



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년07월13일
(11) 등록번호 10-1878400
(24) 등록일자 2018년07월09일

(51) 국제특허분류(Int. Cl.)
H03H 17/06 (2006.01) G06F 17/10 (2006.01)
H03H 21/00 (2006.01)
(52) CPC특허분류
H03H 17/0621 (2013.01)
G06F 17/10 (2013.01)
(21) 출원번호 10-2017-0016179
(22) 출원일자 2017년02월06일
심사청구일자 2017년02월06일
(56) 선행기술조사문헌
"An Enhanced Low-Power High-Speed Adder For Error-Tolerant Application", Proceeding of the 12th International Symposium on Integrated Circuits (ISIC), 2009
"Statistical analysis and modeling for error composition in approximate computation circuits", IEEE 31st International Conference on Computer Design (ICCD), 2013

(73) 특허권자
울산과학기술원
울산광역시 울주군 언양읍 유니스트길 50
(72) 발명자
강석형
울산광역시 울주군 언양읍 유니스트길 50
강예성
울산광역시 울주군 언양읍 유니스트길 50
(74) 대리인
전용준

전체 청구항 수 : 총 10 항

심사관 : 최규돈

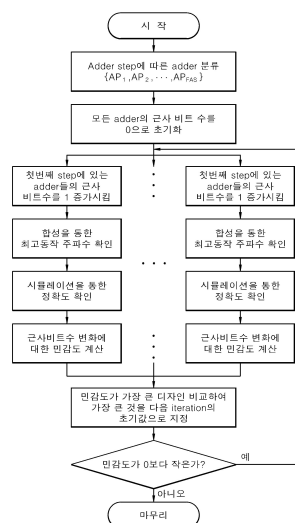
(54) 발명의 명칭 근사연산을 이용한 FIR필터 연산방법

(57) 요약

본 발명은 FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계 및 FIR필터 덧셈기를 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되, 연산 알고리즘은, 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 교체부분의 숫자열을 근사 교체하는 근사연산을 이용한 FIR필터 연산방법을 제공한다.

상기한 바에 따르면 FIR 필터에서 기존의 가산/감산기(adder/subtractor)를 자동화된 합성 flow를 가진 가산/감산법으로 대체함으로써 에너지 소비를 줄일 수 있다.

대표도 - 도2



(52) CPC특허분류
H03H 21/0043 (2013.01)

공지예외적용 : 있음

명세서

청구범위

청구항 1

FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계; 및

상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되,

상기 연산 알고리즘은,

상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체하고,

상기 연산 알고리즘은,

수학식 1에 의하여 정확도가 정의되는 근사연산을 이용한 FIR필터 연산방법.

수학식 1

$$accuracy = \min_{k=1, \dots, M} \left(1 - \frac{|result_k - ref_k|}{|ref_k|} \right) \times 100\% \quad (1)$$

여기서, $result_k$ 는 k번째 입력 패턴으로부터 생성되는 근사 결과 이고 ref_k 는 옳은 결과를 나타낸다.

청구항 2

청구항 1에 있어서,

상기 연산 알고리즘은,

상기 정확부분과 상기 교체부분을 구분하는 경계위치를 동일 애더 스텝(Adder step)에 대하여 동일하게 설정하는 근사연산을 이용한 FIR필터 연산방법.

청구항 3

청구항 2에 있어서,

상기 경계위치는,

각 스텝에서 저에너지를 기준으로 설정하는 근사연산을 이용한 FIR필터 연산방법.

청구항 4

청구항 2에 있어서,

상기 경계위치는,

이전 스텝에서 결정된 경계위치를 이용하는 근사연산을 이용한 FIR필터 연산방법.

청구항 5

삭제

청구항 6

FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계; 및

상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되,

상기 연산 알고리즘은,

상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체하고,

상기 연산 알고리즘은,

수학식 2에 의하여 민감요소(SF)가 정의되는 근사연산을 이용한 FIR필터 연산방법.

수학식 2

$$SF = \begin{cases} \frac{accuracy - accuracy_{min}}{delay} & , \text{if } accuracy > accuracy_{min} \\ 0 & , \text{else} \end{cases} \quad (2)$$

청구항 7

FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계; 및

상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되,

상기 연산 알고리즘은,

상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체하고,

상기 연산 알고리즘은,

영상 출력의 질을 위하여 최대 신호대비 잡음비(PSNR)는 수학식 3에 의하여 정의되는 근사연산을 이용한 FIR필터 연산방법.

수학식 3

$$PSNR = 10 \times \log\left(\frac{255^2}{\sigma_{noise}^2}\right) \quad (3)$$

여기서, σ_{noise}^2 는 오차의 분산을 의미한다.

청구항 8

FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계; 및

상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되,

상기 연산 알고리즘은,

상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체하고,

상기 연산 알고리즘에 따는 상기 합성작업은,

에더 스텝(adder step)에 대응하여 각각의 에더(adder)들로 분류하는 단계와,

분류된 상기 에더들의 근사 비트수를 초기화하는 단계와,

상기 각 에더들을 합성하여 민감도와 정확도를 계산하는 단계와,

상기 민감도와 정확도 계산을 통한 각 디자인과 비교하여 민감도가 가장 큰 디자인의 인터랙션(interaction)을 초기값으로 지정하는 단계와,

상기 민감도를 설정된 값과 비교 판단하는 단계를 포함하는 근사연산을 이용한 FIR필터 연산방법.

청구항 9

청구항 8에 있어서,

상기 분류된 상기 애더들의 근사 비트수를 초기화하는 단계에서,

분류된 상기 애더의 근사 비트수는 "0"으로 초기화하는 근사연산을 이용한 FIR필터 연산방법.

청구항 10

청구항 8에 있어서,

상기 민감도를 설정된 값과 비교 판단하는 단계는,

상기 민감도가 0보다 작은 경우 상기 상기 각 애더들을 합성하여 민감도와 정확도를 계산하는 단계를 다시 수행하는 근사연산을 이용한 FIR필터 연산방법.

청구항 11

청구항 8에 있어서,

상기 각 애더를 합성하는 단계는,

첫 번째 스텝에 있는 애더들의 근사 비트수를 "1" 증가시키는 단계와,

합성을 통한 최고 동작 주파수를 확인하는 단계와,

시뮬레이션을 통한 정확도를 확인하는 단계와,

근사 비트수 변화에 대한 민감도를 계산하는 단계를 포함하는 근사연산을 이용한 FIR필터 연산방법.

발명의 설명

기술 분야

[0001] 본 발명은 근사연산을 이용한 FIR필터 연산방법에 관한 것으로서, 보다 상세하게는 영상처리 어플리케이션 등에 적용되는 FIR필터를 근사화시켜 전력소모를 줄일 수 있는 근사연산을 이용한 FIR필터 연산방법에 관한 것이다.

