Posit기반 실수 가산기 구현

임지환, 이유진, 전병철, *유호영
충남대학교 전자공학과
e-mail : *jhlim.cas@gmail.com, yjlee01.cas@gmail.com, bcjeon.cas@gmail.com, hyyoo.cnu@gmail.com*

Implementation of Real Number Adder Using Posit

Jihwan Lim, Yujin Lee, Byeongcheol Jeon, and *Hoyoung Yoo
Department of Electronics Engineering
Chungnam National University

Abstract

The Posit has been proposed as an alternative to overcome the limited range, overflow/underflow problem, which is a disadvantage of the IEEE 754 floating points. In this paper, we implement a 16-bit posit adder for each ES, and compare with IEEE 754 16-bit/32-bit floating-point adder for area, delay, power, and range. A 28nm 909MHz ASIC implementation, the max fraction bit size of floating point adder was 60.87% smaller than that of posit adder, but the numerical representation range per hardware resource was 68.86% larger.

I. 서론

Posit 숫자 체계는 John L. Gustafson이 IEEE 754 floating-point 숫자 체계의 단점을 보완하고자 대안으로 제안하였다[1]. Posit 숫자 체계가 가지는 이점으로는 동적 범위, 정확도, 오버플로우 및 언더플로우 극복 등이 있다[2]. 최근 딥러닝[3], HPC[4] 등 분야의 연구에서 Posit 숫자 체계를 사용한 연구가 기존 숫자 체계보다 효율적이라는 결과를 보여준다. Posit은 그림 1과 같이 부호비트(s), regime 비트(r), 지수비트(e), 가수비트(f)로 구성되며, N-bit의 Posit 형식으로 표현된 숫자 $Posit<N,ES>$는 수식 (1)과 같이 정의된다.

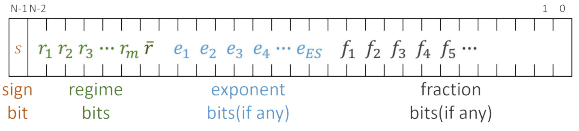

그림 1. Posit 숫자 체계 형식

$$Posit<N,ES> = (-1)^s \times 2^{c+k\times 2^{ES}} \times 1.f \quad (1)$$

$Posit<N,ES>$의 지수는 exponent bits로 표현되는 e와 regime bits의 연속된 r비트의 개수인 m으로 표현되는 k에 의해 동적으로 결정된다. k는 수식 (2)와 같이 regime bits의 최상위 비트인 r_1의 부호에 따라 다른 값으로 결정되며, regime bits로 표현되는 지수는

$$k=\begin{cases} m-1 & (r_0=1) \\ -m & (r_0=0) \end{cases} \quad (2)$$

floating-point와 다르게 bias를 가지지 않는다[1].
가수는 기존과 동일한 방식으로 결정된다.

본 논문에서는 posit 숫자 체계로 표현된 데이터를

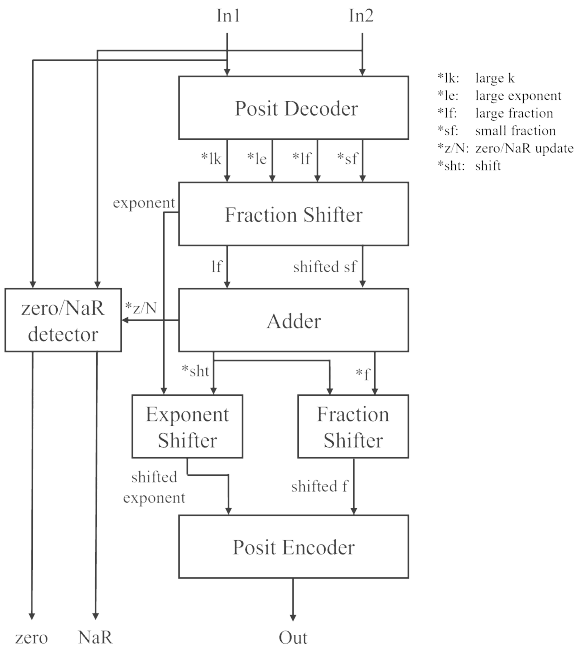

그림 2. Posit Adder 블록다이어그램

더하기 연산하는 posit adder의 하드웨어 설계와 구현에 대해 다루고, 구현된 하드웨어의 성능을 기존의 floating-point adder와 비교하여 평가한다.

