



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2024년09월12일
(11) 등록번호 10-2705926
(24) 등록일자 2024년09월06일

(51) 국제특허분류(Int. Cl.)
G11C 11/16 (2006.01) G06N 3/063 (2023.01)
(52) CPC특허분류
G11C 11/1675 (2013.01)
G06N 3/063 (2013.01)
(21) 출원번호 10-2023-0117170
(22) 출원일자 2023년09월04일
심사청구일자 2023년09월04일
(56) 선행기술조사문헌
KR102517156 B1*
US20220375520 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
인하대학교 산학협력단
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
(72) 발명자
김형진
서울특별시 금천구 시흥대로47길 43, 9동 1205호
(시흥동, 럭키아파트)
윤상욱
경기도 부천시 경인로 558, 나동 501호(괴안동, 남영홈타운)
이정진
인천광역시 미추홀구 경인남길102번길 56, 201호
(용현동)
(74) 대리인
양성보

전체 청구항 수 : 총 6 항

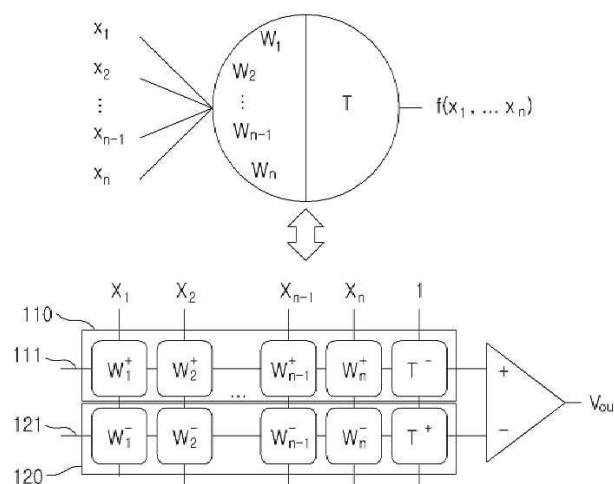
심사관 : 윤석채

(54) 발명의 명칭 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조 및 그 동작 방법

(57) 요약

프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조 및 그 동작 방법 및 시스템이 제시된다. 본 발명에서 제안하는 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조는 전압을 인가하여 비트라인(bitline)의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하는 워드라인(wordline) 및 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되는 비트라인을 포함하고, 상기 메모리 소자 어레이 구조는 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되고, 상기 가중치와 임계값의 차동 메모리 소자 쌍은 서로 다른 부호로 매핑된다.

대표도 - 도1



(52) CPC특허분류

G11C 11/1655 (2013.01)

G11C 11/1657 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711183491
과제번호	2020M3H5A1081111
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	원천기술개발사업
연구과제명	[Ezbaro] 대기환경 감지 및 분류를 위한 멤리스터 기반 하드웨어 인공지능 기술
기 여 율	4/10
과제수행기관명	인하대학교
연구기간	2023.01.30 ~ 2024.01.29

이 발명을 지원한 국가연구개발사업

과제고유번호	1711193592
과제번호	2021-0-02052-003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	대학ITRC센터육성지원사업(교육운영수익)
연구과제명	[Ezbaro][정부] 스마트 모빌리티를 위한 인공지능 시스템반도체 핵심 기술 개발 및

인력 양성(3차년도)

기 여 율	2/10
과제수행기관명	인하대학교
연구기간	2023.01.01 ~ 2023.12.31

이 발명을 지원한 국가연구개발사업

과제고유번호	1711183638
과제번호	2021R1C1C1014530
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	우수신진연구
연구과제명	[Ezbaro] 3차원 적층형 멤리스터 크로스바 어레이 구조를 이용한 물리적 복제 불가

함수 및 하드웨어 보안 시스템 개발

기 여 율	4/10
과제수행기관명	인하대학교
연구기간	2023.03.01 ~ 2024.02.29

명세서

청구범위

청구항 1

메모리 소자 어레이 구조에 있어서,

워드라인(wordline) 및 비트라인(bitline)을 포함하고,

상기 워드라인에 전압을 인가하고, 상기 비트라인의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하며,

상기 비트 라인은 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되며,

상기 메모리 소자 어레이 구조는,

부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되고, 상기 가중치와 임계값의 차동 메모리 소자 쌍은 서로 다른 부호로 매핑되어, 인접한 두 비트라인의 전류를 별도의 비교 신호 인가 없이 메모리 어레이에서의 행렬-곱 연산 과정으로 가중치와 임계값을 비교하는

메모리 소자 어레이 구조.

청구항 2

제1항에 있어서,

부울 논리에 대한 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정하는

메모리 소자 어레이 구조.