배경 기술

[0002] 최근 반도체 기술 개발이 지속되면서, 전자 기기는 더 작고 더 이동이 간편해져 가고 있다. 결과적으로 IoT(Internet of Things) 기기들의 배터리 크기가 작아지고 전력소비는 늘어나, 에너지 효율 시스템에 대한 수요가 절박해지면서 근사 연산(approximate computing)기술에 대한 관심이 증가했다.

[0003] 한편, FIR 필터는 IIR 필터 보다 더 나은 선형성과 안정성을 나타낸다. 그러나 복잡한 구조로 인해 전력을 더 많이 소모하므로, 전체 시스템 에너지 효율은 떨어진다. 이에 FIR 필터의 에너지 효율을 증가시키기 위하여, 그 설계 복잡도를 줄이기 위한 기술이 개발되고 있으나 이러한 접근들은 추가 단계의 개수를 줄이고, 정확한 평가 모델을 제공하거나, 별도의 근사 근사가산기(approximate adder)를 개발하는 데에만 집중하고 있는 실정이다.

[0004] 도 1은 일반적인 FIR 필터의 MAC(multiply-and-accumulate) 구조를 보여준다. FIR 필터의 복잡도를 줄이기 위해 곱셈기를 쉬프터(shifter)와 애더(adder)로 대체한 FIR 필터가 일반적으로 사용된다. 먼저 정수 계수들을 shift와 addition연산에 적합한 형태로 변형한다. 부호화 거듭제곱 (SPT)을 사용하면 "0" 아닌 숫자의 개수를 줄일 수 있기 때문에, 모든 계수가 부호화 이진 보다는 부호화 거듭제곱 (SPT)으로 표현된다. SPT 코드 중 CSD(canonical signed digit) 코드는 FIR 필터의 복잡도를 효과적으로 줄이는 것으로 잘 알려져 있다.

[0005] CSE(common subexpression elimination) 알고리즘은 FIR 필터의 복잡도를 줄이기 위해 사용되는 또 다른 기법 중 하나이다. Chia et al.은 CSD 계수들 사이에서 중복입력을 줄이기 위하여 CSE 알고리즘을 제안한 바 있다. Choi et al.은 FIR 필터의 각 계수의 정확도의 미치는 영향력을 분석하여 CSE 알고리즘 동안 더 중요한 계수에 더 제약을 가한다. Choi의 FIR 필터는 더 적은 통과대역/정지대역의 파문으로 더 낮은 전압으로 25~30% 전력 소모 절약을 출력한다. Kahng et al.는 회로 합성 레벨에서 근사화(approximation)를 이용하여 FIR 필터를 시행했다. 특정 모듈을 출력물의 질에는 손상을 덜 주면서 전력 소비만을 줄이기 위하여 순람표에 기초한 근사 모듈로

대체하였으며 Gupta et al.는 근사 회로를 이용하여 FIR 필터를 시행한 바 있다.

[0006] 하지만 Chia et al.과 Malcolm et al.은 근사 가산기의 에러와 전력 소비를 위하여 수학적 모델에 대하여 단지 애더 스텝(adder step)의 개수를 줄이는데 집중한 바 있다. 또한, Choi et al.은 전력을 절약하기 위한 전압 규모 조정을 고려하나 critical path를 따라 일어난 에러가 보통 근사화로 인해 발생하는 에러보다 더 치명적이다. Kahng et al.과 Gupta et al.은 근사 연산을 FIR 필터에는 적용하였으나 근사화를 위한 어떠한 자동화된 합성 플로우(flow)도 제공하지 못하였다. FIR 필터의 설계 규모가 더 커지면, 근사 가산기의 최적 배치를 찾기가 어려워진다.

[0007] 상기한 바와 같이 종래의 FIR필터의 연산방법은 근사화 때문에 일어난 에러에 의하여 정확도가 떨어지고 전력도 많이 소모되는 문제점이 있었다.

선행기술문헌

특허문헌

[0008] (특허문헌 0001) 대한민국 등록특허공보 제10-0294541호

발명의 내용

해결하려는 과제

[0009] 본 발명은, 영상처리 어플리케이션 등에 적용되는 FIR필터를 근사화시켜 허용되는 범위의 정확성을 가지면서도 전력소모를 줄일 수 있는 근사연산을 이용한 FIR필터 연산방법을 제공하는 것을 목적으로 한다.

과제의 해결 수단

[0010] 본 발명은 FIR필터의 덧셈기를 근사연산 덧셈기로 교체하는 단계 및 상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하는 단계를 포함하되, 상기 연산 알고리즘은, 상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체하는 근사연산을 이용한 FIR필터 연산방법을 제공한다.

발명의 효과

[0011] 본 발명에 따른 근사연산을 이용한 FIR필터 연산방법은 다음과 같은 효과를 제공한다.

[0012] 첫째, FIR 필터에서 기존의 가산/감산기(adder/subtractor)를 자동화된 합성 플로우(flow)를 가진 가산/감산법으로 대체함으로써 에너지 소비를 줄일 수 있다.

[0013] 둘째, 전력소모를 줄여 에너지 효율적이고 적정 허용 범위 내 정확성을 향상시킬 수 있도록 하여 효율적이며, 가산/감산의 구성 때문에 생기는 최대 에러는 출력 질을 측정하기 위해서 분석된다.

[0014] 셋째, 근사 합성 플로우(flow)를 사용하여, 에너지/전력 소비를 절약하고 정확도를 합리적인 수준으로 유지하기 위한 기능을 개선할 수 있다.

[0015] 넷째, 최근 컴퓨터 시스템들의 이동성이 중요해 지면서, 부품들의 전력소비를 더 감소시키는 것이 필요한 바, 근사 연산을 이용하면 전력소비를 줄일 수 있다

[0016] 다섯째, 유한 임펄스 응답(FIR)필터의 에너지 효율을 근사 연산을 통하여 개선할 수 있으며, 허용할 만한 정확도를 가진 고효율 FIR 필터용 근사 합성 기술을 제공할 수 있다.

[0017] 여섯째, 공통 부분식의 삭제(CSE) 알고리즘을 사용하여 FIR 필터를 시행하고 기존의 덧셈 뺄셈을 근사의 덧셈 뺄셈으로 대체하여, 수용할 만한 정확률은 출력하면서도 기존 FIR 필터 설계와 비교하여 50.7%의 에너지 소비를 줄일 수 있다.

도면의 간단한 설명

[0018] 도 1은 종래의 FIR 필터의 MAC(multiply-and-accumulate) 구조를 나타내는 도면이다.

도 2 및 도 3은 본 발명의 실시예에 따른 근사연산을 이용한 FIR필터 연산방법을 나타내는 절차도이다.

도 4는 FIR필터의 회로도이다.

도 5는 FIR필터의 상세특징을 나타내는 회로도이다.

도 6은 근사 합성 플로우의 반복적인 결과를 나타내는 그래프이다.

도 7은 최적화된 FIR필터를 이용하여 필터화된 이미지들을 나타내는 사진이다.

