(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2024년08월21일
(11) 등록번호 10-2696794
(24) 등록일자 2024년08월14일

(51) 국제특허분류(Int. Cl.)
G06F 7/544 (2017.01) *G06F 5/01* (2006.01)
G06F 7/501 (2006.01) *G06F 7/53* (2006.01)
G06F 7/533 (2006.01) *G06N 3/063* (2023.01)
(52) CPC특허분류
G06F 7/5443 (2013.01)
G06F 5/01 (2013.01)
(21) 출원번호 10-2023-0157648
(22) 출원일자 2023년11월14일
심사청구일자 2023년11월14일
(56) 선행기술조사문헌
JP2014126995 A*
JP2016045685 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
인하대학교 산학협력단
인천광역시 미추홀구 인하로 100(용현동, 인하대학교)
(72) 발명자
서영교
경기도 부천시 길주로77번길 19-45, 920호(상동, 디아뜨갤러리4오피스텔)
이래인
대전광역시 유성구 엑스포로 501, 109동 405호(전민동, 청구나래아파트)
(74) 대리인
양성보

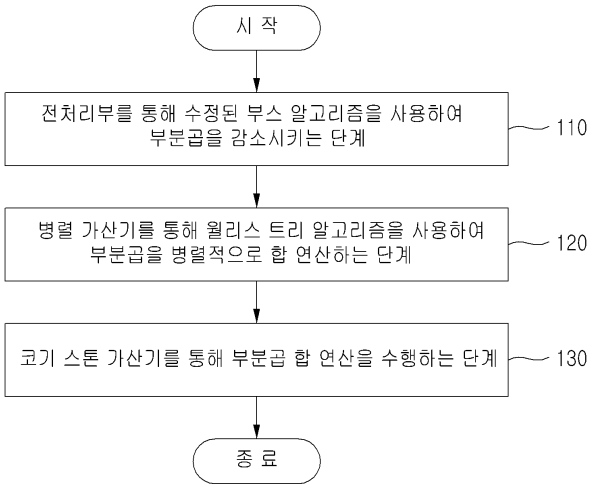
전체 청구항 수 : 총 6 항 심사관 : 지정훈

(54) 발명의 명칭 **고속 병렬 곱셈기 하드웨어 및 그 동작 방법**

(57) 요 약

고속 병렬 곱셈기 및 그 동작 방법이 제시된다. 본 발명에서 제안하는 고속 병렬 곱셈기의 동작 방법은 전처리부를 통해 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 단계, 병렬 가산기(Parallel Adder)를 통해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계 및 코기 스톤 가산기(Koggee Stone Adder)를 통해 상기 부분곱 합(Partial Product Summation) 연산을 수행하는 단계를 포함한다.

대 표 도 - 도1



(52) CPC특허분류
G06F 7/501 (2021.08)
G06F 7/5318 (2013.01)
G06F 7/533 (2013.01)
G06N 3/063 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711194127
과제번호 2021M3F3A2A01037531
부처명 과학기술정보통신부
과제관리(전문)기관명 한국연구재단
연구사업명 신소자원천기술개발
연구과제명 [Ezbaro] Dynamic logic 기반 고성능 STMG 로직 회로 설계
기 여 율 3/10
과제수행기관명 인하대학교
연구기간 2023.01.01 ~ 2023.12.31
이 발명을 지원한 국가연구개발사업
과제고유번호 1415186528
과제번호 20009972
부처명 산업통상자원부
과제관리(전문)기관명 한국산업기술기획평가원
연구사업명 차세대지능형반도체기술개발
연구과제명 고효율 초저전력 경량 엣지 디바이스용 소자회로 및 SoC 개발(4차년도)
기 여 율 3/10
과제수행기관명 인하대학교
연구기간 2023.01.01 ~ 2023.12.31
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 4/10
과제수행기관명 인하대학교 산학협력단
연구기간 2023.01.01 ~ 2023.12.31

공지예외적용 : 있음

명 세 서

청구범위

청구항 1

전처리부를 통해 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 단계;

병렬 가산기(Parallel Adder)를 통해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계; 및

코기 스톤 가산기(Koggee Stone Adder)를 통해 상기 부분곱 합(Partial Product Summation) 연산을 수행하는 단계

를 포함하고,

상기 코기 스톤 가산기를 통해 상기 부분곱 합 연산을 수행하는 단계는,

전처리 과정에서 상기 병렬 가산기로부터 입력 값을 받아 생성 신호 및 전파 신호를 생성하고,

복수의 비트수를 동시에 처리하기 위한 CLA(Carry Look Ahead) 네트워크를 통해 상기 생성 신호 및 전파 신호에 대하여 이전 비트와의 관계에 따른 캐리 값을 구하고,

후처리 과정에서 상기 CLA 네트워크에서 구한 캐리의 값과 상기 전처리 과정에서 구한 전파 신호의 값을 xor한 값을 최종 출력함으로써, CLA 네트워크 기반 코기 스톤 가산기를 통해 캐리 연쇄 연산을 병렬적으로 수행하여 복수의 비트수를 동시에 처리하는

고속 병렬 곱셈기의 동작 방법.

청구항 2

제1항에 있어서,

상기 전처리부를 통해 수정된 부스 알고리즘을 사용하여 부분곱을 감소시키는 단계는,

부분곱을 감소시키기 위해 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 수정된 부스 코딩을 통해 리코드 값을 생성하고 상기 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)하며, 리코드 값이 음수일 때에는 2의 보수를 통해 A를 음수로 바꾸어 연산하는

고속 병렬 곱셈기의 동작 방법.