청구항 3

제2항에 있어서,

상기 가중치 및 조정된 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑하여, 상기 가중치 및 조정된 임계값을 컨덕턴스로 변환하는

메모리 소자 어레이 구조.

청구항 4

제3항에 있어서,

상기 가중치와 컨덕턴스를 차동 메모리 소자 쌍으로 구성함으로써 인버터 없이 NOT 연산이 가능한

메모리 소자 어레이 구조.

청구항 5

워드라인(wordline) 및 비트라인(bitline)을 포함하고, 상기 워드라인에 전압을 인가하고, 상기 비트라인의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하며, 상기 비트 라인은 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되며, 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되는 메모리 소자 어레이의 동작 방법에 있어서, 프로그래밍할 부울 논리를 수신하는 단계;

부울 논리의 진리표에 따라 ILP(Integer Linear Programing) 알고리즘을 통해 가중치와 임계값을 설정하는 단계;

프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정하고, 가중치와 임계값을 컨덕턴스로 변환하는 단계;

부울 논리에 대해 가중치와 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑

하는 단계; 및

가중치와 임계값 컨덕턴스를 메모리 소자에 전사하는 단계

를 포함하고,

상기 가중치와 임계값의 차동 메모리 소자 쌍은 서로 다른 부호로 매핑되어, 인접한 두 비트라인의 전류를 별도의 비교 신호 인가 없이 메모리 어레이에서의 행렬-곱 연산 과정으로 가중치와 임계값을 비교하는

메모리 소자 어레이 동작 방법.

청구항 6

삭제

청구항 7

제5항에 있어서,

상기 가중치와 컨덕턴스를 차동 메모리 소자 쌍으로 구성함으로써 인버터 없이 NOT 연산이 가능한

메모리 소자 어레이 동작 방법.

발명의 설명

기술 분야

[0001] 본 발명은 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조 및 그 동작 방법에 관한 것이다.

배경 기술

[0002] 메모리 내 연산(in-memory computing)은 기존의 컴퓨팅 아키텍처의 한계를 해결하기 위한 유망한 패러다임으로 떠오르고 있다. 이 기술은 메모리 소자 내에서 직접 계산을 수행하는 것을 의미한다. 특히, 임계값 논리를 이용한 메모리 내 연산 기술은 가중치와 임계값이 서로 다른 논리 상태를 구분하는 선형 전달 함수를 활용하여 작동된다. 메모리 소자 내의 프로그래밍 정확도와 노이즈 신호의 영향을 최소화하면서 메모리 내 연산을 빠르고 안정적으로 수행하는 것이 메모리 내 연산의 핵심 동작 원리이다.

[0003] 하지만, 메모리 소자로 구성된 어레이 구조 내에서 각 소자를 프로그래밍 과정 동안 인접 셀의 영향 등을 받아 프로그래밍 정확도가 떨어질 수 있으며, 이로 인해 메모리 내 연산 과정에서 에러가 발생할 수 있다. 따라서, 프로그래밍 가능한 컨덕턴스 내에서 프로그래밍 오차를 최소화하고, 가중치와 임계값에 대한 메모리 내 연산을 빠르고 안정적으로 수행할 수 있는 프로그래밍 가능한 임계값 논리 기술을 필요로 한다.

선행기술문헌

특허문헌

[0004] (특허문헌 0001) 국제 공개특허 제2015-006342호 (2015.01.15)

발명의 내용

해결하려는 과제

[0005] 본 발명이 이루고자 하는 기술적 과제는 임계값 비교를 위한 별도의 신호 인가를 필요로 하지 않도록 가중치와 임계값을 서로 반대 방향으로 메모리 어레이의 두 워드라인에 매핑하여 반대 부호를 가진 차동 메모리 소자 쌍으로 구성하는 메모리 소자 어레이 구조 및 그 동작 방법을 제공하는데 있다. 가중치와 임계값을 서로 반대 방향으로 메모리 어레이의 두 워드라인에 매핑하여 벡터-곱 연산 과정에서 가중치와 임계값을 비교하고, 또한, 가중치와 임계값을 메모리 소자의 상태로 매핑하는 과정에서 소자가 사용 가능한 최소 및 최대 상태를 임계값 논리의 최소 및 최대값에 매핑하여 프로그래밍 오차 및 노이즈 신호 문제를 해결하고자 한다.