발명을 실시하기 위한 구체적인 내용

[0019] 이하, 첨부된 도면을 참조하여 본 발명의 바람직한 실시예를 상세히 설명하기로 한다.

[0020] 먼저, 도 2 및 도 3을 참조하면, 본 발명의 실시예에 따른 근사연산을 이용한 FIR필터 연산방법은, FIR필터의 덧셈기를 근사연산 덧셈기로 교체하되, 상기 FIR필터 덧셈기를 상기 근사연산 덧셈기로 교체 시 설정된 연산 알고리즘에 따라 합성작업을 하도록 되어 있다.

[0021] 여기서, 상기 연산 알고리즘은, 상기 근사 연산덧셈기의 숫자열을 정확부분(accurate part)과 교체부분(inaccurate part)로 구분하여, 상기 교체부분의 숫자열을 근사 교체한다.

[0022] 상기 연산 알고리즘은, 상기 정확부분과 상기 교체부분을 구분하는 경계위치를 동일 스텝(step)에 대하여 동일하게 설정한다. 여기서, 상기 경계위치는, 상기 각 스텝에서 저에너지를 기준으로 설정한다. 또한, 상기 경계위치는, 이전 스텝에서 결정된 경계위치를 이용한다. 여기서, 에너지는 전력소모 딜레이 (Power delay product, PDP)곱을 의미하는 것으로서, 단위연산 당 에너지 소모량으로도 해석할 수 있다. 때문에 상기 저에너지는 상대적으로 작은 양의 에너지를 사용하였다는 의미이다.

[0023] 상기 연산 알고리즘은, 수학적 식 1에 의하여 정확도가 정의된다.

[0024] 수학적 식 1

$$accuracy = \min_{k=1, \dots, M} \left(1 - \frac{|result_k - ref_k|}{|ref_k|} \right) \times 100\% \quad (1)$$

[0026] 또한, 상기 연산 알고리즘은, 수학적 식 2에 의하여 민감요소(SF)가 정의된다.

[0027] 수학적 식 2

$$SF = \begin{cases} \frac{accuracy - accuracy_{min}}{delay} & , \text{if } accuracy > accuracy_{min} \\ 0 & , \text{else} \end{cases} \quad (2)$$

[0029] 또한, 상기 연산 알고리즘에서, 영상 출력의 질 측정을 위한 최대 신호대비 잡음비(PSNR)은 수학적 식 3에 의하여 정의된다.

[0030] 수학적 식 3

$$PSNR = 10 \times \log \left(\frac{255^2}{\sigma_{noise}^2} \right) \quad (3)$$

[0032] 여기서, σ_{noise}^2 는 오차의 분산을 의미한다.

[0033] 한편, 상기 연산 알고리즘에 따른 상기 합성작업에 대하여 살펴보면, 먼저 애더 스텝(adder step)에 대응하여 정확도를 적정 기준 이상 유지하면서 에너지 소모를 최대한으로 줄이는 최적의 AP 배열을 이루도록 각각의 애더(adder)를 애더 스텝(adder step)별로 분류하고, 분류된 상기 애더들의 근사 비트수를 초기화한 후 상기 각 애더들을 합성하여 민감도와 정확도를 계산하고 이러한 민감도와 정확도 계산을 통한 각 디자인과 비교하여 민감도가 가장 큰 디자인의 인터렉션(interaction)을 다음 루프의 초기값으로 지정하여 상기 민감도를 설정된 값과 비교 판단한다. 여기서, 상기 최적의 범위는 상기한 바와 같이 적정 기준 이상의 정확도와 에너지소모를 최대한 줄일 수 있는 범위로서, 이는 FIR필터의 종류 및 설계에 따라 그 기준을 적절하게 설정할 수 있다.

- [0034] 여기서, 분류된 상기 애더의 근사 비트수는 "0"으로 초기화한다.
- [0035] 또한, 상기 민감도를 0을 설정값으로 하여 0보다 작은 경우 상기 각 애더들을 합성하여 민감도와 정확도를 계산하는 단계를 다시 수행한다.
- [0036] 또한, 상기 각 애더를 합성하는 단계는, 첫 번째 스텝에 있는 애더들의 근사 비트수를 1 증가시키고, 그런 다음 합성을 통한 최고 동작 주파수를 확인하며, 시뮬레이션을 통한 정확도를 확인한 후, 근사 비트수 변화에 대한 민감도를 계산한다.
- [0037] 상기한 바에 따르면, 상기 근사연산을 이용한 FIR필터 연산방법은, 100%의 정확도를 요구하지 않는 영상, 검색, 그리고 이미지처리에 적용되는 것이 효과적이며, 이미지처리를 위한 디지털 필터로서 유한 임펄스 응답(IIR) 필터와 무한 임펄스 응답(FIR) 필터에 적용될 수 있다.
- [0038] - CSE (common subexpression elimination)
- [0039] CSE 알고리즘은 FIR 필터의 설계 복잡도를 줄일 수 있다. 이하에서 사용되는 용어들에 대하여 살펴보면, 애더 스텝(Adder Step;AS)은 FIR 필터의 계수를 실행하기 위해서 사용되는 애더의 개수를, 필터 애더 스텝(Filter Adder Step;FAS)은 FIR 필터의 critical path를 따르는 애더의 개수를 나타낸다.
- [0040] CSE 알고리즘 초기에, 모든 계수들은 canonical signed-digit codes로 변환되고 그것들의 연속 "0"들은 우 시프트 수식을 이용하여 삭제된다. 집합 C_N 은 변환 계수로 구해지고, 또 다른 집합 N_C 는 C_N 을 분해함으로써 구해진다. CSE알고리즘의 첫 반복에서, C_N 에서의 각 값은 $C_N \cup \{1\}$ 에서의 다른 값들에 의해 분해될 수 있는 지 아닌지를 결정하기 위해 체크된다. 그 값이 분해가능하다면, C_p 로 이동한다. 그렇지 않으면, 그 알고리즘은 그 값이 $C_N \cup N_C \cup \{1\}$ 의 값을 이용하여 분해가능 한지를 체크한다. 그리고 그 분해된 값은 C_p 로 이동한다. 이 구성에서의 N_C 에서의 값은 C_N 으로 이동한다. 이 과정들은 C_N 이 빌 때까지 반복된다. 상기 CSE 알고리즘을 따라, C_p 에서의 CSD 값은 도 3(b) 에서 승수 블록을 합성하기 위해서 사용된다. 이에 대한 예로서, FAS가 4라면, 계수들은 아래와 같이 되며, 간단하게는, CSD 계수들은 정수 형식으로 나타내 진다. 첫 번째 반복 전에, 첫 번째 반복에서, 815와 621은 831과 105로 각각 분해된다.