II. 본론

하드웨어 수준에서 posit adder는 floating point adder와 유사하게 그림 2와 같이 decoder, adder, encoder로 설계된다. 주요한 차이점은 decoder에서 regime bits를 해석하는 회로가 추가된다는 점이다[5]. 본론에서는 posit adder의 하드웨어 구현을 decoder, adder, encoder의 세 부분으로 나누어 설명한다.

2.1 Posit decoder

Posit decoder는 posit 형식의 입력을 regime, exponent, fraction으로 디코딩하며 두 가지 과정으로 수행한다. 첫 번째로 두 입력의 zero/NaR(Not a Real) 여부를 확인하고, 더 큰 입력을 결정한다. Posit에서 zero는 모든 비트가 0임을 의미하며, NaR은 sign 비트만 1인 경우를 의미한다[1]. 두 번째로 regime을 디코딩하여 k를 결정하며 k에 따라 exponent와 fraction의 값을 결정한다. k는 입력의 연속되는 비트 중 1을 감지하는 LOD(Leading One Detect) 로직을 통해 결정하고[6], LOD 결과 정해진 regime의 범위에 따라 최종적으로 exponent bits와 fraction bits의 크기를 결정한다.

2.2 Posit adder

표 1. 형식별 하드웨어 리소스 비교

Format	Es/Exp Size	EGC	Delay	Power
Posit<16,1>	1	2437.29	1.08 ns	0.6331 mW
Posit<16,2>	2	2517.78	1.09 ns	0.6548 mW
Posit<16,3>	3	2514.91	1.09 ns	0.6521 mW
Posit<16,4>	4	2157.89	1.09 ns	0.5665 mW
IEEE-FLP16	5	936.24	1.08 ns	0.3368 mW
IEEE-FLP32	8	2806.08	1.08 ns	0.8645 mW

표 2. 형식별 숫자 표현 범위 비교

Format	Es/Exp Size	Dynamic Range
Posit<16,1>	1	3.73×10^{-9} to 2.68×10^{8}
Posit<16,2>	2	5.42×10^{-20} to 1.84×10^{19}
Posit<16,3>	3	1.93×10^{-34} to 5.19×10^{33}
Posit<16,4>	4	3.71×10^{-68} to 2.70×10^{67}
IEEE-FLP16	5	5.96×10^{-8} to 6.55×10^{4}
IEEE-FLP32	8	1.40×10^{-45} to 3.40×10^{38}

Posit adder는 디코딩 된 입력을 더하기 연산하는 역할을 수행한다. 더하기 연산을 수행하기 전 두 입력의 지수 차이에 따라 디코더에서 결정된 작은 입력의 fraction bits를 시프트하여 자리수를 맞춘다. 시프트된 작은 입력과 큰 입력에 대하여 더하기 연산을 수행한 후, 결과에 따라 다음 중 하나의 동작을 수행한다:

- 모든 비트가 0일 경우 zero or NaR 업데이트
- Carry가 존재할 경우 오른쪽으로 시프트
- 정수부가 0일 경우 왼쪽으로 시프트
- $1.f$ 형식일 경우 결과 출력

연산 결과를 왼쪽으로 시프트 하는 경우 fraction bits를 Posit의 fraction 표현 형식인 $1.f$으로 표현하기 위해 LOD 로직과 shifter를 사용한다.

2.3 Posit encoder

Posit encoder는 adder의 결과로 출력된 데이터를 다시 posit 형식으로 인코딩하며 세 가지 과정으로 수행한다. 첫 번째로 큰 입력의 지수에 adder 연산결과 시프트 된 지수를 반영한다. 두 번째로 시프트가 반영된 지수를 k와 exponent 값으로 분리한다. 세 번째 k값을 인코딩하여 regime bits를 결정하고, regime bits에 따라 exponent bits와 fraction bits를 결정한다.