청구항 3

제1항에 있어서,

상기 병렬 가산기를 통해 월리스 트리 알고리즘을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계는,

CSA(Carry Save Adder)를 통해 캐리(Carry)를 기다리지 않고 중간과정에서 캐리를 보관하여 출력함으로써 캐리의 전파시간을 감소시키고, 복수의 입력 값에 대하여 병렬적으로 합 연산을 반복 수행하여 상기 CSA가 최종적으로 한 개가 되어 출력이 하나만 나올 때까지 결과 값을 더하는

고속 병렬 곱셈기의 동작 방법.

청구항 4

삭제

청구항 5

수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 전처리부;

월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 병렬 가산기

(Parallel Adder); 및

상기 부분곱 합(Partial Product Summation) 연산을 수행하는 코기 스톤 가산기(Koggee Stone Adder)

를 포함하고,

상기 코기 스톤 가산기는,

전처리 과정에서 상기 병렬 가산기로부터 입력 값을 받아 생성 신호 및 전파 신호를 생성하고,

복수의 비트수를 동시에 처리하기 위한 CLA(Carry Look Ahead) 네트워크를 통해 상기 생성 신호 및 전파 신호에 대하여 이전 비트와의 관계에 따른 캐리 값을 구하고,

후처리 과정에서 상기 CLA 네트워크에서 구한 캐리의 값과 상기 전처리 과정에서 구한 전파 신호의 값을 xor한 값을 최종 출력함으로써, CLA 네트워크 기반 코기 스톤 가산기를 통해 캐리 연쇄 연산을 병렬적으로 수행하여 복수의 비트수를 동시에 처리하는

고속 병렬 곱셈기.

청구항 6

제5항에 있어서,

상기 전처리부는,

부분곱을 감소시키기 위해 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 수정된 부스 코딩을 통해 리코드 값을 생성하고 상기 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)하며, 리코드 값이 음수일 때에는 2의 보수를 통해 A를 음수로 바꾸어 연산하는

고속 병렬 곱셈기.

청구항 7

제5항에 있어서,

상기 병렬 가산기는,

CSA(Carry Save Adder)를 통해 캐리(Carry)를 기다리지 않고 중간과정에서 캐리를 보관하여 출력함으로써 캐리의 전파시간을 감소시키고, 복수의 입력 값에 대하여 병렬적으로 합 연산을 반복 수행하여 상기 CSA가 최종적으로 한 개가 되어 출력이 하나만 나올 때까지 결과 값을 더하는

고속 병렬 곱셈기.

청구항 8

삭제

발명의 설명

기 술 분 야

[0001] 본 발명은 고속 병렬 곱셈기 하드웨어 및 그 동작 방법에 관한 것이다.

배 경 기 술

[0002] 최근에 딥러닝 등 대규모의 연산능력이 필요한 어플리케이션이 증가하며 높은 계산 성능이 요구되고 있다. 특히 CNN(Convolutional Neural Networks)은 다양한 계층(layer)이 존재하며 이들 간에 곱셈 연산이 반복되며 데이터를 처리한다. 이에 따라 높은 효율과 성능을 위해서는 고속의 곱셈기(multiplier)가 필수적인 상황이다.

[0003] 곱셈기는 시스템의 전체성능에 큰 영향을 미칠 정도로 중요하다. 곱셈기는 부분곱을 더하는 덧셈이 성능을 결정하므로 가산기(adder)의 속도를 중심으로 설계하면 곱셈기의 성능과 시스템 전체성능을 향상 시킬 수 있다.

선행기술문헌

비특허문헌

[0004] (비특허문헌 0001) [1]M..Motamedi ET AL(2021), "A Survey of FPGA-Based Accelerators for Convolutional Neural Networks", in IEEE Access, Vol.32 no.9, pp84238-84267.doi:10.1109

(비특허문헌 0002) [2] Mannae Renuka, G..Mary Valantina, "3-D graphics of difitla mltiplier whith Kogge-Stone Adder", Anaglog intergrated circuit and signal processing.Vol.113 p343-352,2022

(비특허문헌 0003) [3] 김홍욱(1966), "0.6um COMOS공정을 위한 32×32-bit Parallel Multiplier 설계", 졸업논문(동국대학교 전자공학과)

(비특허문헌 0004) [4] Anurag Sindhu(2012),"8Bit Kogge Stone Adder", EE619 Course Project, https://www.cl.cam.ac.uk/research/srg/han/ACS-P35/8-bit-KoggeStone-Adder.pdf

발명의 내용

해결하려는 과제

[0005] 본 발명이 이루고자 하는 기술적 과제는 곱셈기의 구조를 가속에 맞게 3개를 알고리즘을 기반으로 세 단계로 동작하는 고속 병렬 곱셈기 하드웨어 및 그 동작 방법을 제공하는데 있다. 첫 번째 단계에서는 부분곱(Partial product)을 줄이기 위한 방법으로 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하고, 다음 단계에서는 가산기의 단을 줄이기 위해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 입력(다시 말해, 부분곱)을 3개씩 받아 병렬적으로 덧셈 처리를 하며, 최종곱 결과를 얻는 과정(Partial Product Summation)에서는 월리스 트리 알고리즘의 출력에 대한 캐리 연쇄 연산을 고속덧셈기인 코기 스톤 가산기(Koggee Stone Adder)로 처리한다.

과제의 해결 수단

[0006] 일 측면에 있어서, 본 발명에서 제안하는 고속 병렬 곱셈기의 동작 방법은 전처리부를 통해 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 단계, 병렬 가산기(Parallel Adder)를 통해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계 및 코기 스톤 가산기(Koggee Stone Adder)를 통해 상기 부분곱 합(Partial Product Summation) 연산을 수행하는 단계를 포함한다.