$$\begin{aligned} h_0 &= 105_{(10)} = 10101001_{(2)} & h_1 &= 831_{(10)} = 10101000001_{(2)} \\ h_2 &= 621_{(10)} = 1010010101_{(2)} & h_3 &= 815_{(10)} = 10101010001_{(2)} \end{aligned}$$

[0041]

$$\begin{aligned} C_p &= \phi \\ C_N &= \{105, 831, 621, 815\} \\ N_C &= \{3, 5, 7, 9, 13, 15, 17, 19, 23, 27, 31, 39, 47, 51, 63, 67, 97, \\ &\quad 109, 113, 123, 125, 127, 129, 137, 155, 159, 193, 209, 257, \\ &\quad 273, 493, 497, 509, 513, 625, 637, 641, 751, 767, 1007, 1023, \\ &\quad 1071, 1087\} \end{aligned}$$

[0042]

$$815 = 831 - 1 \times 2^4, \quad 621 = 831 - 105 \times 2^1$$

[0043]

[0044] 그 다음 단계에서는, 105와 831이 분해된다. 분해의 결과는 각각 아래와 같다.

$$105 = 15 \times 2^3 - 15 \quad \text{and} \quad 831 = 15 \times 2^6 - 129$$

[0045]

[0046] 그리고 마지막 단계로, 15 와 129 가 분해된다.

$$1 \times 2^4 - 1 \quad \text{and} \quad 129 = 1 \times 2^7 + 1$$

[0047]

[0048] 다음이 반복된다.

$C_P = \{105, 831, 621, 815, 15, 129\}$
 $C_N = \phi$
 $N_C = \{3, 5, 7, 9, 13, 17, 19, 23, 27, 31, 39, 47, 51, 63, 67, 97, 109,$
 $113, 123, 125, 127, 137, 155, 159, 193, 209, 257, 273, 493,$
 $497, 509, 513, 625, 637, 641, 751, 767, 1007, 1023, 1071,$
 $1087\}$

[0049]

[0050] 반복은 C_N 이 비었을 때 끝난다. 도2는 CSE 알고리즘으로부터 나온 합성된 FIR 필터이다.

[0051] - 근사 연산 (Approximate computing)

[0052] 근사 연산은 정확한 결과보다는 더 낮은 전력을 이용하여 충분히 좋은 결과를 출력한다. 그것은 일정 수준의 노이즈를 허용하는 어플리케이션(application)등에 사용될 수 있다. 여러 근사 연산 설계가 이전 연구들에서 행해졌다. Lu et al.은 자리 올림 비트를 계산하는데 있어서 입력의 이전 k 비트만을 고려하는 더 ??은 올림 chain을 가진 빠른 덧셈기를 소개한다. Verma et al.은 VLSA(variable latency speculative adder)를 제안했는데, error 감지와 수정 기능을 가진 Lu adder의 신뢰할 만한 버전이다. Shin et al.은 또한 올림 chain에서 critical path의 길이를 줄이는 여러 adder를 위한 data path 재설계 기술을 제안한다. Zhu et al.은 세 개의 근사 가산기들 ETAI, ETAII, ETAIIM을 제안한다. ETAI는 근사 결과를 얻기 위하여 정확한 파트 그리고 부정확한 파트로 나뉘어 진다. ETAII는 애더의 속도를 올리기 위하여 캐리 진행을 감소시키고, ETAIIM는 정확한 MSB 부분에서 올림 chain을 연결함으로써 ETAII를 수정한다. Gupta et al.는 transistor 레벨에서의 근사 연산을 제안하였는데, 영상기기용 멀티 비트(multi-bit) 덧셈기의 전력소모와 공간 사용을 줄이기 위하여 근사 전 가산기를 제안하였다. Kahng et al.는 정확도 조절가능 ACA를 제안하였다. 근사 모드에서는 올림 chain을 잘라냄으로써 근사연산을 수행한다. 정확 모드에서는 에러 감지 및 수정 회로에 의해서 정확성을 회복한다. ACA 애더는 근사 모드에서는 에너지 소비를 절약하고 정확 모드에서는 정확한 결과를 제공한다. Venkatesan et al.은 본래의 노드 세트에서 특정한 노드들을 제거하기 위한 근사연산용 시스템 설계 방법론을 제안하였고, 근사된 디자인의 정확도와 전력 소비를 분석하였다. 근사 곱셈기에 관하여도 여러 연구가 이루어졌다. DSP 기기들을 위하여 고정 비트 근사곱셈기(approximate multipliers)들이 에서 제안되었다. 그것들은 $W \times W$ 곱셈으로부터 얻어진 $(W-1)$ LSBs $(2W-1)$ 부분곱들을 제거한다. Cho et al. 과 Wand et al.는 곱셈에서 올림을 근사화하는 기술을 제안하였다. Lu et al.은 broken booth 승수를 제안하였으나, 이는 정확한 결과를 출력할 가능성이 낮다. Kulkarni et al.은 1/16의 에러 가능성을 가진 2×2 단위 곱셈기를 사용한 근사 곱셈기를 제안하였다. 단순화된 2×2 approximate 단위곱셈기는 단지 다섯 개의 게이트(Gate)를 가지고 있고 반면, 정확 단위곱셈기는 여덟 개의 게이트를 가지고 있다. 단순화는 근사 승수들의 임계 경로(critical path)의 길이를 줄일 뿐 아니라, 저전력 고성능 연산을 가능하게 한다.

[0053] - FIR 필터용 근사 합성

[0054] (1) 제안된 근사 가감산기

[0055] FIR 필터의 근사를 위하여, 정확도 조절 가능한 가감산기를 제시한다. 제안된 가감산기의 기본적인 법칙은 근본적인 Zhu가 제안한 애더의 법칙과 유사하다. 이 애더는 올림 발생 조건을 감지하고 더 상위의 비트로 올림을 진행시키는 대신 낮은 비트를 모두 1로 변경한다. 누적곱셈(Multiply and Accumulate) 회로를 구현하기 위해서는, 가산기와 감산기가 필요하다. 가산기 앞에 XOR gate들을 배치시킴으로써 가산기와 감산기를 스위치 할수 있게끔 하였다. 정확한 뺄셈 식을 위해서, 1을 1의 보수에 더함으로써 2의 보수를 만들어 더해야 하지만 제안된 근사 가감산기는 근사 부분(approximate part)의 올림이 정확 부분(Accurate part)에 확장되지 않기 때문에 1의 보수를 가지고 연산한다.

