[5A-107]

MBA 기반 고성능 병렬 MAC의 VLSI 구조

서영호, 김동욱
광운대학교 전자재료공학과
design@kw.ac.kr ddntlab.kw.ac.kr

VLSI Architecture of High-Performance Parallel MAC based on MBA

Young-Ho Seo, and Dong-Wook Kim
Department of Electronic Materials Eng., Kwangwoon University

본 논문에서는 고속의 곱셈-누적 연산을 수행할 수 있는 새로운 MAC(Multiplier-Accumulator)의 구조를 제안하였다. 부분곱의 생성을 위해서 1의 보수 기반의 고속 Booth 알고리즘(Modified booth algorithm, MBA)를 이용하였고 다수의 부분곱을 더하기 위해서 CSA(Carry save adder)를 이용하였다. 부분곱을 더하는 과정에서 Booth 인코딩 시 이용한 1의 보수 체계를 2의 보수 체계로 보상하고 이전 합(Sum)과 캐리(Carry)를 누적하는 연산을 수행하여 고속의 누적 연산이 가능한 구조를 제안한다. 또한 부분곱의 덧셈에서 하위 비트들을 2 비트 CLA(Carry look ahead adder)를 이용하여 연산함으로써 최종 덧셈기의 입력 비트수를 줄임으로써 전체적인 임계경로를 감소시켰다.

Ⅰ. 서 론

일반적으로 곱셈기에는 Booth 알고리듬[1]과 전가산기의 배열, 또는 Booth 알고리듬과 Wallace 트리(Tree)를 이용한 방법이 많이 이용되고 있으며 이러한 곱셈기는 크게 Booth 인코더, 부분곱 압축 트리, 최종 덧셈기의 세부분으로 구성된다[2]. Wallace 트리는 인코더로부터 나오는 부분곱들의 덧셈을 최대한 병렬로 수행하는데 (N-bit) 데이터의 곱셈을 위하여 $O(log_2N)$에 비례하는 수행시간을 가진다. 즉, CSA(Carry save adder)를 사용하여 N개의 입력 중에서 1의 개수를 카운트하여 출력으로 내보내면 출력 비트의 개수는 log_2N개로 줄어드는 원리를 이용한다. 실제로는 다수의 비트수를 입력으로 사용하는 CSA를 구현하는데 많은 면적이 필요하므로 (3:2) 혹은 (7:3) 카운터를 다수 사용하여 파이프라인 단계마다 이러한 카운터의 개수가 줄어들도록 하고 있다. 곱셈 과정은 일련의 부분곱 덧셈 과정을 반복적으로 수행하는 것이기 때문에 고속 승산을 위해 부분 곱의 수를 줄임으로서 계산 단계를 줄이는 MBA 알고리즘이 적용되고 있으며 또한 Wallace 트리를 적용하여 부분곱의 빠른 덧셈을 수행하고 있다. 최근 MBA의 속도를 높이기 위해 병렬 곱셈구조가 많이 연구되고 있는다[3]. 일반적으로 "Baugh-Wooley Algorithm(BWA)"에 기초하여 MAC의 구조가 개발되어 왔고 제안된 구조들은 다양한 디지털 필터링 연산에 도입되었다[4].

범용적인 디지털 신호처리를 위한 가장 발달된 형태의 MAC 중에 하나가 [5]에서 제안되었는데 부분곱을 압축하는 CSA 트리에 누적 연산을 결합한 구조를 제시하고 있다. 본 논문에서는 [5]의 구조를 바탕으로 고속의 곱셈-누적 연산을 수행할 수 있는 새로운 MAC의 구조를 제안한다. [5]와 마찬가지로 부분곱의 생성을 위해서 1의 보수 기반의 MBA 알고리즘를 이용하였고 다수의 부분곱을 더하기 위해서 합과 캐리형태의 누적 연산을 수행하는 하이브리드 형태의 CSA 구조를 바탕으로 새로운 MAC의 구조를 제안한다.

Ⅱ. Booth 곱셈기와 MAC의 구조

곱셈기는 크게 세 부분으로 나눌 수 있는데, 첫 번째는 입력으로 들어오는 Multiplicand와 Multiplier로부터 부분곱(Partial product)들을 만들어내는 인코더 부분이고 두 번째는 생성된 부분곱들을 모두 더하여 합과 캐리형태의 값으로 만드는 덧셈기 배열 혹은 부분곱 압축 부분이다. 그리고 마지막은 합과 캐리를 더하고 최종적인 곱셈결과를 만드는 최종 덧셈기 부분이다. 여기에 곱한 결과를 누적하는 과정을 포함시키면 총 네 가지 단계로 나누어진다.

일반적인 MAC의 연산은 식(1)처럼 표현할 수 있다.

$$X=-2^{N-1}x_{N-1}+\sum_{i=0}^{N-2}x_i2^i, \qquad x_i\in\{0,1\} \tag{1}$$

식 (1)을 Booth 알고리즘을 적용하기 위하여 Base-4 형태