[0007] 상기 전처리부를 통해 수정된 부스 알고리즘을 사용하여 부분곱을 감소시키는 단계는 부분곱을 감소시키기 위해 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 수정된 부스 코딩을 통해 리코드 값을 생성하고 상기 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)하며, 리코드 값이 음수일 때에는 2의 보수를 통해 A를 음수로 바꾸어 연산한다.

[0008] 상기 병렬 가산기를 통해 월리스 트리 알고리즘을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계는 CSA(Carry Save Adder)를 통해 캐리(Carry)를 기다리지 않고 중간과정에서 캐리를 보관하여 출력함으로써 캐리의 전파시간을 감소시키고, 복수의 입력 값에 대하여 병렬적으로 합 연산을 반복 수행하여 상기 CSA가 최종적으로 한 개가 되어 출력이 하나만 나올 때까지 결과 값을 더한다.

[0009] 상기 코기 스톤 가산기를 통해 상기 부분곱 합 연산을 수행하는 단계는 상기 병렬 가산기로부터 입력 값을 받아 생성 신호 및 전파 신호를 생성하고, 상기 생성 신호 및 전파 신호에 대하여 이전 비트와의 관계에 따른 캐리 값을 구하고, 캐리 연쇄 연산을 병렬적으로 수행함으로써 복수의 비트수를 동시에 처리한다.

[0010] 또 다른 일 측면에 있어서, 본 발명에서 제안하는 고속 병렬 곱셈기 하드웨어 시스템은 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 전처리부, 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 병렬 가산기(Parallel Adder) 및 상기 부분곱 합(Partial Product Summation) 연산을 수행하는 코기 스톤 가산기(Koggee Stone Adder)를 포함한다.

발명의 효과

[0011] 본 발명의 실시예들에 따르면 고속 병렬 곱셈기 하드웨어 및 그 동작 방법을 통해 곱셈기의 구조를 가속에 맞게

3개를 알고리즘을 기반으로 세 단계로 동작하도록 한다. 첫 번째 단계에서는 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱(Partial product)을 줄이고, 다음 단계에서는 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 가산기의 단을 줄이고 입력(다시 말해, 부분곱)을 3개씩 받아 병렬적으로 덧셈 처리를 하며, 최종곱 결과를 얻는 과정(Partial Product Summation)에서는 월리스 트리 알고리즘의 출력에 대한 캐리 연쇄 연산을 고속덧셈기인 코기 스톤 가산기(Koggee Stone Adder)로 처리한다.

도면의 간단한 설명

[0012] 도 1은 본 발명의 일 실시예에 따른 고속 병렬 곱셈기 하드웨어 시스템의 구성을 나타내는 도면이다.

도 2는 본 발명의 일 실시예에 따른 고속 병렬 곱셈기의 동작 방법을 설명하기 위한 흐름도이다.

도 3은 본 발명의 일 실시예에 따른 수정된 부스 알고리즘의 비트쌍 리코딩을 나타내는 도면이다.

도 4는 본 발명의 일 실시예에 따른 수정된 부스 알고리즘의 4bit 연산을 설명하기 위한 예시도이다.

도 5는 본 발명의 일 실시예에 따른 월리스 트리 알고리즘의 8 병렬 곱을 설명하기 위한 예시도이다.

도 6은 본 발명의 일 실시예에 따른 4bit CLA 네트워크를 설명하기 위한 도면이다.

도 7은 본 발명의 일 실시예에 따른 16bit 병렬 곱셈기 동작을 설명하기 위한 도면이다.

발명을 실시하기 위한 구체적인 내용

[0013] 곱셈기(Multiplier)는 DSP(Digital Signal Processing System) 시스템, 인공 신경망 등 여러 디지털 신호처리에서 핵심적인 요소로 고속 동작이 요구된다. 본 발명에서는 수정된 부스 알고리즘(Modified Booth Algorithm), 월리스 트리 알고리즘(Wallace Tree Algorithm) 및 코기 스톤 가산기(Koggee Stone Adder)를 이용하여 속도에 핵심을 고속 병렬 곱셈기를 설계하였다.

[0014] 곱셈기의 속도는 부분곱의 덧셈이 속도를 결정하는 주안점으로 부분곱을 줄이는 단계, 상기 부분곱을 병렬 가산기(Parallel Adder)를 통해 더하는 단계와 부분곱 합(Partial Product Summation) 연산을 수행하는 단계로 구성되어있다. 첫 번째 단계에서는 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 절반으로 줄이고, 다음 단계인 월리스 트리 알고리즘(Wallace Tree Algorithm)에서 부분곱을 병렬적으로 더하였다. 마지막 단계에서는 코기 스톤 가산기(Koggee Stone Adder)를 사용하여 부분곱 합(Partial Product Summation) 연산을 처리하였다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0016] 도 1은 본 발명의 일 실시예에 따른 고속 병렬 곱셈기의 동작 방법을 설명하기 위한 흐름도이다.

[0017] 본 발명의 실시예에 따른 고속 병렬 곱셈기의 동작 방법은 전처리부를 통해 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시키는 단계(110), 병렬 가산기(Parallel Adder)를 통해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산하는 단계(120) 및 코기 스톤 가산기(Koggee Stone Adder)를 통해 상기 부분곱 합(Partial Product Summation) 연산을 수행하는 단계(130)를 포함한다.

[0018] 단계(110)에서, 전처리부를 통해 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시킨다.

[0019] 본 발명의 실시예에 따르면, 부분곱을 감소시키기 위해 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 수정된 부스 코딩을 통해 리코드 값을 생성하고 상기 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)하며, 리코드 값이 음수일 때에는 2의 보수를 통해 A를 음수로 바꾸어 연산한다.

[0020] 단계(120)에서, 병렬 가산기(Parallel Adder)를 통해 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산한다.