[0056] 상기 덧셈기는 두 부분으로 나뉘는데 도 5(a)에서 보여지는 바와 같이 accurate part 와 approximate part 이다. 애더의 비트 대역은 N이고 approximate part의 비트대역은 AP이다. Accurate part 의 연산 방법은 일반적인 덧셈기의 연산방법과 동일하다. Approximate part의 구조는 도 5(b)에 보여진다. AP-bit 올림 발생기와 AP-bit 합계 발생기로 이루어져 있다. 도 5(b)에서 보는 바와 같이, approximate part 에서의 올림은 최상위비트(Most Significant Bit)로 부터 최하위비트(Least Significant Bit)로 진행된다. 올림진행 방향은 기존의 덧셈기의 올림 진행의 방향과 역방향이다. 도 5(c)는 올림 발생기와 합계 발생기의 회로를 나타낸다. 이전 올림 발생기에서 올림이 발생되면, 다음 올림으로 넘어간다. 그렇지 않으면, 두 입력 피연산함수들이 비교되어 양쪽 다 "1"인 경우에는 올림이 발생한다. 합계 발생기는 올림 발생기로부터 올림을 받는다. 올림이 생기면, 합계 발생

기는 "1" 을 돌려준다. 그렇지 않으면, 합계 발생기는 두개의 입력 피연산함수를 더하여 합계값을 돌려준다. 가감산기의 정확도는 매개변수 AP와 approximate part의 비트대역을 바꿈으로써 조절가능하다. AP는 0부터 N까지의 값을 가질 수 있다. 만약 AP가 0이면, 제안된 가감산기의 결과는 기존의 가감산기의 결과와 동일하다. 만약 AP가 증가하면, 입력의 정확도는 감소하나 전력소비는 줄어들고 성능이 향상된다. 그러나, AP가 어떤 값 보다 더 크면, approximate 부분의 확장 지연이 accurate part의 확장지연이 되고, 추가 근사화로 인해 생기는 이득이 감소한다. 그러므로, AP 값은 근사 합성 flow 동안에는 적절하게 설정되어야 한다.

[0057] 근사(Approximation)에서의 최대 에러는 근사 부분(approximate part)의 모든 입력 비트가 1 일 때 발생한다. 이 경우, 두개의 입력 피연산함수는 $2^{AP}-1$ 이다. 기존 애더들의 출력은 $(2^{AP}-1) \times 2$ 이고, 반면 근사 애더는 $2^{AP}-1$ 이다. 결과적으로, 제안된 근사 가감산기에서 발생할수 있는 최대 에러는 $2^{AP}-1$ 이다. 반면, 근사 부분 연산을 무시하는 경우에, 최대 에러는 제안된 애더에 의해서 발생된 것의 두배인 $(2^{AP}-1) \times 2$ 이다. 예를 들면, N, AP 와 두 입력이 각각 8, 4, 01101111₍₂₎와 00011111₍₂₎ 이면, 네 개의 상위비트들이 정확 부분(accurate part)에 의해 계산되고 네 개의 하위비트들은 근사 부분(approximate part)에 의해 더해진다. 정확 부분과 근사 부분 들의 출력물은 각각 0111₍₂₎ $\times 24$ 와 1111₍₂₎ 이고 결과는 01111111₍₂₎와 127₍₁₀₎ 이다. 이 덧셈의 정확한 결과는 10001110₍₂₎, 142₍₁₀₎이고 근사연산으로 인해 발생한 에러는 2^4-1 인 15₍₁₀₎이다. 이 결과로부터, 근사 연산으로 인해 발생하는 에러가 하위비트를 무시함으로써 발생하는 에러 보다 작음을 알 수 있다.

[0058] 근사 연산으로 얻어지는 출력물의 양을 증명하기 위하여 수학적 식 1과 같이 정의된 정확도 미터를 사용하였다.

[0059] 수학적 식 1

$$accuracy = \min_{k=1, \dots, M} \left(1 - \frac{|result_k - ref_k|}{|ref_k|} \right) \times 100\% \quad (1)$$

[0060]

[0061] 여기서, M은 입력 패턴의 갯수이다. result_k는 k번째 입력 패턴으로부터 생성되는 근사 결과 이고 ref_k는 옳은 결과이다.

[0062] (2)근사 합성 플로우

[0063] 이하에서는 제안된 근사 합성 플로우를 묘사한다. 합성 플로우의 목적은 근사 애더의 최적의 AP배열을 찾아내는 것이다. 이들 최적의 배열을 이용하여, 특정 최소 제약, accuracy_{min} 보다 더 높은 정확도를 유지하면서 에너지/전력 소비를 절약하고 기능을 개선할 수 있다. 그러나 최적으로 계산된 애더의 AP를 찾는 것은 계산의 가능한 조합의 개수가 M_{adder}^N (여기서, M_{adder}는 애더의 수 이고 N은 애더의 비트대역이다)에 비례하기 때문이다.

Algorithm 1 Sensitivity-based approximate synthesis flow.

```

1: Classify adders according to AS
2:  $AP_i \leftarrow 0$ , where  $i = 1, \dots, FAS$ .
3: while  $SF_{best} > 0$  do
4:   for  $i := 1$  to  $FAS$  do
5:      $AP_i \leftarrow AP_i + 1$ 
6:     Synthesis  $\{newAP_0, newAP_1, \dots, newAP_{FAS}\}$ 
7:     Calculate  $delay_i$ 
8:     Gate level simulation
9:     Calculate  $accuracy_i$ 
10:    Power analysis
11:    Calculate  $power_i$ 
12:    Calculate  $SF_i$ 
13:    Recover design  $AP_i \leftarrow AP_i - 1$ 
14:   end for
15:    $SF_{best} = \max(SF_1, SF_2, \dots, SF_{FAS})$ 
16:   if  $SF_{best} > 0$  then
17:     Select  $\{AP_1, AP_2, \dots, AP_{FAS}\}_{best}$ 
18:   end if
19: end while
20: Return  $\{AP_1, AP_2, \dots, AP_{FAS}\}$ 

```

[0064]

[0065]

추가적인 설명을 위하여 도 4는 입력의 비트대역, 계수, 그리고 출력물은 각각 15, 12, 그리고 28비트 이다. 계수들은 Section 2-1 에서 FAS = 3으로 소개되었던 CSE 알고리즘으로부터 합성된 계수 105, 831, 621, 815이다. M_{adder} 가 6 이고 N 가 28 비트라고 가정했을 때, AP들의 가능한 조합의 개수는 대략 6.14×1021 . 기존의 FIR 필터의 AP계산의 가능한 조합은 이 예에서 보다 더 수가 훨씬 많다. 모든 조합을 찾아내는 것은 시간 및 자원낭비이고 더 큰 설계일 경우 불가능하다.

[0066]

이러한 문제를 해결하기 위하여, 두개의 가정을 한다. 첫 번째는, 가산기와 감산기의 지연이 비슷하다는 것이다. 두 번째, 가감산기의 데이터 실제 도착 시간(Actual arrival time)이 같은 AS를 가진 또 다른 가감산기의 그것과 비슷하다는 것이다. 그러므로 한 경로에서만 AP를 바꾸는 것은 애더의 AP를 동시에 바꾸는 것 보다 덜 효과적이다. 가능한 조합의 개수는 FAS^M 에 비례한다. FIR 필터의 FAS가 M_{adder} 의 그것 보다 훨씬 작다는 것을 고려하면, 조사해야 할 조합의 개수를 현저히 줄이는 것이 가능하다. FAS 는 3이고 N은 28 비트라는 것을 가정할 때, AP의 가능한 조합의 개수 2.28×1013 이다. 근사 합성 플로우 동안, AP는 보통 N의 반 보다 작으며, 실제 설계 공간은 대략 $FAS^{M/2}$ (이 경우 470만)이다. 이는, 모든 가능한 조합 6.14×1021 보다 더 합리적인 값이다.

