부동소수점 라운딩 병렬화 알고리즘의 하드웨어 구현을 위한 구조 설계

이원희 강준우
한국외국어대학교 전자공학과
449-791 경기도 용인시 모현면 왕산리 산 89 번지
Tel : 0335-330-4502, Fax : 0335-330-4120, E-mail : leewh@san.hufs.ac.kr

Architectural Design for Hardware Implementations of Parallelized Floating-point Rounding Algorithm

Won-Hee Lee and Jun-Woo Kang
Dept. of Electronics Engineering, Hankuk University of Foreign Studies
89 Wangsan, Mohyun, Youngin, Kyonggi-Do, 449-791, KOREA
Tel : 0335-330-4502, Fax : 0335-330-4120, E-mail : leewh@san.hufs.ac.kr

Abstract

Hardware to implement the parallelized Floating-point rounding algorithm is described. For parallelized additions, we propose an addition module which has carry selection logic to generate two results according to the input values. A multiplication module for parallelized multiplications is also proposed to generate Sum and Carry bits as intermediate results. Since these modules process data in IEEE standard Floating-point double precision format, they are designed for 53-bit significands including hidden bits. Multiplication module is designed with a Booth multiplier and an array multiplier.

I. 서론

부동소수점 연산에서 라운딩은 정확한 결과 값을 얻기 위해 필수적인 절차이다. 라운딩을 부동소수점 연산기에서 구현하기 위하여 일반적으로는 덧셈, 정규화, 라운딩의 순서로, 또는 덧셈, 라운딩, 정규화의 순서로 처리한다. 그 결과 처리 시간이 길어지고 많은 하드웨어가 필요하다. 이러한 지연시간과 추가적인 하드웨어를 줄이기 위해서 연산과 라운딩을 병렬화하는 연구가 발표된 바 있다[1-5]. [1,2,5]에서는 덧셈 라운딩 병렬화 알고리즘을 하드웨어로 구현하기 위하여 올림수 입력이 '0'일 때와 '1' 일 때의 두 가지 결과 값을 출력할 수 있는 덧셈 모듈을 사용하였다. 이 덧셈 모듈을 복식 가산기(Compound adder)라고도 한다. 복식 가산기를 만들 수 있는 덧셈기로는 올림수 선택 덧셈기(Carry select adder), 조건합 덧셈기(Conditional-sum adder), 올림수 예측 덧셈기(Carry lookahead adder) 등이 있다[6]. 이 덧셈기들은 부분 결과 값으로 두 가지 결과 값을 출력한다.

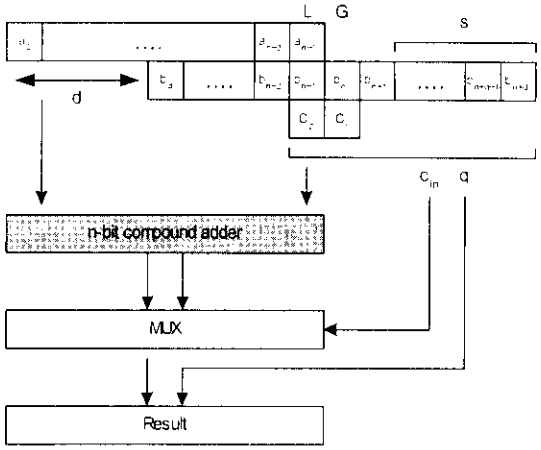
본 논문에서는 올림수 선택 덧셈기(이하 CSA)의 구조를 바탕으로 두 가지 최종 결과 값을 출력하는 복식 가산기를 제안하였다.

[3-5]에서는 중간 결과값으로 합과 올림수를 출력할 수 있는 곱셈기를 사용하였다. 이러한 곱셈기로는 배열 곱셈기가 사용되었다[6].