[0021] 본 발명의 실시예에 따르면, CSA(Carry Save Adder)를 통해 캐리(Carry)를 기다리지 않고 중간과정에서 캐리를 보관하여 출력함으로써 캐리의 전파시간을 감소시키고, 복수의 입력 값에 대하여 병렬적으로 합 연산을 반복 수행하여 상기 CSA가 최종적으로 한 개가 되어 출력이 하나만 나올 때까지 결과 값을 더한다.

[0022] 단계(130)에서, 코기 스톤 가산기(Koggee Stone Adder)를 통해 상기 부분곱 합(Partial Product Summation) 연

산을 수행한다.

[0023] 본 발명의 실시예에 따르면, 상기 병렬 가산기로부터 입력 값을 받아 생성 신호 및 전파 신호를 생성하고, 상기 생성 신호 및 전파 신호에 대하여 이전 비트와의 관계에 따른 캐리 값을 구하고, 캐리 연쇄 연산을 병렬적으로 수행함으로써 복수의 비트수를 동시에 처리한다.

[0025] 도 2는 본 발명의 일 실시예에 따른 고속 병렬 곱셈기 하드웨어 시스템의 구성을 나타내는 도면이다.

[0026] 본 실시예에 따른 고속 병렬 곱셈기 하드웨어 시스템(200)은 프로세서(210), 버스(220), 네트워크 인터페이스(230), 메모리(240) 및 데이터베이스(250)를 포함할 수 있다. 메모리(240)는 운영체제(241) 및 고속 병렬 곱셈 루틴(242)을 포함할 수 있다. 프로세서(210)는 전처리부(211), 병렬가산기(212) 및 KSA 가산기(213)를 포함할 수 있다. 다른 실시예들에서 고속 병렬 곱셈기 하드웨어 시스템(200)은 도 2의 구성요소들보다 더 많은 구성요소들을 포함할 수도 있다. 그러나, 대부분의 종래기술적 구성요소들을 명확하게 도시할 필요성은 없다. 예를 들어, 고속 병렬 곱셈기 하드웨어 시스템(200)은 디스플레이나 트랜시버(transceiver)와 같은 다른 구성요소들을 포함할 수도 있다.

[0027] 메모리(240)는 컴퓨터에서 판독 가능한 기록 매체로서, RAM(random access memory), ROM(read only memory) 및 디스크 드라이브와 같은 비소멸성 대용량 기록장치(permanent mass storage device)를 포함할 수 있다. 또한, 메모리(240)에는 운영체제(241)와 고속 병렬 곱셈 루틴(242)을 위한 프로그램 코드가 저장될 수 있다. 이러한 소프트웨어 구성요소들은 드라이브 메커니즘(drive mechanism, 미도시)을 이용하여 메모리(240)와는 별도의 컴퓨터에서 판독 가능한 기록 매체로부터 로딩될 수 있다. 이러한 별도의 컴퓨터에서 판독 가능한 기록 매체는 플로피 드라이브, 디스크, 테이프, DVD/CD-ROM 드라이브, 메모리 카드 등의 컴퓨터에서 판독 가능한 기록 매체(미도시)를 포함할 수 있다. 다른 실시예에서 소프트웨어 구성요소들은 컴퓨터에서 판독 가능한 기록 매체가 아닌 네트워크 인터페이스(230)를 통해 메모리(240)에 로딩될 수도 있다.

[0028] 버스(220)는 고속 병렬 곱셈기 하드웨어 시스템(200)의 구성요소들간의 통신 및 데이터 전송을 가능하게 할 수 있다. 버스(220)는 고속 시리얼 버스(high-speed serial bus), 병렬 버스(parallel bus), SAN(Storage Area Network) 및/또는 다른 적절한 통신 기술을 이용하여 구성될 수 있다.

[0029] 네트워크 인터페이스(230)는 고속 병렬 곱셈기 하드웨어 시스템(200)을 컴퓨터 네트워크에 연결하기 위한 컴퓨터 하드웨어 구성요소일 수 있다. 네트워크 인터페이스(230)는 고속 병렬 곱셈기 하드웨어 시스템(200)을 무선 또는 유선 커넥션을 통해 컴퓨터 네트워크에 연결시킬 수 있다.

[0030] 데이터베이스(250)는 고속 병렬 곱셈을 위해 필요한 모든 정보를 저장 및 유지하는 역할을 할 수 있다. 도 2에서는 고속 병렬 곱셈기 하드웨어 시스템(200)의 내부에 데이터베이스(250)를 구축하여 포함하는 것으로 도시하고 있으나, 이에 한정되는 것은 아니며 시스템 구현 방식이나 환경 등에 따라 생략될 수 있고 혹은 전체 또는 일부의 데이터베이스가 별개의 다른 시스템 상에 구축된 외부 데이터베이스로서 존재하는 것 또한 가능하다.

[0031] 프로세서(210)는 기본적인 산술, 로직 및 고속 병렬 곱셈기 하드웨어 시스템(200)의 입출력 연산을 수행함으로써, 컴퓨터 프로그램의 명령을 처리하도록 구성될 수 있다. 명령은 메모리(240) 또는 네트워크 인터페이스(230)에 의해, 그리고 버스(220)를 통해 프로세서(210)로 제공될 수 있다. 프로세서(210)는 전처리부(211), 병렬가산기(212) 및 KSA 가산기(213)를 위한 프로그램 코드를 실행하도록 구성될 수 있다. 이러한 프로그램 코드는 메모리(240)와 같은 기록 장치에 저장될 수 있다.

[0032] 전처리부(211), 병렬가산기(212) 및 KSA 가산기(213)는 도 1의 단계들(110~130)을 수행하기 위해 구성될 수 있다.

[0033] 고속 병렬 곱셈기 하드웨어 시스템(200)은 전처리부(211), 병렬가산기(212) 및 KSA 가산기(213)를 포함할 수 있다.