[0067]

알고리즘은 제안된 근사 합성 플로우의 과정을 묘사한다. 그 플로우는 최소 지연과 요구되는 정확성을 가진(즉, $accuracy_{min}$ 보다 높은) 근사 설계를 찾는다. 첫 번째 단계로, 기준치 설계가 장전되고 모든 애더들이 그들의 AS(Line2)에 따라 분류된다. 해당 AS의 모든 AP들은 0 (Line2) 으로 세팅이 된다. 그 다음, 해당 AS에서 AP들은 1을 더한다(Line5). 바뀐 베릴로그 설계는 합성되고 설계상의 지연이 계산된다 (Line 6-7). 합성된 설계를 사용하여, 전력과 정확성을 계산하기 위해 게이트 레벨 시뮬레이션과 정적 시간 분석이 실행된다 (Lines 8-11). Slack 과 정확성을 이용하여 민감요소 (SF)가 계산된다 (Line 12). 정의된 민감요소는 수학적식 2와 같다.

[0068]

수학적식 2

$$SF = \begin{cases} \frac{accuracy - accuracy_{min}}{delay} & , \text{if } accuracy > accuracy_{min} \\ 0 & , \text{else} \end{cases} \quad (2)$$

[0069]

[0070]

여기서 정확도는 수학적식 1로 정의된다. 계산된 SF_i 는 SF 목록에 추가된다. 다음의 계산은, 해당 루프의 초기값

으로 돌아간다(Line 13). 만약 모든 해당 루프에서 바뀐 SF 계산이 SF 목록에서 완료되면, 가장 높은 SF를 가진 설계가 선택이 된다 (Line 15). 선택된 설계는 다음 루프의 초기값으로 쓰인다. 만약, 가장 높은 SF가 0 이거나 마이너스 이면, 플로우는 이전 루프의 초기값을 최종답으로 반환하고 끝난다. 제안된 합성 플로우는 도 4에 요약되어진다. 주요 관심사인 낮은 전력과 높은 에너지 효율 디자인은 적절한 시간 제한, 즉 최소한의 기준설계의 허용시간으로 합성 플로우는 최종 답을 재합성하는 것을 이루어 낼 수 있다.

	Delay [ps]	Power [uW]	Energy [fJ]
Baseline	1199	2796	3352
Flow result	1076	1687	1815
Min. Energy design	1198	1379	1652
Improvement [%]			
	Delay	Power	Energy
Flow result	10.3	39.7	44.7
Min. Energy design	0.0	50.7	50.7

Table 2: Approximation results in 25-tap filter with $FAS = 4$.

	Delay [ps]	Power [uW]	Energy [fJ]
Baseline	1988	10.7	21.3
Flow result	1876	8.9	16.7
Min. Energy design	1983	8.2	16.3
Improvement [%]			
	Delay	Power	Energy
Flow result	5.6	16.8	21.6
Min. Energy design	0.0	23.3	23.5

- 실험 결과

(1) 실험 준비

제안된 근사 합성 플로우는 사용하여 FIR 필터를 구현하였다. 계수가 105,831,621,815 인 4개의 계수를 가진 FIR 필터를 합성한다. 도 2는 시행된 FIR 필터의 구조를 보여준다. 이 실험에서, 계수의 비트 넓이는 12로 정해진다. 가장 큰 계수는 4개 탭 FIR 필터에서 831이기 때문에, 12비트는 SPT에서 4개 계수를 반영하기에 충분하다. 입력과 출력의 비트 밴드는 각각 15 비트와 28비트로 정해진다. 연산을 위하여 28비트 가감산기들이 사용되었다. 네 개의 주어진 계수들은 이전 소개된 CSE 알고리즘에 따라 6개의 가감산기로 구현되었다. 각 계수의 AS는 다르다. 15, 129의 AS는 1이고 105,831의 계수는 2 이다. 합성 플로우는 각 반복에서, 같은 AS를 가진 애더들의 정확도 배열이 하나씩 바뀐다. 바뀐 설계 ($\{1,0,0\}$, $\{0,1,0\}$, $\{0,0,1\}$)은 합성되고 모의된다. 가장 높은 SF를 가진 $\{0,0,1\}$ 이 선택되고 다음 반복에 초기값으로 세팅된다. 몇 번 반복 후에, 최종 출력은 $\{11,16,14\}$ 가 된다. 도5(a), (b), (c)는 제안된 합성 플로우는 사용하여 구현된 디자인의 전력소모와 성능 그리고 정확도를 보여준다. 검은 점들은 각각의 가감산기의 AP를 모두 랜덤하게 발생시킴으로써 얻어진다. 빨간 점들은 근사 합성 플로우는 반복의 결과를 나타낸다. 하얀 공간은 $accuracy_{min}$ 보다 낮은 정확도를 가진 도달 가능한 설계 공간을 보여준다. 도 5(a)에서 보여주듯이, 제안된 합성 플로우는 계산 지연을 최대한으로 줄일 수 있다. 게다가, 제안된 합성 플로우는 효과적으로 전력과 에너지 소비를 줄일 수 있다.

주요 관심사는 높은 에너지 효율을 얻어내는 것이기 때문에, 합성 플로우로부터 획득된 설계를 시간 제약을 바꿔가며 재합성시킨다. 그러면, 기초 설계의 에너지 소비를 초과하지 않으면서 지연으로 가진 가장 낮은 에너지 소비의 결과를 선택할 수 있다. 도 5(a) 에서, EDA의 톨 불안정성 때문에 정확도 97.5%에 가까운 한 설계가 최종 답 설계보다 약간 낮은 지연과 전력 소비를 보인다. 그러나, 다음 재 합성에서는 포인트의 에너지 소비가 최종 답의 에너지 소비 보다 크다.

표 1은 근사 합성 플로우는의 결과를 요약한다. 기능이 10.3% 개선되고 전력소비가 기존의 FIR 필터 설계보다 39.7% 감소한다. 에너지는 지연과 전력을 곱하여 계산된다. 연산당 에너지 소비는 44.7% 감소한다. 에너지를 더

줄이기 위하여, 시간제한을 바꾸었고 지연이 기초 설계보다 더 짧은 최소 에너지 설계를 찾았다. 이런 방법으로, 에너지 소비를 50.7% 까지 줄일 수 있다. 제안된 합성플로우를 이용하여 FIR 필터를 설계하는데 84분의 시간이 소요된다.