과제의 해결 수단

- [0006] 일 측면에 있어서, 본 발명에서 제안하는 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조는 전압을 인가하여 비트라인(bitline)의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하는 워드라인(wordline) 및 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되는 비트라인을 포함하고, 상기 메모리 소자 어레이 구조는 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되고, 상기 가중치와 임계값의 차동 메모리 소자 쌍은 서로 다른 부호로 매핑된다.
- [0007] 본 발명의 실시예에 따르면, 부울 논리에 대한 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정한다.
- [0008] 본 발명의 실시예에 따르면, 상기 가중치 및 조정된 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑하여, 상기 가중치 및 조정된 임계값을 컨덕턴스로 변환한다.
- [0009] 본 발명의 실시예에 따르면, 상기 가중치와 컨덕턴스를 차동 메모리 소자 쌍으로 구성함으로써 인버터 없이 NOT 연산이 가능하다.
- [0010] 일 측면에 있어서, 본 발명에서 제안하는 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이의 동작 방법은 프로그래밍할 부울 논리를 수신하는 단계, 부울 논리의 진리표에 따라 ILP(Integer Linear programming) 알고리즘을 통해 가중치와 임계값을 설정하는 단계, 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정하고, 가중치와 임계값을 컨덕턴스로 변환하는 단계, 부울 논리에 대해 가중치와 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑하는 단계 및 가중치와 임계값 컨덕턴스를 메모리 소자에 전사하는 단계를 포함한다.

발명의 효과

- [0011] 본 발명의 실시예들에 따르면 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조를 통해 가중치와 임계값을 반대 부호를 가진 차동 메모리 소자 쌍으로 구성함으로써 임계값 비교를 위한 별도의 신호 인가를 필요로 하지 않는다. 가중치와 임계값을 서로 반대 방향으로 메모리 어레이의 두 워드라인에 매핑할 경우 벡터-곱 연산 과정에서 가중치와 임계값을 비교할 수 있다. 또한, 가중치와 임계값을 메모리 소자의 상태로 매핑하는 과정에서 소자가 사용 가능한 최소 및 최대 상태를 임계값 논리의 최소 및 최대값에 매핑할 경우 프로그래밍 오차 및 노이즈 신호 문제를 해결할 수 있다.

도면의 간단한 설명

- [0012] 도 1은 본 발명의 일 실시예에 따른 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조를 나타내는 도면이다.
- 도 2는 본 발명의 일 실시예에 따른 부울 대수의 진리표에서 최적의 솔루션을 찾는 ILP 알고리즘의 예시를 나타내는 도면이다.
- 도 3은 본 발명의 일 실시예에 따른 가중치 및 임계값을 메모리 소자가 갖는 최대 및 최소 상태에 따라 조절하여 매핑하는 알고리즘 나타내는 도면이다.
- 도 4는 본 발명의 일 실시예에 따른 부울 논리에 대해 ILP 알고리즘을 이용하여 가중치 및 임계값을 매핑한 결과를 나타내는 도면이다.
- 도 5는 본 발명의 일 실시예에 따른 가중치와 임계값을 사용하여 메모리 소자 어레이에서 부울 논리를 구현한 예시를 나타내는 도면이다.
- 도 6은 본 발명의 일 실시예에 따른 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이를 프로그래밍하는 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

- [0013] 본 발명은 메모리 소자로 구성된 어레이 구조에서 프로그래밍 가능한 임계값 논리 기술을 제안한다. 임계값 논리 기술에서 하나의 소자로 가중치를 매핑할 경우, 가중치와 임계값을 비교하기 위해 별도의 신호 인가가 필요

하다. 그러나 가중치와 임계값을 차동 메모리 소자 쌍으로 구성하면 임계값 비교를 위한 별도의 신호 인가를 필요로 하지 않는다. 가중치와 임계값을 서로 반대 방향으로 메모리 어레이의 두 워드라인에 매핑할 경우 벡터-곱 연산 과정에서 가중치와 임계값을 비교할 수 있다. 또한, 프로그래밍 오차 및 노이즈 신호는 임계값 논리의 동작에 치명적인 영향을 미칠 수 있는데, 가중치와 임계값을 메모리 소자의 상태로 매핑하는 과정에서 소자가 사용 가능한 최소 및 최대 상태를 임계값 논리의 최소 및 최대값에 매핑할 경우 이러한 문제를 해결할 수 있다.

[0014] 본 발명의 실시예에 따르면, 임계값 논리의 가중치와 임계값을 반대 부호를 가진 차동 메모리소자 쌍으로 구성함으로써 논리 구현 범위를 확장하고, 프로그래밍 부정확성 및 노이즈 신호에 대한 허용 오차를 갖도록 최적화된 알고리즘을 제공한다. 따라서, 메모리 내 연산 기술에서 임계값 논리를 활용하는 경우, 효율적이고 안정적인 동작으로 응용될 수 있다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0016] 도 1은 본 발명의 일 실시예에 따른 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조를 나타내는 도면이다.