[0034] 본 발명의 실시예에 따른 전처리부(211)는 수정된 부스 알고리즘(Modified Booth Algorithm)을 사용하여 부분곱을 감소시킨다.

[0035] 본 발명의 실시예에 따른 전처리부(211)는 부분곱을 감소시키기 위해 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 수정된 부스 코딩을 통해 리코드 값을 생성하고 상기 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)하며, 리코드 값이 음수일 때에는 2의 보수를 통해 A를 음수로 바꾸어 연산한다.

[0036] 본 발명의 실시예에 따른 병렬가산기(212)는 월리스 트리 알고리즘(Wallace Tree Algorithm)을 사용하여 상기 부분곱을 병렬적으로 합 연산을 수행한다.

[0037] 본 발명의 실시예에 따른 병렬가산기(212)는 CSA(Carry Save Adder)를 통해 캐리(Carry)를 기다리지 않고 중간 과정에서 캐리를 보관하여 출력함으로써 캐리의 전파시간을 감소시키고, 복수의 입력 값에 대하여 병렬적으로 합 연산을 반복 수행하여 상기 CSA가 최종적으로 한 개가 되어 출력이 하나만 나올 때까지 결과 값을 더한다.

[0038] 본 발명의 실시예에 따른 KSA 가산기(213)는 상기 부분곱 합(Partial Product Summation) 연산을 수행한다.

[0039] 본 발명의 실시예에 따른 KSA 가산기(213)는 상기 병렬 가산기로부터 입력 값을 받아 생성 신호 및 전파 신호를 생성하고, 상기 생성 신호 및 전파 신호에 대하여 이전 비트와의 관계에 따른 캐리 값을 구하고, 캐리 연쇄 연산을 병렬적으로 수행함으로써 복수의 비트수를 동시에 처리한다.

[0041] 도 3은 본 발명의 일 실시예에 따른 수정된 부스 알고리즘의 비트쌍 리코딩을 나타내는 도면이다.

[0042] 곱셈기는 크게 부분곱과 이 부분곱을 합하는 가산기 부분으로 나눌 수 있다. 곱셈기는 이 두 부분에 어떠한 알고리즘을 적용하냐에 따라 그 성능 결정된다.

[0043] 본 발명의 실시예에 따른 수정된 부스 알고리즘(Modified Booth Algorithm)은 부분곱의 수를 줄이기 위한 방식이다.

[0044] 수정된 부스 알고리즘을 적용하기 위해서는 A, B의 곱을 가정했을 때 B의 LSB 오른쪽에 0을 추가하고 3bit씩 묶어 도 3과 같은 수정된 부스 알고리즘을 통해 리코드(recode) 값을 갖는다. 리코드 값이 0일때에 부분곱은 0이며 +1일 때에는 A, +2일 때 때에는 A를 레프트 쉬프트(left shift)한다. 그리고 리코드가 음수일 때에는 2의 보수(two's complement)를 통해 A를 음수로 바꿔 연산 한다.

[0045] 또한, 부분곱에서 다음 단으로 갈 때에는 기존의 곱셈기에서 1번 쉬프트(shift)되는 것과 다르게 2번 쉬프트되어 부분곱의 개수가 절반으로 줄어들게 된다.

[0047] 도 4는 본 발명의 일 실시예에 따른 수정된 부스 알고리즘의 4bit 연산을 설명하기 위한 예시도이다.

[0048] 본 발명의 일 실시예에 따르면, 예를 들어 A=0110, B=0010 일 때, B는 $\underline{00\overline{10(0)}}$로 3 bit씩 묶어 리코드 -2, +1을 얻고 도 4와 같은 연산을 거치게 된다.

[0049] 부분곱의 왼쪽은 부호 확장(sign extension)으로 채우고 오른쪽은 쉬프트를 진행에 비어진 곳이므로 0으로 채워 8bit로 만든다 원래 4bit 곱셈의 부분곱은 8bit이 4단 생기지만 2단으로 줄어들었다.

[0051] 도 5는 본 발명의 일 실시예에 따른 월리스 트리 알고리즘의 8 병렬 곱을 설명하기 위한 예시도이다.

[0052] 본 발명의 실시예에 따른 CSA(Carry Save Adder)은 FA(Full Adder)와 다르게 이전 캐리(Carry)를 기다리지 않고 중간과정에서 캐리를 보관 하여 출력한다. 따라서 캐리의 전파시간을 줄일 수 있다. 세 개의 입력을 넣어 병렬적으로 처리하였고 CSA가 한 개가 되어 출력(sum, cout)이 하나씩만 나올 때까지 결과 값을 더한다.

[0054] 도 6은 본 발명의 일 실시예에 따른 4bit CLA 네트워크를 설명하기 위한 도면이다.

[0055] 본 발명의 실시예에 따른 코기 스톤 가산기(Koggee Stone Adder)는 CLAA(Carry Look Ahead Adder)를 기반으로 변형한 것으로 전처리 과정(pre-processing)(610), CLA 네트워크(620), 후처리 과정(post-processing)이 있다. 이 과정을 통해 캐리의 이동 단을 줄여 시간복잡도를 $O(\log n)$으로 줄일 수 있다.

[0056] 본 발명의 실시예에 따른 전처리 과정(610)에서, 입력을 받아 다음과 같은 생성(generate)과 전파(propagate) 신호를 만든다:

[0057]
$$p[i] = A[i] \, xor \, B[i]$$
$$g[i] = A[i] \, \text{and} \, B[i]$$

[0058] p[i],g[i]는 본 발명의 실시예에 따른 전처리 과정에서 만들어지는 생성(generate)과 전파(propagate) 값이다.

