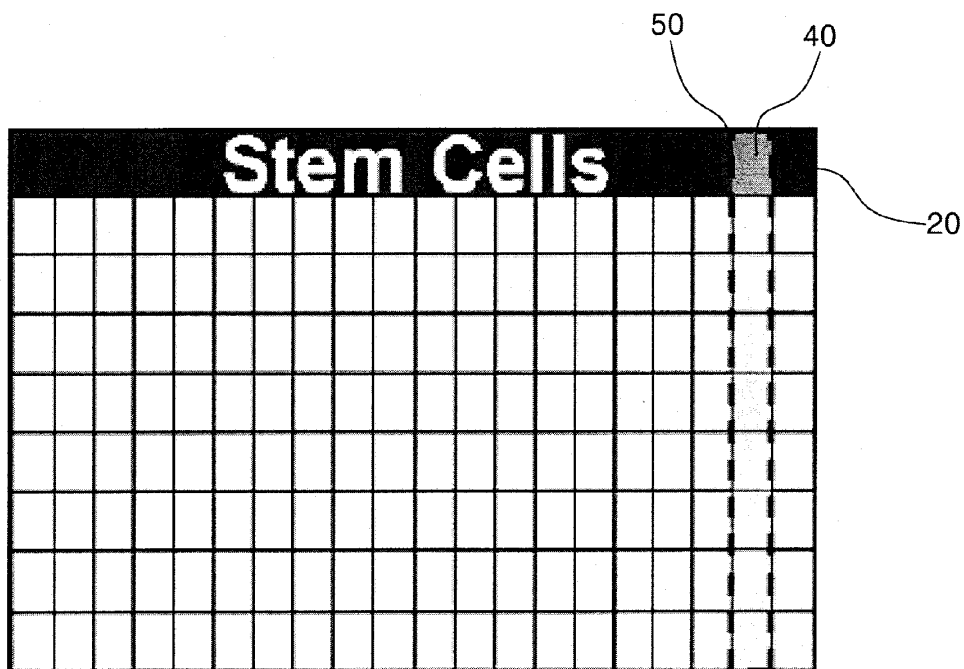
도면4b



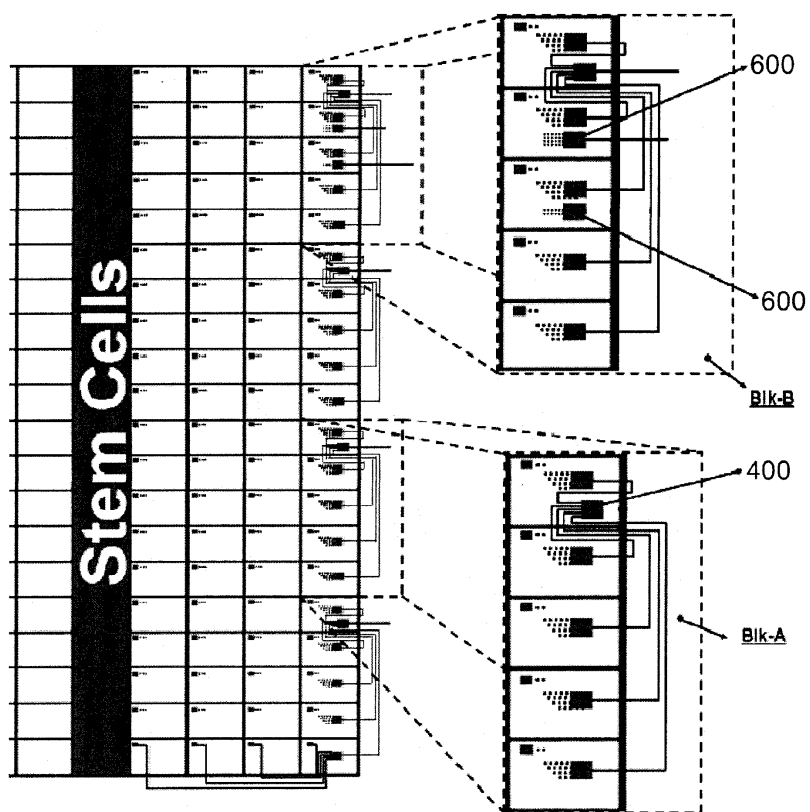
도면5a