[0017] 도 1은 메모리 어레이 구조에서 임계 로직을 매핑하는 방법을 보여준다. 메모리 어레이에서 워드라인(wordline)(111, 121)에 전압을 인가하고 비트라인(bitline)($X_1, X_2, \dots, X_{n-1}, X_n$)의 전류를 읽어 벡터, 행렬 곱 연산 수행이 가능하다. 임계값 논리의 가중치와 임계값은 각각 메모리 소자 쌍(110, 120)으로 구성되어 서로 다른 비트라인에 매핑되며, 각각 양($W_1^+, W_2^+, \dots, W_{n-1}^+, W_n^+$)과 음($W_1^-, W_2^-, \dots, W_{n-1}^-, W_n^-$)의 상태를 담당한다. 가중치와 임계값의 메모리 소자 쌍(110, 120)은 서로 다른 부호로 매핑되어, 인접한 두 비트라인의 전류를 별도의 비교 신호 인가 없이 비교할 수 있다. 논리 연산 중에 입력 신호는 0/입력 전압으로 인코딩되어 적용되고 입력 전압은 임계값 장치에 적용된다.

[0018] 본 발명의 실시예에 따른 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이 구조는 전압을 인가하여 비트라인(bitline)의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하는 워드라인(wordline)(111, 121) 및 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되는 비트라인($X_1, X_2, \dots, X_{n-1}, X_n$)을 포함한다.

[0019] 본 발명의 실시예에 따른 메모리 소자 어레이 구조는 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍(110, 120)으로 구성되고, 상기 가중치와 임계값의 차동 메모리 소자 쌍(110, 120)은 서로 다른 부호, 각각 양($W_1^+, W_2^+, \dots, W_{n-1}^+, W_n^+$)과 음($W_1^-, W_2^-, \dots, W_{n-1}^-, W_n^-$)으로 매핑된다.

[0020] 임계값 논리의 가중치와 임계값을 차동 메모리 소자 쌍(110, 120)으로 매핑하게 되면, 구현 가능한 논리를 더욱 확장할 수 있으며 별도의 임계값 비교 신호 인가 없이 메모리 어레이에서의 행렬-곱 연산 과정으로 가중치와 임계값을 비교가 가능하다.

[0021] 본 발명의 실시예에 따른 메모리 소자 어레이 구조는 부울 논리에 대한 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정한다.

[0022] 상기 가중치 및 조정된 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑하여, 상기 가중치 및 조정된 임계값을 컨덕턴스로 변환한다.

[0023] 예를 들어, 가중치 및 임계값을 멤리스터의 컨덕턴스에 매핑하는 과정에서 임계값을 0.5만큼 조정하고 소자 상태가 가질 수 있는 가용 범위의 최소와 최대에 매핑할 경우 프로그래밍오류 및 노이즈 신호의 허용 오차를 가질 수 있다. 이는 프로그래밍가능 임계값 논리의 신뢰할 수 있는 동작 수행을 보장할 수 있다.

[0024] 본 발명의 실시예에 따르면, 상기 가중치와 컨덕턴스를 차동 메모리 소자 쌍으로 구성함으로써 인버터 없이 NOT 연산이 가능하다.

[0026] 도 2는 본 발명의 일 실시예에 따른 부울 대수의 진리표에서 최적의 솔루션을 찾는 ILP 알고리즘의 예시를 나타내는 도면이다.

[0027] 도 2는 부울 대수의 진리표에서 제약 조건(constraint)을 고려하면서 최적의 솔루션을 찾는 정수 선형 프로그래밍(Integer Linear programming: ILP) 알고리즘의 예시를 나타내고, 부울 논리의 진리표에 따라 ILP 알고리즘을 통해 가중치와 임계값을 설정한다.

[0028] 예를 들어, 가중치 및 임계값을 멤리스터의 컨덕턴스에 매핑하는 과정에서 임계값을 0.5만큼 조정하고 소자 상태가 가질 수 있는 가용 범위(possible solutions)의 최소와 최대에 매핑할 경우 프로그래밍오류 및 노이즈 신

호의 최적 허용 오차(optimal solution)를 가질 수 있다. 이는 프로그래밍가능 임계값 논리의 신뢰할 수 있는 동작 수행을 보장할 수 있다.