[0059] 도 6을 참조하면, 본 발명의 실시예에 따른 CLA 네트워크 구조(620)와 같이 이전 비트와의 관계를 고려하여 다음과 같이 캐리를 구한다:

[0060] $$P[i] = p \text{ and } p_{pre}$$
$$G[i] = g \text{ or } (p \text{and} g_{pre})$$

[0061] 여기서, p, g는 직전에서 전파된 p, g이며 p_{pre} , g_{pre} 는 네트워크에 따라 받는 p, g 값이다.

[0062] 본 발명의 실시예에 따르면, 병렬적으로 동작하기에 많은 비트수를 동시에 처리할 수 있다.

[0063] 본 발명의 실시예에 따른 후처리 과정에서 CLA 네트워크에서 구한 캐리의 값과 전처리 과정에서 구한 p의 값을 xor한 값이 최종 Sum이고, 다음과 같이 나타낼 수 있다:

[0064] $$S[i] = P[i] \chi \text{or} C[i-1]$$

[0066] 도 7은 본 발명의 일 실시예에 따른 16bit 병렬 곱셈기 동작을 설명하기 위한 도면이다.

[0067] 본 발명의 일 실시예에 따른 16×16 곱셈기는 도 7과 같은 블록 다이어그램을 가진다. 16bit 입력(711)을 받아 수정된 부스 알고리즘(Modified Booth Algorithm)(710)에 의해 8개의 32bit 부분곱(Partial product)(712)를 만들고 월리스 트리 알고리즘(Wallace Tree Algorithm)(720)을 통해 32bit의 출력(sum,cout)을 만든다. 마지막으로 코기 스톤 가산기(Koggee Stone Adder)(730)를 통해 최종 Output을 출력 한다.

[0068] 본 발명에서 제안하는 알고리즘에 의한 곱셈기와 종래기술에 따른 곱셈기의 비교가 표 1에 나와 있다.

[0069] <표 1>

[0070]

	Parallel Multiplier	Basic Multiplier
Levels	10	14
Routes	11	15
Total Delay	6.753	7.937
Logic Delay	0.916	1.133
Net Delay	5.837	6.804

[0071] Vivado 2018.3 버전에서 곱셈를 설계하여 구현을 통해 곱셈기의 지연 시간 등을 측정하여 성능을 비교하였다.

[0072] 비교를 위한 기본 곱셈기(Basic Multiplier)는 16개의 부분곱을 전가산기를 통해 더한 것이다. 시뮬레이션 결과는 Worst case의 값으로 측정하였으며 표3에서 보인 것과 같이 레벨(Level)과 라우팅 경로(routes)가 더 짧고 지연(Delay) 시간이 향상되었다. 레벨은 회로의 깊이(deapth)라 불리는 것으로 작을수록 빠르게 동작하며 라우팅 경로는 신호가 한곳에서 다른 곳으로 이동하는 경로로 작을수록 짧은 경로를 갖는다. 병렬 곱셈기(Parallel Multiplier의 levels)의 라우팅 경로가 더 짧으므로 로직 지연(logic delay), 넷 지연(net delay)이 더 낮아 전체 지연(Total Delay)이 더 작게 나타나는 것을 확인할 수 있다. 따라서 일반 곱셈기보다 제안하는 곱셈기가 더 빠르고 짧은 경로로 동작하는 것을 알 수 있다.

[0074] 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.

[0075] 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(embody)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.

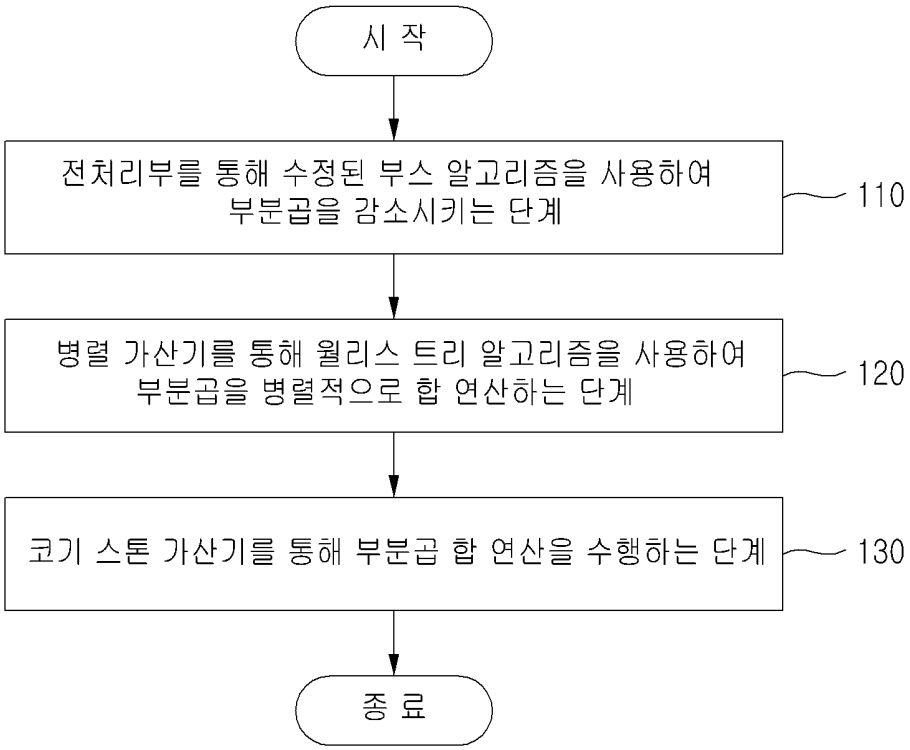
[0076] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic 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.