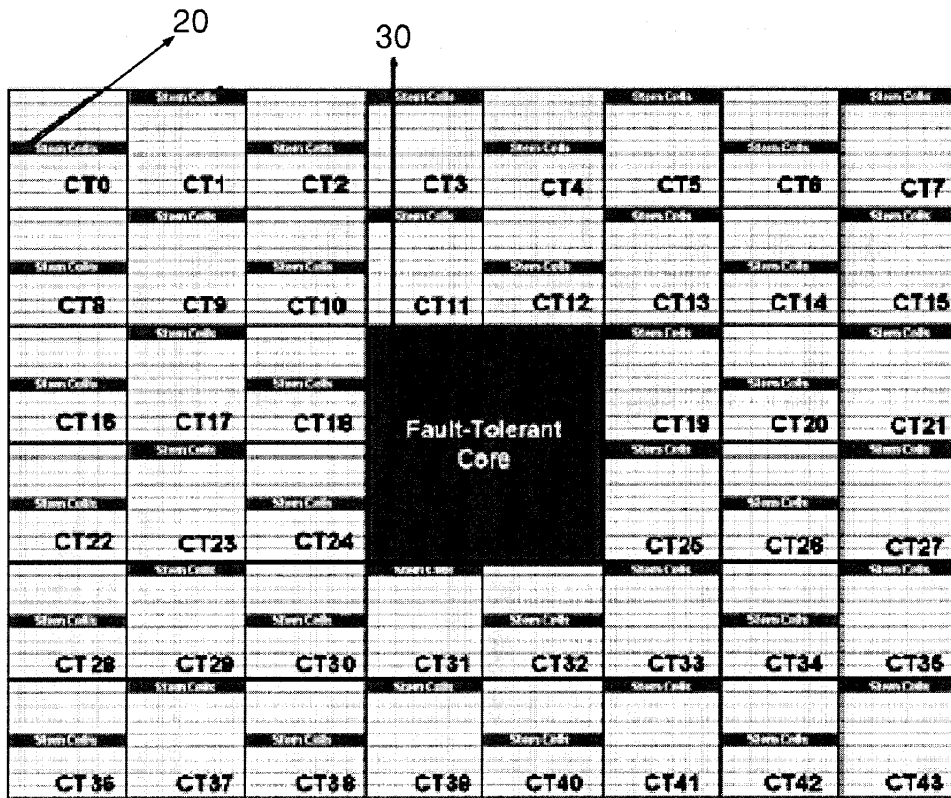
도면5b



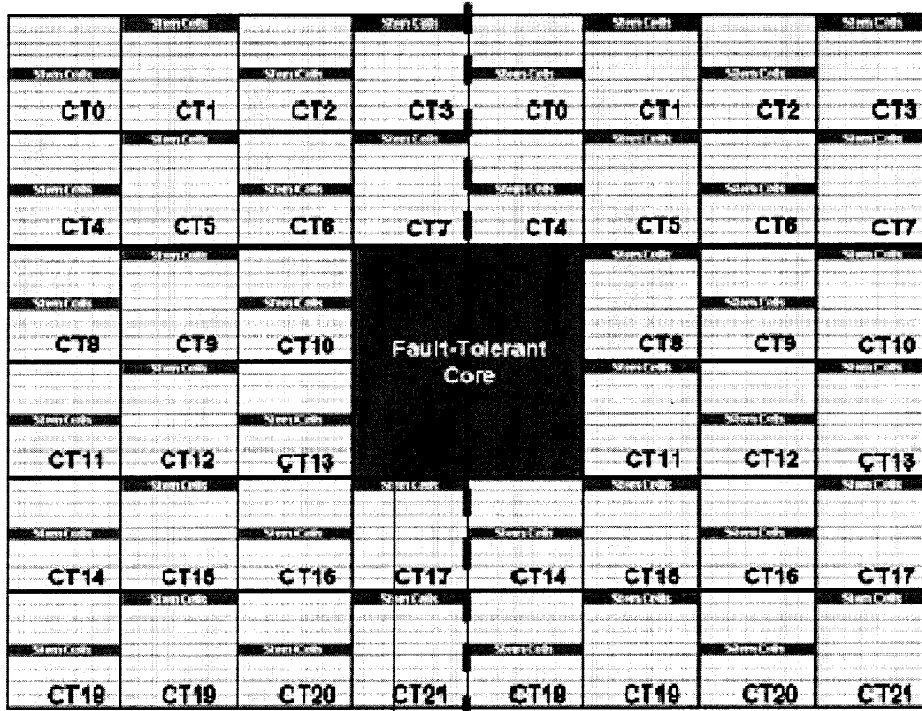
도면6



도면7a