- [0030] 도 3은 본 발명의 일 실시예에 따른 가중치 및 임계값을 메모리 소자가 갖는 최대 및 최소 상태에 따라 조절하여 매핑하는 알고리즘 나타내는 도면이다.
- [0031] 도 3은 가중치 및 임계값을 메모리 소자가 갖는 최대 및 최소 상태에 따라 조절하여 매핑하는 알고리즘이다. ILP 방식에 의해 특정 논리 연산을 구현하기 위해 필요한 가중치 및 임계값을 설정한 다음, 해당 값을 메모리 소자가 가질 수 있는 상태로 매핑하는 알고리즘이다. 해당 방식은 메모리 소자의 일종인 메모리 소자가 가질 수 있는 전도성의 최대, 최소를 기준으로 진행한 구체적인 일 실시예일뿐, 본 발명은 이에 제한되지 않고, 다양한 소자에 적용될 수 있다. 먼저, 가중치 프로그래밍 부정확성 및 노이즈 신호에 대한 허용 오차를 갖도록 정수 임계값을 0.5만큼 조정한 다음, 가중치 및 조정된 임계값의 최대와 최소를 각각 컨덕턴스의 최댓값과 최솟값에 맞게 스케일링한다.
- [0033] 도 4는 본 발명의 일 실시예에 따른 부울 논리에 대해 ILP 알고리즘을 이용하여 가중치 및 임계값을 매핑한 결과를 나타내는 도면이다.
- [0034] 도 4는 부울 논리에 대해 ILP 알고리즘을 이용하여 가중치 및 임계값을 메모리 소자의 최대(8 μ S) 및 최소(0.4 μ S)에 맞게 매핑한 결과를 요약한 것이다. 모든 부울 논리에 대해 임계값을 0.5만큼 조정하고 컨덕턴스를 최솟값과 최댓값에 매핑함으로써 가중치 프로그래밍 부정확성 및 노이즈에 대한 허용 오차를 최대화하였다. 특히, 가중치와 컨덕턴스를 차동 메모리 소자 쌍으로 구성함으로써 인버터 없이 NOT 연산을 구현할 수 있다. 도 4는 일 실시예일뿐, 본 발명은 이에 제한되지 않고, 다양한 부울 논리 게이트에 적용될 수 있다.
- [0036] 도 5는 본 발명의 일 실시예에 따른 가중치와 임계값을 사용하여 메모리 소자 어레이에서 부울 논리를 구현한 예시를 나타내는 도면이다.
- [0037] 도 5는 결정된 가중치와 임계값을 사용하여 메모리 어레이에서 0.2 μ S의 튜닝 마진으로 4입력 부울 논리를 구현한 예시를 보여준다. 부울 논리가 메모리 어레이 내에서 병렬로 구현되기 때문에 논리 연산이 동시에 수행될 수 있다. 도 5는 일 실시예일뿐, 본 발명은 이에 제한되지 않고, 다양한 부울 논리 게이트에 적용될 수 있다.
- [0039] 도 6은 본 발명의 일 실시예에 따른 프로그래밍 가능한 임계값 논리 구현을 위한 메모리 소자 어레이를 프로그래밍하는 방법을 설명하기 위한 흐름도이다.
- [0040] 본 발명의 실시예에 따른 전압을 인가하여 비트라인(bitline)의 전류를 읽어 벡터, 행렬곱 연산 수행을 수행하는 워드라인(wordline) 및 논리 연산을 위한 입력 신호가 0 또는 입력 전압으로 인코딩되어 입력되는 비트라인을 포함하고, 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되는 메모리 소자 어레이의 동작 방법은 프로그래밍할 부울 논리를 수신하는 단계(610), 부울 논리의 진리표에 따라 ILP(Integer Linear programming) 알고리즘을 통해 가중치와 임계값을 설정하는 단계(620), 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정하고, 가중치와 임계값을 컨덕턴스로 변환하는 단계(630), 부울 논리에 대해 가중치와 임계값을 상기 차동 메모리 소자 상태의 가용 범위의 최소 및 최대 컨덕턴스로 매핑하는 단계(640) 및 가중치와 임계값 컨덕턴스를 메모리 소자에 전사하는 단계(650)를 포함한다.
- [0041] 본 발명의 실시예에 따른 메모리 어레이에서 워드라인(wordline)에 전압을 인가하고 비트라인(bitline)의 전류를 읽어 벡터, 행렬곱 연산 수행이 가능하다. 임계값 논리의 가중치와 임계값은 각각 메모리 소자 쌍으로 구성되어 서로 다른 비트라인에 매핑되며, 각각 양과 음의 상태를 담당한다. 가중치와 임계값의 메모리 소자 쌍은 서로 다른 부호로 매핑되어, 인접한 두 비트라인의 전류를 별도의 비교 신호 인가 없이 비교할 수 있다. 논리 연산 중에 입력 신호는 0/입력 전압으로 인코딩되어 적용되고 입력 전압은 임계값 장치에 적용된다.
- [0042] 먼저 단계(610)에서, 프로그래밍할 부울 논리를 수신한다.
- [0043] 단계(620)에서, 부울 논리의 진리표에 따라 ILP 알고리즘을 통해 가중치와 임계값을 정한다.
- [0044] 본 발명의 실시예에 따른 메모리 소자 어레이 구조는 부울 논리에 대한 가중치와 임계값 각각이 매핑되는 차동 메모리 소자 쌍으로 구성되고, 상기 가중치와 임계값의 차동 메모리 소자 쌍은 서로 다른 부호, 각각 양과 음으로 매핑된다.
- [0045] 임계값 논리의 가중치와 임계값을 차동 메모리 소자 쌍으로 매핑하게 되면, 구현 가능한 논리를 더욱 확장할 수