[0077] 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 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

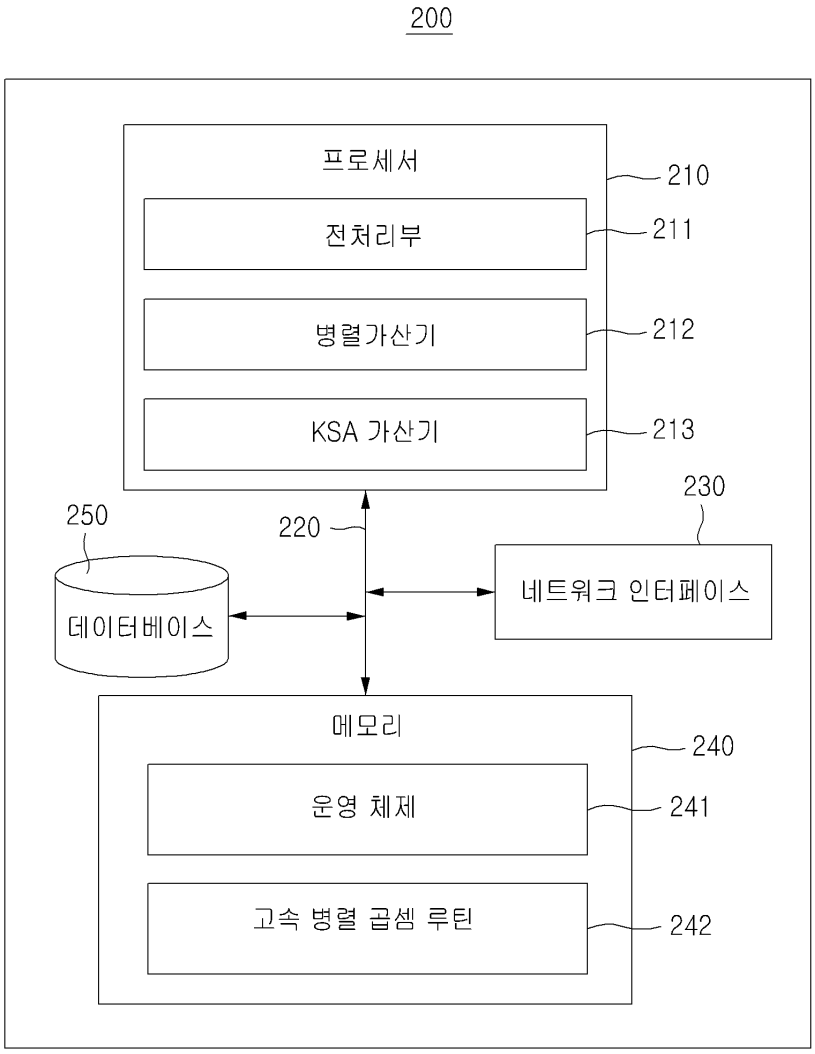
[0078] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

도면1



도면2


200
프로세서
210
전처리부
211
병렬가산기
212
KSA 가산기
213
230
250
220
네트워크 인터페이스
데이터베이스
메모리
240
운영 체제
241
고속 병렬 곱셈 루틴
242

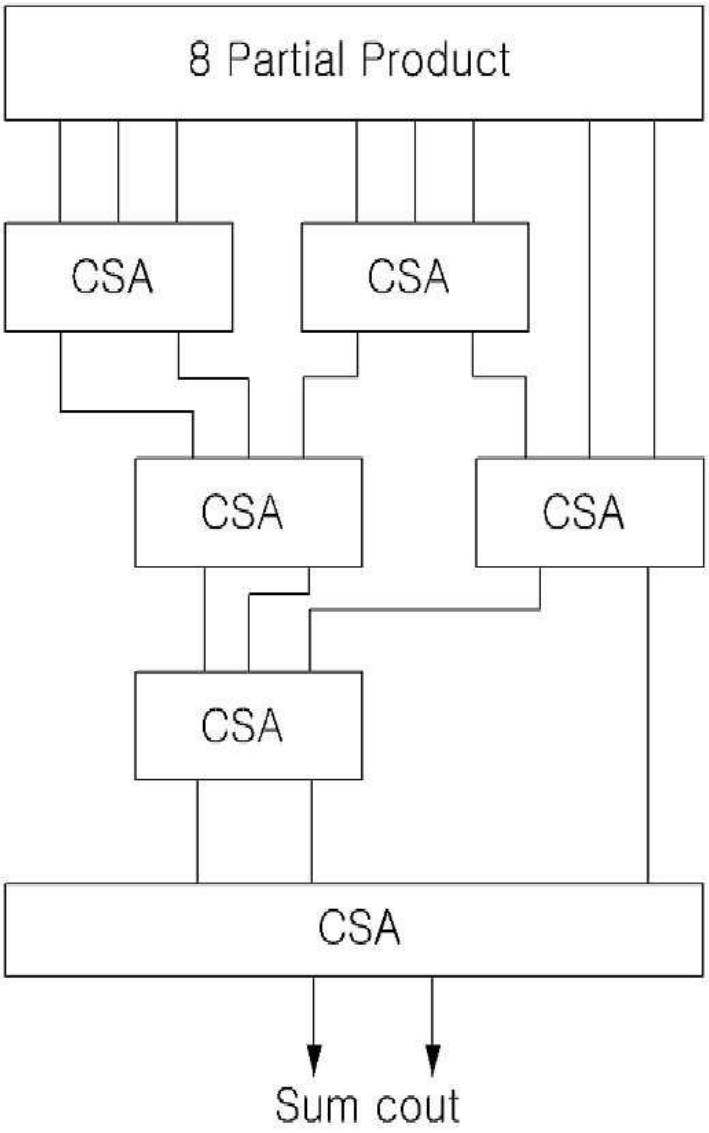
도면3

B_{i+1}	B_i	B_{i-1}	recode	operation
0	0	0	0	0
0	0	0	+1	+A
0	1	1	+1	+A
0	1	1	+2	+2A
1	0	0	-2	-2A
1	0	1	-1	-A
1	1	0	-1	-A
1	1	1	0	0