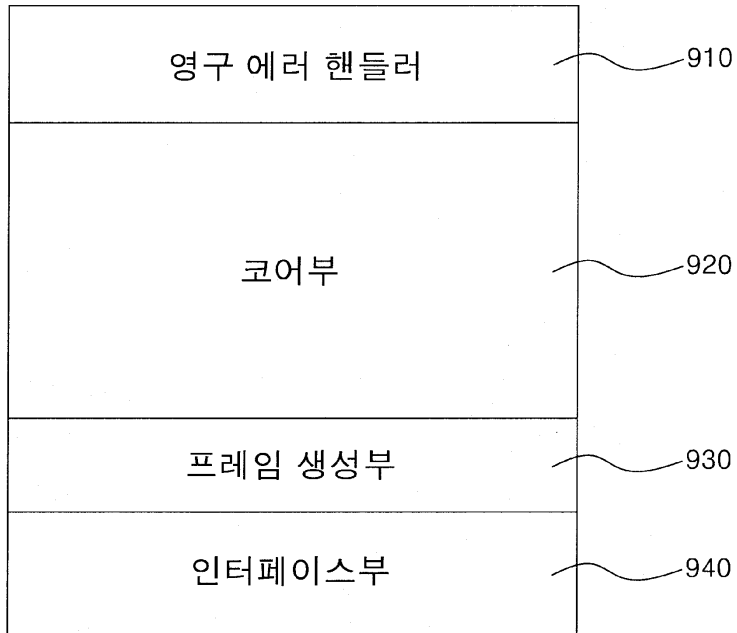


도면7b

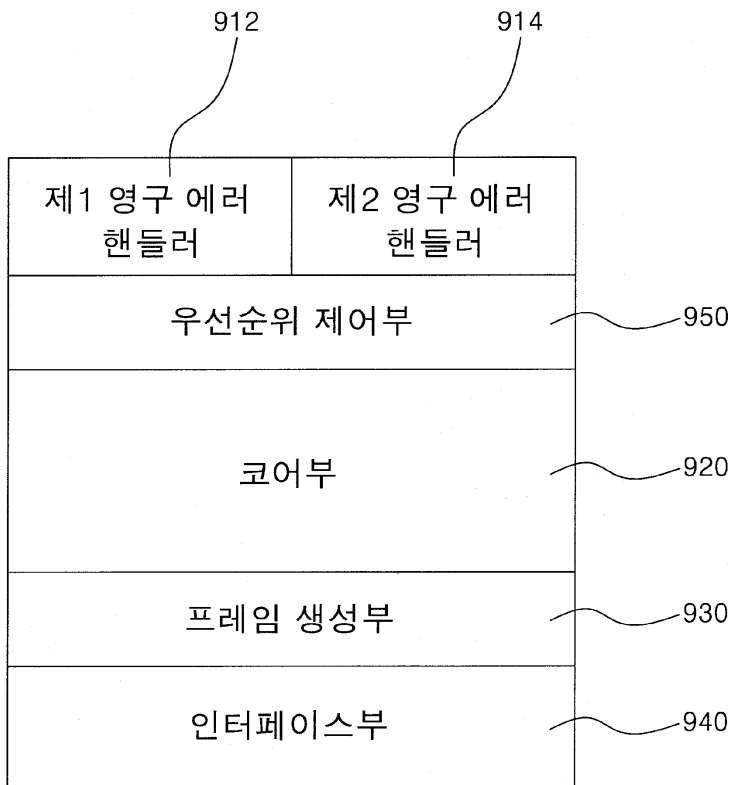


30