있으며 별도의 임계값 비교 신호 인가 없이 메모리 어레이에서의 행렬-곱 연산 과정으로 가중치와 임계값을 비교가 가능하다.

- [0046] 단계(630)에서, 임계값을 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 임계값을 0.5만큼 조정하고 본 발명의 알고리즘에 따라 가중치와 임계값을 컨덕턴스로 변환한다.
- [0047] 본 발명의 실시예에 따르면, 부울 논리에 대한 프로그래밍 오류 및 노이즈 신호에 대한 허용 오차를 위해 상기 가중치 및 임계값을 매핑하는 과정에서 임계값을 미리 정해진 크기만큼 조정한다.
- [0048] 예를 들어, 가중치 및 임계값을 멤리스터의 컨덕턴스에 매핑하는 과정에서 임계값을 0.5만큼 조정하고 소자 상태가 가질 수 있는 가용 범위의 최소와 최대에 매핑할 경우 프로그래밍오류 및 노이즈 신호의 허용 오차를 가질 수 있다. 이는 프로그래밍가능 임계값 논리의 신뢰할 수 있는 동작 수행을 보장할 수 있다.
- [0049] 단계(640)에서, 이 방식은 위에서 언급한 바와 같이 모든 부울 논리에 대해 가중치와 임계값을 멤리스터의 최소 및 최대 컨덕턴스로 매핑할 수 있다.
- [0050] 마지막으로 단계(650)에서, 가중치와 임계값 컨덕턴스를 멤리스터 소자에 전사한다.
- [0051] 본 발명의 실시예에 따르면, 프로그래밍가능 임계값 논리를 통해 구현 가능 논리 범위를 확장할 수 있다. 본 발명의 실시예에 따르면, 가중치 및 임계값을 차동 메모리 소자 쌍으로 구성하여 구현 가능한 논리를 더욱 확장하였으며, 제안하는 알고리즘을 통해 프로그래밍 오차 및 노이즈 신호의 허용 오차를 가질 수 있다.
- [0053] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.
- [0054] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 구체화(embodiment)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.
- [0055] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.
- [0056] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형

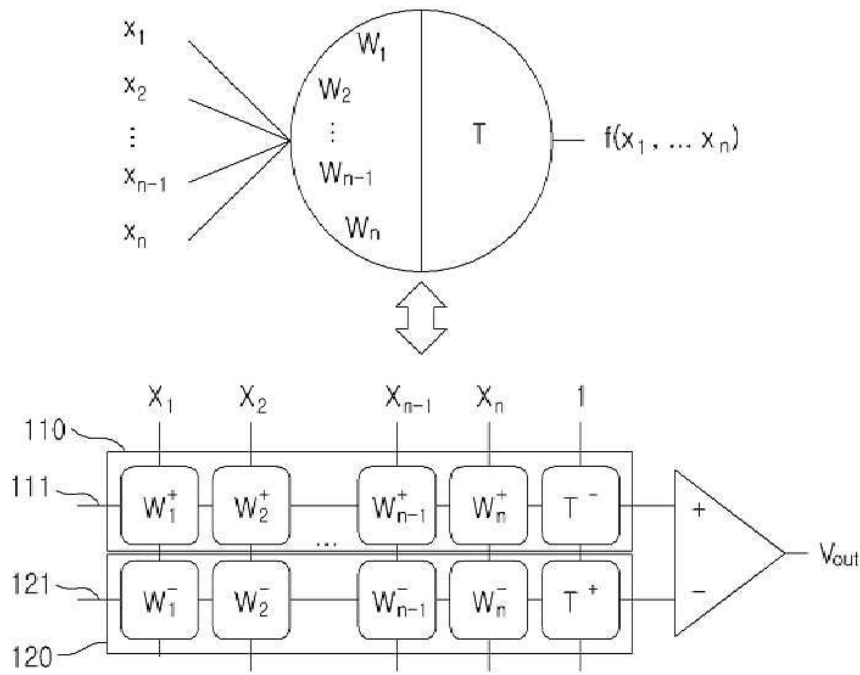
태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