[0077] 근사 합성 플로우를 계수가-2423, -113, 1564, 762, -1816, -1517, 2276, 3140, -2434, -6205, 2726, 20680, 30093, 20680, 2726, -6205, -2434, 3140, 2276, -1517, -1816, 762, 1564, -113, -2423인 25개 탭 FIR 필터에 적용한다. 결과는 표 2에 보여진다. 계수가 25개인 경우에는, 전력과 에너지 소비가 각각 23.3%, 23.5% 까지 절약되어 기능이 5.6% 개선된다. 25개 계수의 FIR 필터를 합성하는 데는 407분이 소요된다.

[0078] 표 1

FIR Filter	Tap	FAS	Delay [ns]	Power [mW]	Energy [pJ]
[21]	15	3	0.98	5.68	5.5
[22]	15	4	1.27	4.06	5.1
[23]	28	4	1.15	11.6	13.4
[24]	34	3	1.17	13.4	15.7
[25]	49	3	1.20	18.4	22.1

Table 4: Following the proposed synthesis flow

FIR Filter	Accuracy [%]	Delay [ns]	Power [mW]	Energy [pJ]	Energy Reduction [%]
[21]	97.83	0.93	4.34	4.06	26.9
[22]	95.32	1.14	3.17	3.62	29.5
[23]	96.03	1.15	8.13	9.34	30.1
[24]	95.66	1.15	8.34	9.59	38.9
[25]	95.19	1.12	13.60	15.27	30.8

[0079]

[0080] 2. 이미지 FIR 필터 실험

[0081] 방법론을 증명하기 위하여, 제안된 합성 플로우를 5개의 다른 FIR 필터들 (21-25)에 적용한다. FIR 필터의 상세 특징이 도 3에 요약되어진다. FIR 필터의 기초 설계에 대한 지연, 전력, 그리고 에너지 정보는 도 3에 요약된다. FIR 필터는 제안된 합성 플로우를 사용하여 합성되는 반면, 입력, 계수, 출력의 비트밴드는 각각 8, 16, 그리고 24 이다. 합성 플로우의 결과는 도 4에 나온다. 필터의 정확도가 95%한계점 보다 더 높다. FIR 필터의 에너지 소비는 평균 38.9% 그리고 31.2% 까지 줄어든다.

[0082] (23)에서 FIR 저주파 필터를 구현하였다. 그 영상이 이차원으로 사용되어지기 때문에, 수직 방향으로 FIR필터를 한번 적용하고 출력값을 필터 이득으로 나눈다. 그 다음, FIR 필터가 수평방향으로 적용되고 출력이 다시한번 필터이득으로 나누어진다. 도 7(a)는 원래 이미지이고, 도 7(b)는 기초 FIR 필터로 희미한 영상을 처리한 모습이다. 도 7(c) 제안된 FIR 필터로 이미지를 처리한 모습이다. 처리 영상의 출력 질을 증명하기 위하여 정점 PSNR (소음대 신호율, 최대 신호대비 잡음비)이 사용되어진다. PSNR은 다음의 수학적 식 3으로 정의된다.

[0083] 수학적 식 3

$$PSNR = 10 \times \log\left(\frac{255^2}{\sigma_{noise}^2}\right) \quad (3)$$

[0084]

[0085] σ_{noise}^2 은 도 7(b) 와 다른 것들 사이에서의 차이의 변화를 보여준다. 여러 정확도를 가진 FIR 필터들이 모의 실험된다. 정확도가 줄어들수록 이미지는 어두워진다. 이는 제안된 애더가 이전 올림을 근사치 시키고 근사 애더가 결과의 값을 정확한 결과보다 더 낮게 만들기 때문이다. 근사 애더가 계속 증가한다면 결과는 마이너스 값이 될 것이고 하얀 점으로 나타난다. 46.3dB PSNR로 30.8%의 에너지 절약을 얻어낼 수 있다.

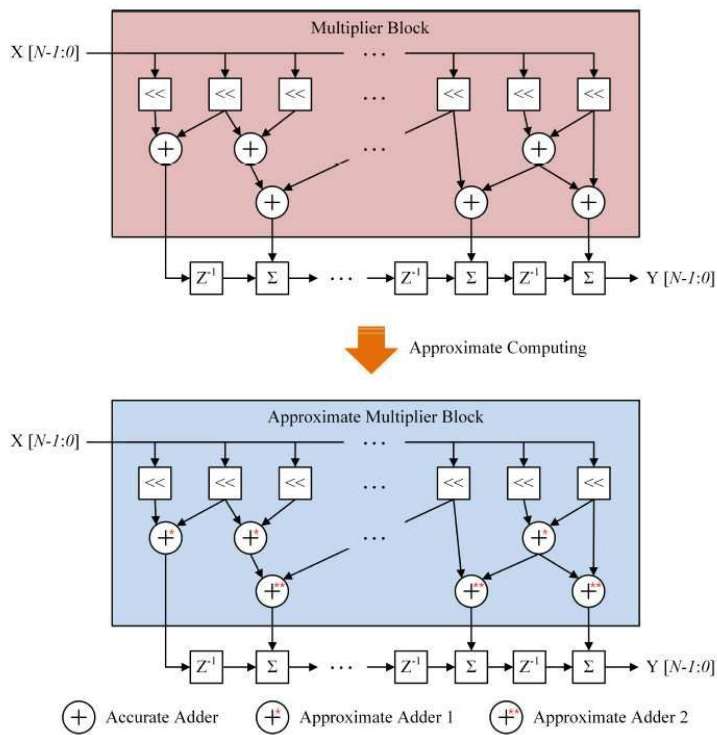
[0086] - 결론

[0087] 에너지 소비를 강화하기 위하여 근사 계산을 FIR 필터에 적용하였다. FIR 필터는 MAC 구조를 가지고, 곱셈기는 쉬프터와 근사화된 가감산기로 대체된다. 근사를 위하여, 근사 가감산기의 정확도가 조절가능하고 가/감산기로의 변환이 가능한 근사 가감산기를 제안한다. 게다가, 근사 애더들을 위하여 최적의 배열을 찾아내기 위하여 새로운 근사 합성 플로우를 제공할 수 있으며, 이러한 합성 플로우를 사용하여 10.3%의 성능 향상과 50.7%의 전력과 에너지 절약을 이루어냈다.