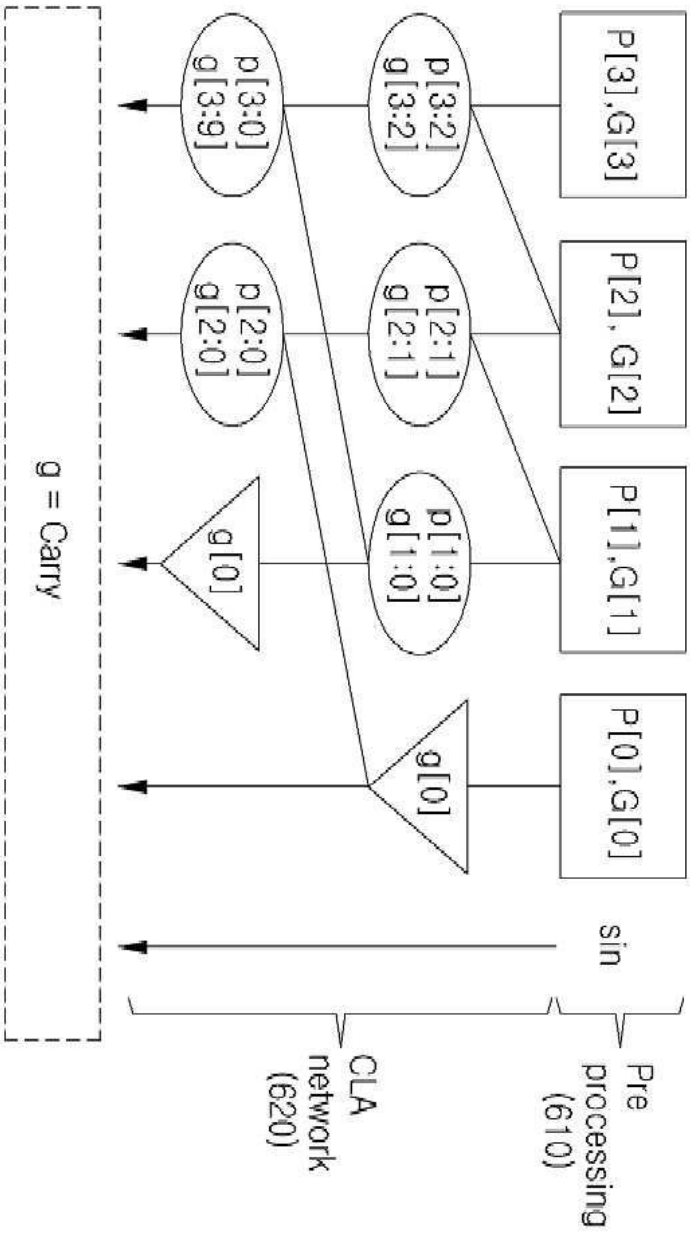
도면4

B=1110 A=0110		
Partial Product (recode)		
1111	1010	(-2)
0001	1000	(+1)

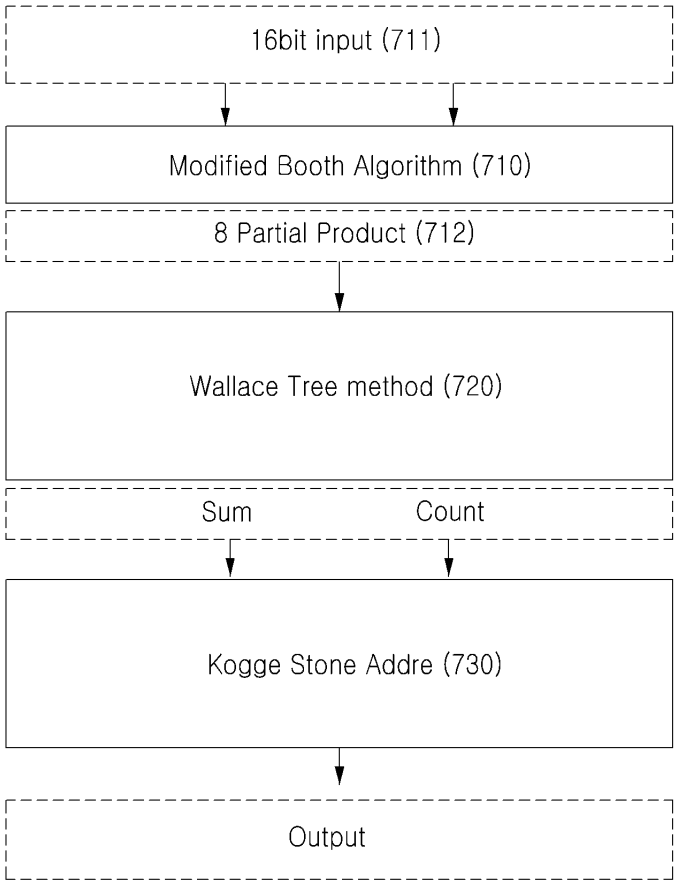
도면5



도면6


P[3],G[3]
P[2], G[2]
P[1],G[1]
P[0],G[0]
sin
p[3:2]
g[3:2]
p[2:1]
g[2:1]
p[1:0]
g[1:0]
g[0]
p[3:0]
g[3:9]
p[2:0]
g[2:0]
g[0]
g = Carry
Pre
processing
(610)
CLA
network
(620)

도면7


16bit input (711)
Modified Booth Algorithm (710)
8 Partial Product (712)
Wallace Tree method (720)
Sum
Count
Kogge Stone Addre (730)
Output