[0057]

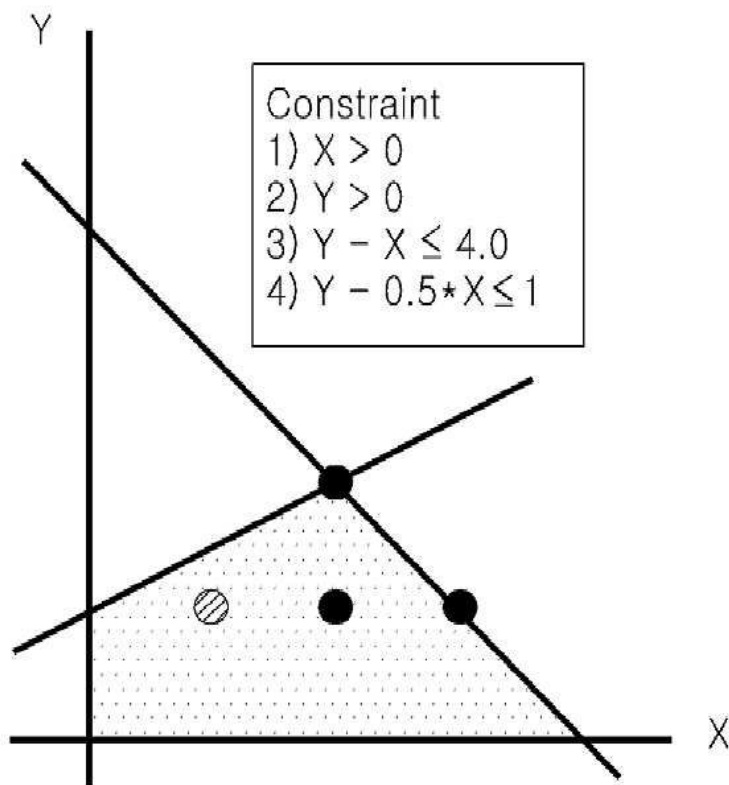
그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

도면1



도면2



Possible solutions

$(x,y) = (1,1), (2,1), (2,2), (3,1)$

Optimal solution

$\min(x,y) = (1,1)$

도면3

Algorithm 1: Weight and Threshold Mapping Algorithm for Memristive Threshold Logic

```

1 Input:
2 //extracted from ILP formulation
3  $W_{i,j} \leftarrow$  Weight value of  $i^{th}$  input for  $j^{th}$  threshold logic gate;
4  $T_j \leftarrow$  Threshold of the  $j^{th}$  threshold logic gate;
5  $G_{min}, G_{max} \leftarrow$  Conductance range of memristor;

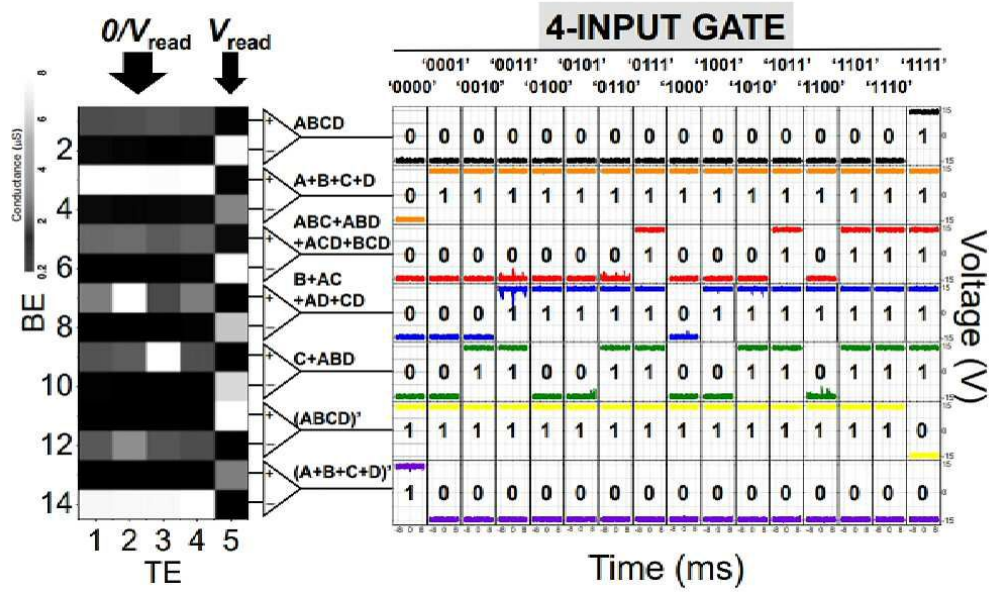
6 Output:
7 //shift T for better weight, threshold margin
8 if  $T_j \geq 0$  then  $T'_j = T_j - 0.5$ ;
9 Else  $T'_j = T_j + 0.5$ ;
10  $WT_{max_j} = \max(abs(W_{i,j}), abs(T'_j)) \leftarrow$  find max value among  $W_{i,j}, T'_j$ ;
11  $\Delta G_j = (G_{max} - G_{min}) \div WT_{max_j} \leftarrow$  unit conductance;
12 //mapping  $W_{i,j}$  to  $W^+$  and  $W^-$ ,  $T'_j$  to  $T^+$  and  $T^-$ 
13 for Each threshold logic gate  $iter_{max}$  do
14   for each input in threshold logic  $inter_{max}$  do
15     if  $W_{i,j} > 0$  then
16        $W_{i,j}^+ = G_{min} + (W_{i,j} \times \Delta G_j)$ ;  $W_{i,j}^- = G_{min}$ ;
17     else if  $W_{i,j} < 0$  then
18        $W_{i,j}^+ = G_{min}$ ;  $W_{i,j}^- = G_{min} + (abs(W_{i,j}) \times \Delta G_j)$ ;
19     else
20        $W_{i,j}^+ = G_{min}$ ;  $W_{i,j}^- = G_{min}$ ;
21     if  $T'_j > 0$  then
22        $T_j^{++} = G_{min}$ ;  $T_j^{--} = G_{min} + (T'_j \times \Delta G_j)$ ;
23     else if  $T'_j < 0$  then
24        $T_j^{++} = G_{min} + (abs(T'_j) \times \Delta G_j)$ ;  $T_j^{--} = G_{min}$ ;
25     else
26        $T_j^{++} = G_{min}$ ;  $T_j^{--} = G_{min}$ ;
27 return  $W_{i,j}^+, W_{i,j}^-, T_j^{++}, T_j^{--}$ 