IEEE 표준의 부동 소수점 수 배정도 표현에 따르면 64 비트 부동소수점 수는 지수(exponent)가 11 비트이고 가수(significand)는 52 비트이다[7]. 가수는 숨겨진 비트를 포함하면 53 비트이며, 이 수가 실제 연산 모듈에 입력된다. 따라서, 라운딩 병렬화 알고리즘을 위한 덧셈, 곱셈 모듈은 53 비트이다. 특히, 곱셈 모듈에서 사용된 배열 곱셈기는 2 의 거듭 제곱 비트 연산 모듈 설계에는 적합하나 그 이외의 비트 수 연산 모듈 설계에는 사용하기 어렵다. 그래서 본 논문에서는 53 비트 배열 곱셈기를 설계하기 위해 27 비트 기본 곱셈 블록을 Booth 곱셈기로 설계하여 53 비트 배열 곱셈기를 설계하였다.

II. 라운딩 병렬화 알고리즘

n 비트의 부동소수점 수의 덧셈, 뺄셈, 곱셈의 결과 값은 2n 비트이다. 이 중 상위 n 비트는 취하고 하위 n 비트는 삭제된다. 본 논문에서는 취하는 부분을 정수부로 하고 삭제된 부분을 분수부로 정의한다. 라운딩은 분수부에서 정수부로의 올림수이므로 분수부만이 라운딩 처리에 쓰인다. 라운딩과 연산을 동시에 처리하기 위해서는 정수부와 분수부를 분리하여 처리하여야 한다.



(그림 1) 덧셈 라운딩 병렬화 알고리즘 하드웨어 모델

1. 덧셈과 뺄셈 라운딩 병렬화 알고리즘

덧셈의 경우 분수부의 라운딩 값에 따라 복식 가산기로 계산된 정수부의 두 가지 결과 값 중 하나가 선택된다. 뺄셈의 경우에는 2의 보수를 만들 때 추가되는 1이 정수부에 더해질 때 분수부의 값들은 모두 '0'이 되므로 라운딩 값이 '0'이 된다. 따라서 2의 보수를 만들 때 추가되는 '1'과 라운딩을 동시에 처리할 수 있다. 이 값에 의해서 복식 가산기의 두 가지 결과 값 중 하나가 선택된다. 이러한 하드웨어 모델은 (그림 1)과 같다 [5].

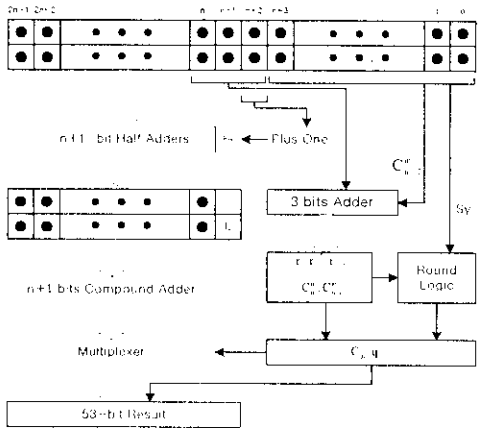
2. 곱셈 라운딩 병렬화 알고리즘

곱셈의 경우 중간 결과값으로 합과 올림수가 발생하고 이 두 값의 합으로 최종 결과 값이 결정된다. 합과 올림수를 더할 때 정수부와 분수부를 분리하여 정수부는 복식 가산기를 사용하여 두 가지 결과 값을 출력하고 이 두 결과 값 중 하나를 분수부에서 구한 라운딩 값에 의해 선택한다. 이러한 하드웨어 모델은 (그림 2)와 같다[5].

III. 덧셈 모듈의 구조

1. CSA의 기본 구조

올림수 선택 덧셈기에서는 n 비트 덧셈의 경우, n 비트를 몇 개의 그룹으로 나누어, 각 그룹이 합(Sum)과 올림수(Carry)를 최하위 비트로의 올림수가 '0' 일 때와 '1' 일 때로 나누어 한 쌍씩 출력하고 올림수 선택 회로(Carry selector)에서 결과 값을 선택한다. 예를 들어 16-bit 올림