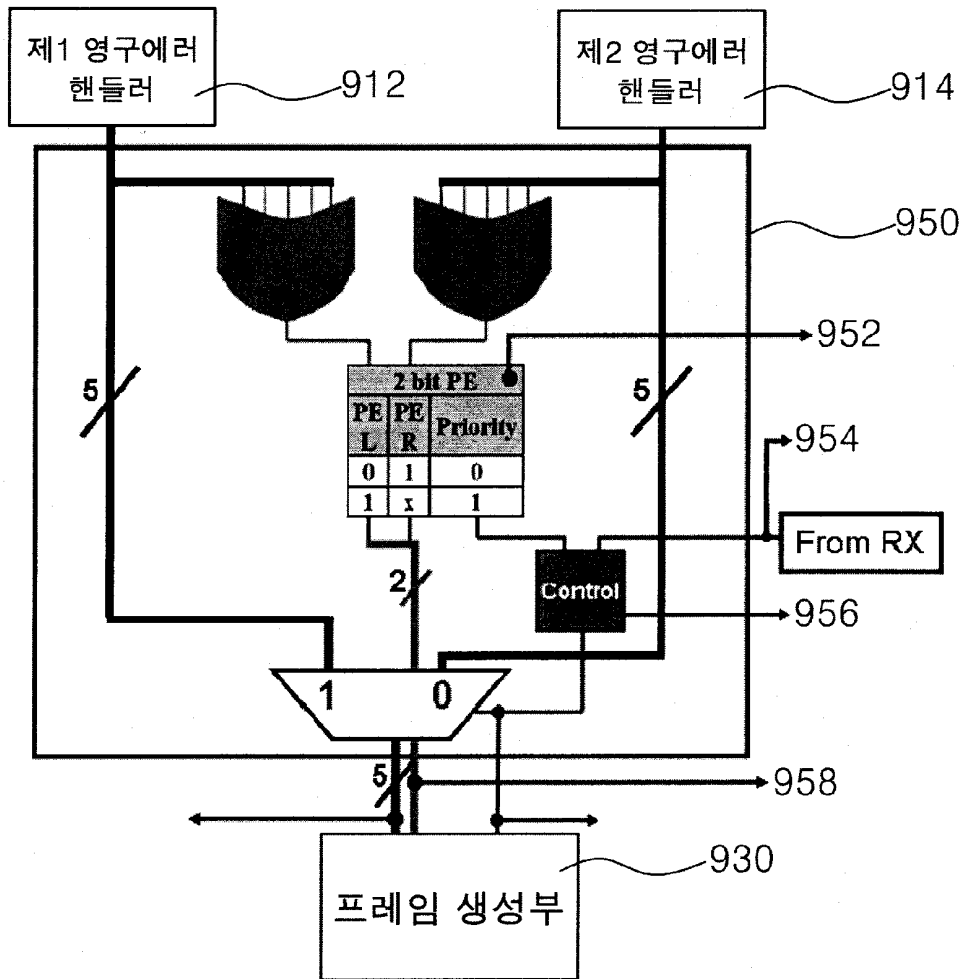
도면8a



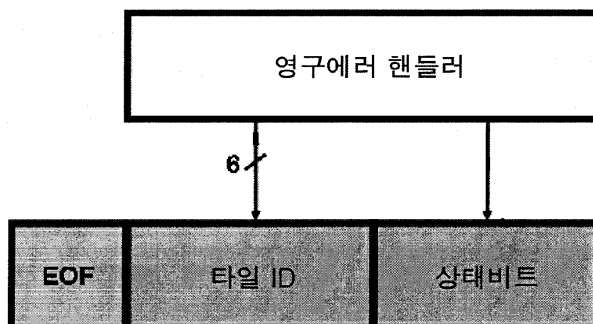
도면8b



도면9



도면10a



도면10b