```

Boolean Function	Weight				Target conductance (s, min = 0.4, max = 8)											
	A	B	C	D	Th	A+	A-	B+	B-	C+	C-	D+	D-	Th+	Th-	
2-Input	AB	1	1	-	-	2	5.45	0.4	5.45	0.4	-	-	-	-	0.4	8
	A+B	1	1	-	-	1	8	0.4	8	0.4	-	-	-	-	0.4	4.2
	(AB) [']	-1	-1	-	-	-2	0.4	5.45	0.4	5.45	-	-	-	-	8	0.4
	(A+B) [']	-1	-1	-	-	-1	0.4	8	0.4	8	-	-	-	-	4.2	0.4
3-Input	ABC	1	1	1	-	3	3.44	0.4	3.44	0.4	3.44	0.4	-	-	0.4	8
	A+B+C	1	1	1	-	1	8	0.4	8	0.4	8	0.4	-	-	0.4	4.2
	AB+BC+CA	1	1	1	-	2	5.45	0.4	5.45	0.4	5.45	0.4	-	-	0.4	8
	AB+AC	2	1	1	-	3	6.5	0.4	3.44	0.4	3.44	0.4	-	-	0.4	8
	C+AB	1	1	2	-	2	4.2	0.4	4.2	0.4	8	0.4	-	-	0.4	6.1
	(ABC) [']	-1	-1	-1	-	-3	0.4	3.44	0.4	3.44	0.4	3.44	-	-	8	0.4
4-Input	(A+B+C) [']	-1	-1	-1	-	-1	0.4	8	0.4	8	0.4	8	-	-	4.2	0.4
	ABCD	1	1	1	1	4	2.57	0.4	2.57	0.4	2.57	0.4	2.57	0.4	0.4	8
	A+B+C+D	1	1	1	1	1	8	0.4	8	0.4	8	0.4	8	0.4	0.4	4.2
	ABC+ABD+ACD+BCD	1	1	1	1	3	3.44	0.4	3.44	0.4	3.44	0.4	3.44	0.4	0.4	8
4-Input	B+AC+AD+CD	1	2	1	1	2	4.2	0.4	8	0.4	4.2	0.4	4.2	0.4	0.4	6.1
	C+ABD	1	1	3	1	3	2.94	0.4	2.94	0.4	8	0.4	2.94	0.4	0.4	6.75
	(ABCD) [']	-1	-1	-1	-1	-4	0.4	2.57	0.4	2.57	0.4	2.57	0.4	2.57	8	0.4
	(A+B+C+D) [']	-1	-1	-1	-1	-1	0.4	8	0.4	8	0.4	8	0.4	8	4.2	0.4

도면4

도면5



도면6