(그림 2) 곱셈 라운딩 병렬화 알고리즘 하드웨어모델

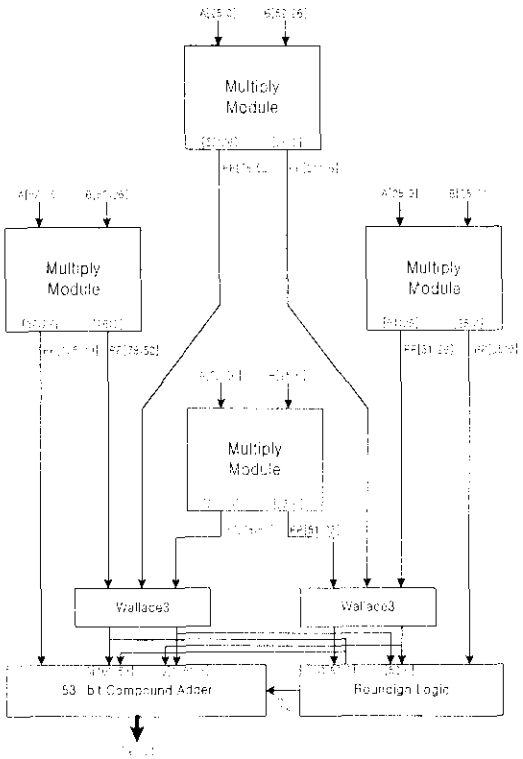
수 선택 덧셈기는 먼저 그룹을 4 비트씩 4 개로 나누고 각 그룹은 올림수 입력이 '0'일 때와 '1'일 때를 4 비트 덧셈 모듈로 계산한다. 이 값이 멀티플렉서로 입력되고 올림수 선택 회로에 의해서 각 그룹의 결과 값을 얻을 수 있다. 16-bit 올림수 선택 덧셈기의 구조는 (그림 3)과 같다[6].

2. 라운딩 병렬화 알고리즘에 쓰이는 CSA

(그림 3)에서 16 비트 CSA는 SA에서 두 가지 결과 값을 출력하나 이것은 MPX에서 한가지만 선택함으로써 최종 결과 값은 한가지가 된다. 여기에 MPX를 각 그룹마다 2 개씩 할당하여 두 가지 합 값을 만들고 CS 부분도 각 그룹마다 2 개씩 할당하여 올림수도 2 가지를 출력할 수 있다. 이 구조는 (그림 4)와 같다. (그림 4)에서 (L)은 '0'일 때 합 혹은 올림수 값이고 (H)는 '1'일 때 값이다. 53 비트 덧셈기는 위에서 설계한 두 가지 결과 값을 갖는 16 비트 CSA 3 개를 사용하고 상위 5 비트가 남는다. 이것은 4 비트 그룹과 1 비트 전가산기를 사용하여 처리하였다. 이 구조는 (그림 5)와 같다. 이 두 가지 결과 값을 갖는 53 비트 CSA가 라운딩 병렬화 알고리즘을 구현할 때 복식 가산기로 쓰인다.

IV. 곱셈 모듈의 구조

곱셈에서 라운딩 병렬화 알고리즘을 수행하기 위해서는 곱셈의 중간 결과 값 합과 올림수가 생성되어야 한다. 중간 결과 값으로 합과 올림수를 생성하기에 알맞은 곱셈기로는 배열 곱셈기(Array multiplier)[6]가 있다. 배열 곱셈기는 면적 크기가 작은 간단한 곱셈기 모듈

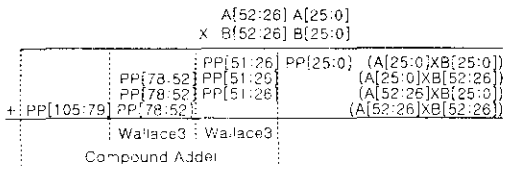


(그림 6) 53-비트 모성능 곱셈기

을 설계하고 이것을 바탕으로 배수 크기로 커지는 곱셈기를 만든다. 예를 들어 8 비트 곱셈기는 먼저 4 비트 기본 곱셈기를 설계하고 8 비트의 상위 4 비트와 하위 4 비트를 분리하여 각각 4 개의 4 비트 곱셈기로 곱하고 이 값을 같은 자릿수끼리 더하면 합과 올림수가 발생하고 이것을 더하면 8 비트 곱셈기가 완성된다. 이것을 확장하여 16 비트, 32 비트, 64 비트 곱셈기를 설계할 수 있다. 라운딩 병렬화 알고리즘을 적용하기 위해 합과 올림수 덧셈으로 복식가산기를 사용한다.