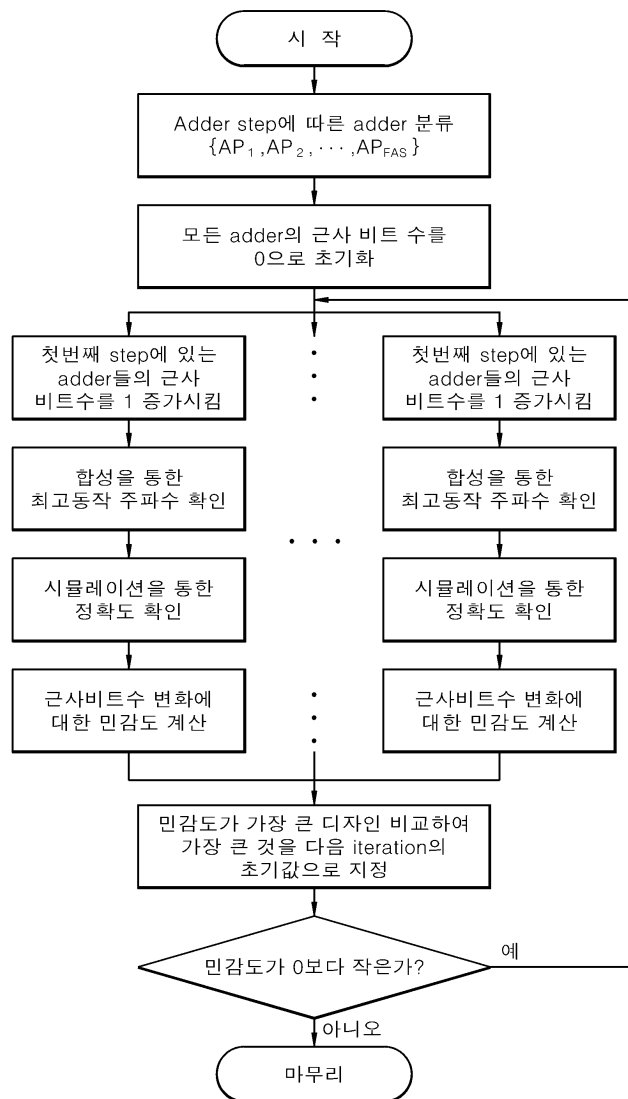
[0088] 본 발명은 도면에 도시된 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 다른 실시예가 가능하다는 점을 이해할 것이다. 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 특허청구범위의 기술적 사상에 의하여 정해져야 할 것이다.

도면

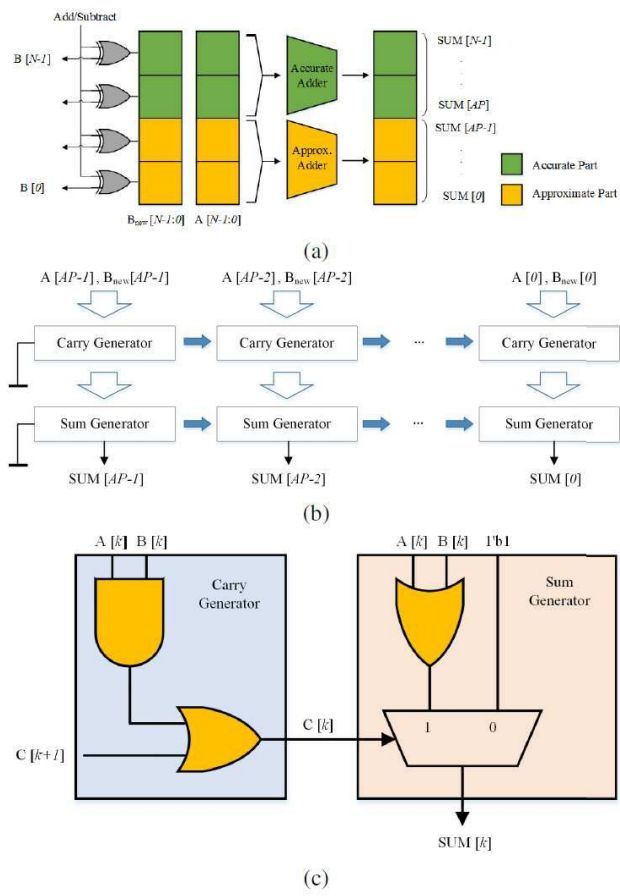
도면1



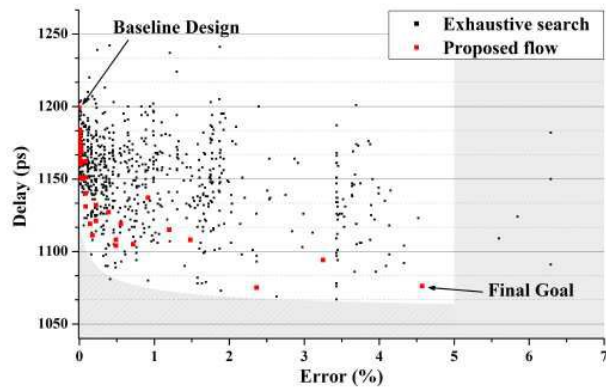
도면2



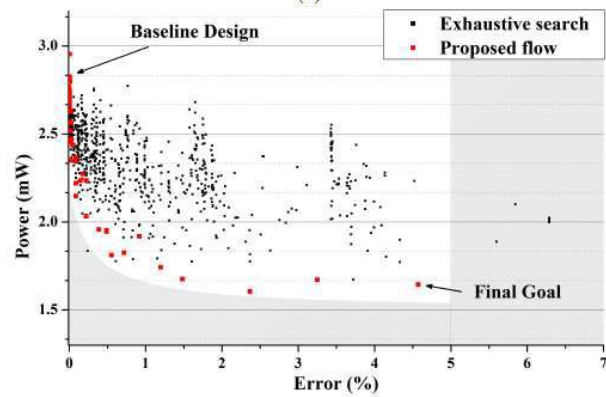
도면5



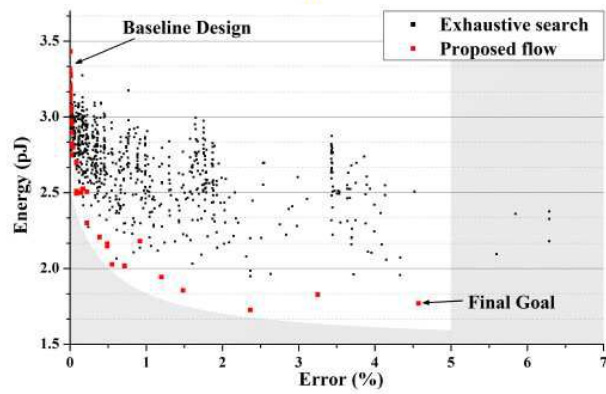
도면6



(a)



(b)



(c)

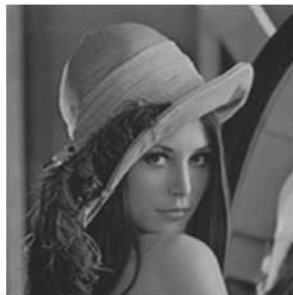
도면7



Original
(a)



Baseline design
(b)



Proposed
PSNR = 46.3
30.0% Energy reduced
(c)



89.2% Accuracy
PSNR = 40.6
36.9% Energy reduced
(d)



65.4% Accuracy
PSNR = 14.7
45.7% Energy reduced
(d)



45.6% Accuracy
PSNR = -9.5
48.8% Energy reduced
(e)