III. 구현

형식에 따른 성능을 ASIC결과로 비교하기 위하여 단일 클럭에서 동작하는 Posit<16,1>, Posit<16,2>, Posit<16,3>, FLP16[7],[8], FLP32[7],[8] adder를 각각

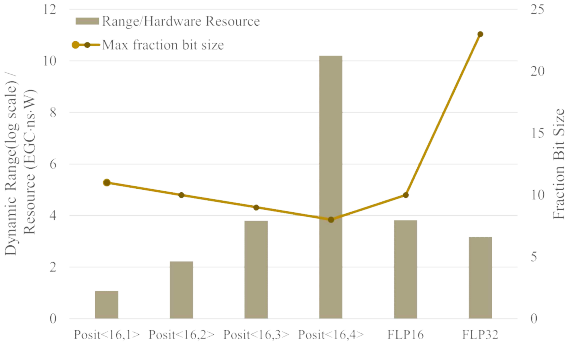

그림 3. 형식별 성능지표

구현하였으며, Synopsys Design Compiler를 사용하여 CMOS 28nm 공정 909MHz로 합성하였다. 표 1은 합성결과 각 형식별 하드웨어 리소스를 비교한 표이다. Posit과 FLP16/32를 area(EGC: Equivalent Gate Count), delay, power로 비교한 결과 FLP16이 가장 우수한 성능을 보였으나, 표 2의 숫자 표현 범위를 고려했을 때 Posit과 FLP32에 비해 매우 저조한 성능을 보인다. Posit 형식 중 가장 우수한 성능을 보이는 ES=4인 Posit<16,4> 형식은 그림 3과 같이 정밀도(Fraction Bit Size)가 FLP32보다 60.87% 떨어지나, 하드웨어 리소스 당 표현범위가 68.86% 크다.

Ⅳ. 결론 및 향후 연구 방향

본 논문에서는 16-bit posit adder를 ES 별로 구현하고 합성 결과를 기존 floating-point adder와 비교 분석하였다. 16-bit posit adder가 32-bit floating-point adder 보다 비트수의 한계로 인해 정밀도 측면에서 한계를 보였으나, 표현범위 및 하드웨어 성능이 우수하다. 결론적으로 하드웨어 수준에서 16-bit posit adder는 정밀도의 제약이 적은 환경에서 IEEE 754 32-bit floating-point adder의 유망한 대안으로 사용 가능하다.

ACKNOWLEDGEMENTS

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2022-0-01170, PIM 반도체 설계 연구센터), supported by National R&D Program through the National Research Foundation of Korea(NRF) funded by Ministry of Science and ICT(2020M3H2A1078119), and EDA tool was supported by the IC Design Education Center(IDEC), Korea.

참고문헌

[1] J. L. Gustafson and I. T. Yonemoto, “Beating floating point at its own game: Posit arithmetic,” Supercomputing Frontiers and Innovations, vol. 4, no. 2, 06 2017.

[2] M. K. Jaiswal and H. K. . -H. So, "PACoGen: A Hardware Posit Arithmetic Core Generator," in IEEE Access, vol. 7, pp. 74586-74601, 2019.

[3] H. Zhang, J. He and S. -B. Ko, "Efficient Posit Multiply-Accumulate Unit Generator for Deep Learning Applications," 2019 IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 2019.

[4] L. van Dam, “Enabling high performance posit arithmetic applications using hardware acceleration,” Master’s thesis, Delft University of Technology, the Netherlands, 2018.

[5] R. Murillo, A. A. Del Barrio and G. Botella, "Customized Posit Adders and Multipliers using the FloPoCo Core Generator," 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 2020.

[6] M. K. Jaiswal and H. K. . -H. So, "Architecture Generator for Type-3 Unum Posit Adder/Subtractor," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 2018.

[7] "IEEE Standard for Binary Floating-Point Arithmetic," in ANSI/IEEE Std 754-1985 , vol., no., pp.1-20, 12 Oct. 1985.

[8] R. Chaurasiya et al., "Parameterized Posit Arithmetic Hardware Generator," 2018 IEEE 36th International Conference on Computer Design (ICCD), Orlando, FL, USA, 2018.