64 비트 부동소수점 수는 IEEE 표준에 따라 숨겨진 비트를 포함하여 53 비트까지 가수이므로 53 비트 곱셈기를 설계해야 한다. 배열 곱셈기로 중간 결과 값 합과 올림수를 출력하는 것은 쉽지만 53 비트의 배열 곱셈기를 설계하기는 어렵다. 53 비트는 홀수이므로 54 비트에 맞추어 27 비트 기본 모듈을 설계한다. 64 비트 배열 곱셈기에서와 같이 4 비트를 확장하여 27 비트 곱셈기를 만들 수는 없으므로 27 비트 곱셈기로는 Booth 곱셈기를 사용한다. 이 27 비트 Booth 곱셈기를 기본 모듈로 하는 배열 곱셈기는 (그림 6)과 같다.

(그림 6)에서 Multiplier Module은 27 비트 Booth 곱셈기로 된 기본 모듈로서 전체 회로는 다음과 같은 연산을 수행한다.



위에서 A[25:0]은 피연산자 A 의 0 비트에서 25 비트까지를 나타낸다. Wallace3 는 3 입력 Wallace Tree 이고 이 부분이 3 개의 같은 자릿수를 더해서 합과 올림수를 출력한다. 이 값의 상위 53 비트가 2 가지 결과 값을 출력하는 복식 가산기로 처리되고 이 결과 값이 멀티플렉서로 입력되어 라운딩 값에 의해서 한 가지가 선택된다.

V. 결론

본 논문에서는 라운딩 병렬화 알고리즘을 구현하기 위한 복식 가산기, 합과 올림수를 중간 결과 값으로 갖는 곱셈기 등의 하드웨어 설계에 대하여 기술하였다. 두 가지 결과 값을 출력하는 CSA를 설계하여 라운딩에 의한 최종 결과 값 선택을 가능케 함으로써 연산 수행 동안에 라운딩을 처리할 수 있다. 곱셈에서는 중간 결과 값으로 합과 올림수를 갖는 배열 곱셈기를 53 비트로 설계함으로써 하드웨어의 낭비를 없앴다. 하드웨어 모델을 Verilog-HDL로 64 비트 부동소수점 덧셈기와 곱셈기를 설계하여 검증하였다.

참고문헌

[1] Nhon Quach and Michael Flynn, "An improved algorithm for high-speed floating-point addition," Technical Report CSL-TR-91-501, Stanford University, Aug. 1990.

[2] Nhon Quach and Michael Flynn, "Design and implementation of the SNAP floating-point adder," Technical Report CSL-TR-91-501, Stanford University, Dec. 1991.

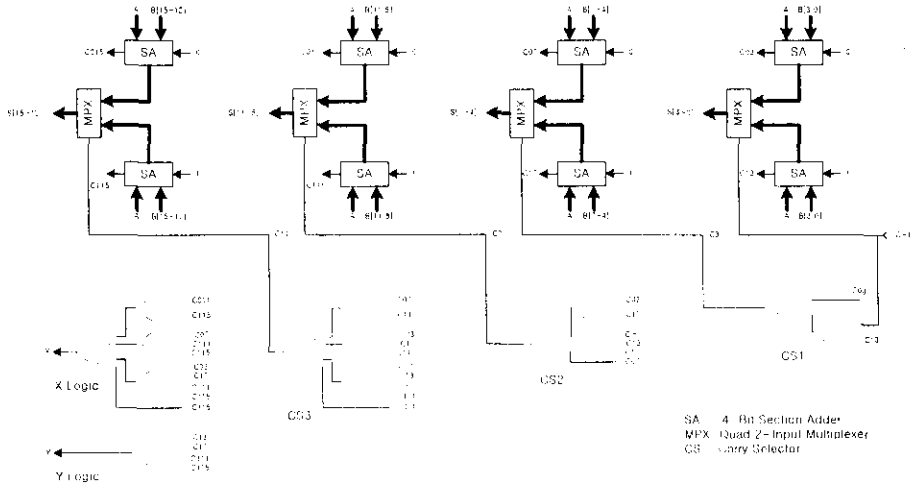
[3] Nhon Quach, "On fast IEEE rounding," Technical Report CSL-TR-91-459, Stanford University, Jan. 1991.

[4] 박우찬 외 3 명, "IEEE 반올림과 덧셈을 동시에 수행하는 부동 소수점 곱셈 연산기 설계," 전자공학회논문지, C 편, 제 11 호, pp. 47-55, 1997.

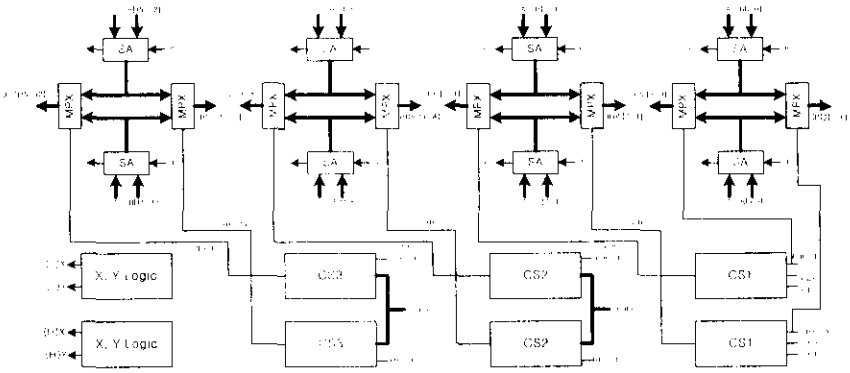
[5] 이위희, 강순우, "1998 년도 추계종합학술대회 논문집," 대한전자공학회.

[6] Kai Hwang, "Computer Arithmetic," John Wiley & Sons, 1979.

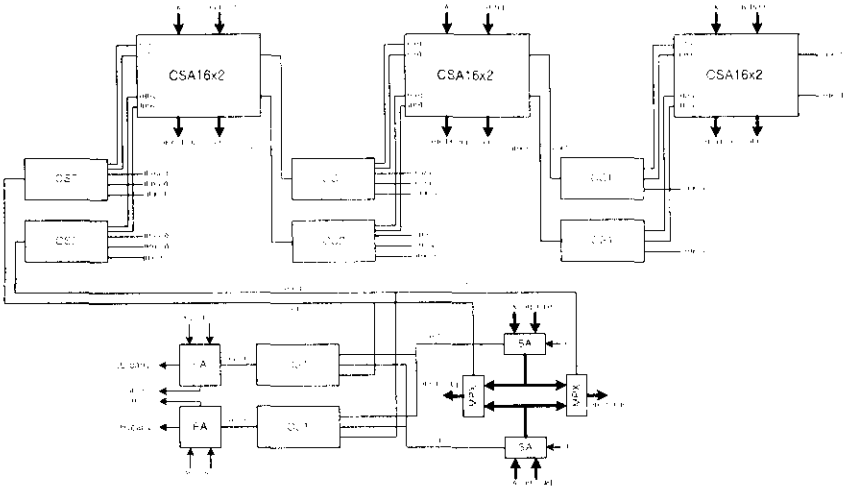
[7] IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Std. 754-1985, New York, The Institute of Electrical and Electronics Engineers, Inc., Aug., 1985.



(그림 3) 16 비트 CSA



(그림 4) 두 가지 결과 값을 갖는 16 비트 CSA



(그림 5) 두 가지 결과 값을 갖는 53 비트 CSA