AN IMPROVED FUSED FLOATING-POINT THREE-TERM ADDER

Mohyiddin K , Nithin Jose, Mitha Raj, Muhamed Jasim TK, Bijith PS, Mohamed Waseem P

ABSTRACT

A fused floating-point three term adder performs two additions in a single unit.It has better performance and accuracy compared to a network of discrete design. To improve the performance of three term adder, several optimization techniques are used. Brent kung adder is used in this architecture to achieve reduction in area. The proposed design is implemented for single precision and synthesized with a 45 nm cmos standard cell library.

INTRODUCTION

Addition is the most frequently used operation in many algorithms and applications. In case of the additions in series, however, a network of the two term adders loses accuracy due to the multiple roundings one after each addition. Many recent floating-point units can accommodate operations that have three inputs. Issues for the design of the fused floating-point three-term adder are (1)Complex exponent processing and significand alignment, (2)Complementation after the significand addition, (3)Large precision significand addition, (4)Massive cancellation management, and (5)Complex round processing. Those issues are addressed by investigating several optimization techniques in previous work. In this work the area is further reduced by replacing Kogge-Stone adder with Brent-Kung adder. The improved fused floating-point three-term adder will contribute to the next generation of floating-point arithmetic unit design.

OPTIMIZATION TECHNIQUES

1. EXPONENT COMPARE AND SIGNIFICAND ALIGNMENT

It is necessary to determine the largest exponent and aligned significands to handle the three operands. The traditional fused floating-point three-term adder determines the largest exponent based on the exponent differences. Then, the exponent differences are used for the significand alignment. This approach requires performing the exponent subtractions, complementation and significand shift sequentially, which takes a large latency. In order to reduce the latency, a new exponent compare and significand alignment logic is proposed. Six subtractions are performed to compute all the combinations of exponent differences ($\exp a - \exp b,\ \exp b - \exp a,\ \exp b - \exp c,\ \exp c - \exp b,\ exp\ a - \exp c,\ \exp c - \exp a$). In each pair of differences, an absolute value is selected based on the exponent comparison result, which enables skipping the complementation after the subtractions.

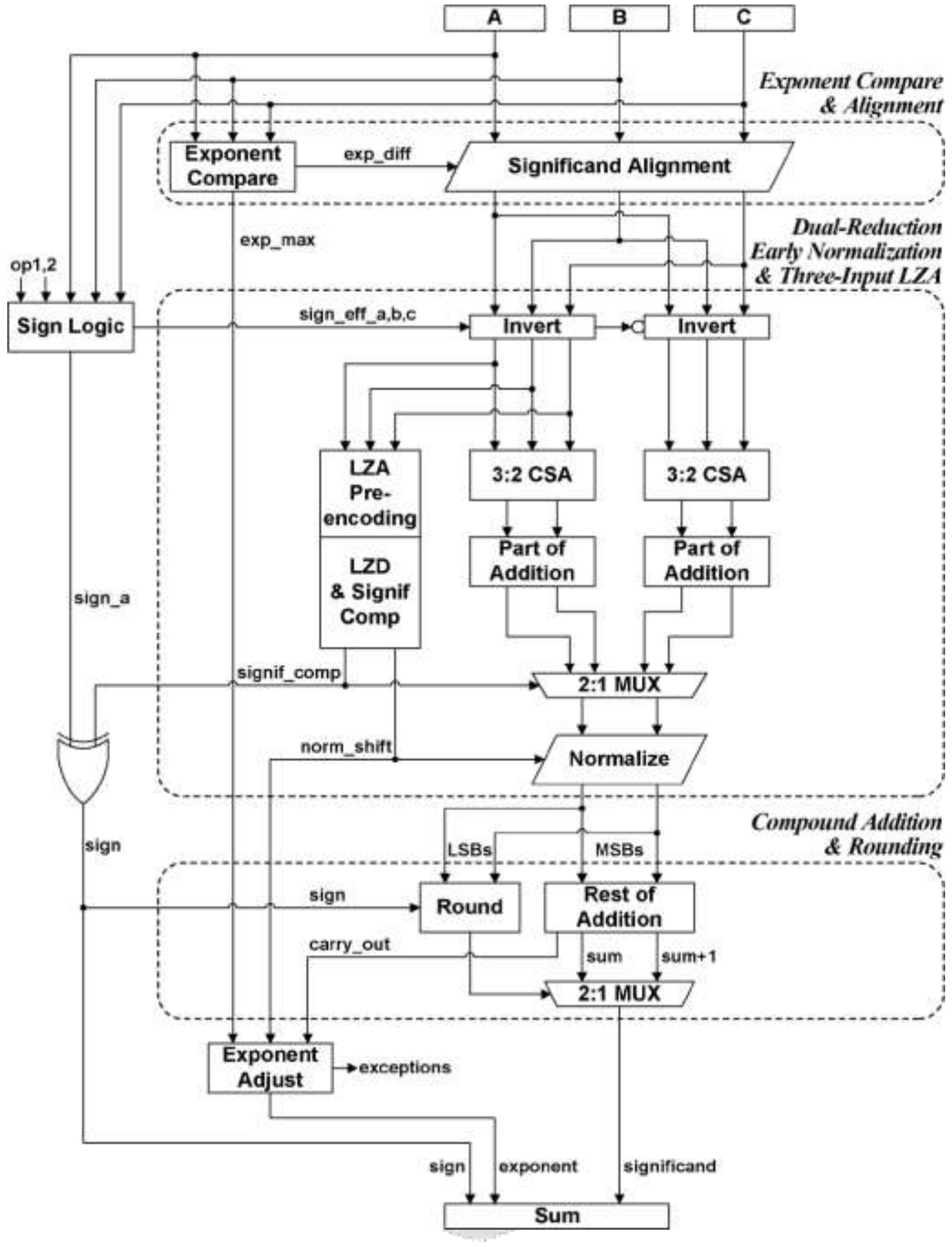

Fig.1 An improved fused floating-point three-term adder

2. INVERT AND DUAL-REDUCTION

The sign logic generates the three effective sign bits ($sign_{eff}a,\ sign_{eff}b\ and\ sign_{eff}c$) based on the three sign bits and two op codes as

$$sign_{eff}a = sign\ a$$

$$sign_{eff}b = sign\ a \oplus (sign\ b\ \oplus op1)$$

$$sign_{eff}c = sign\ a \oplus (sign\ c \oplus op2)$$

Where $op1$and $op2$ are the first and second op codes, respectively. The aligned significands are inverted based on the effective sign bits for the subtraction. The three operand subtraction requires that up to two significands are complimented. If all three operands are negative, they are added and the sign becomes negative, that is –A-B-C = -(A+B+C). In order to avoid the increments after the inverters, 2 bits are extended to the LSB of the significands that are propagated to the significand addition to handle the cases that one or two significands are inverted by effectively adding 1 or 2, respectively.

3. EARLY NORMALIZATION

One of the design issues for the fused floating-point three term adder is the high precision significand addition. The traditional fused floating-point three-term adder aligns the significands to 2f+6 bits, where f is the number of significand bits. Such large significands require a large significand addition and normalization, which the biggest bottleneck of the fused floating-point three-term adder. To reduce the overhead, early normalization is applied. The normalization is performed prior to the significand addition so that the significand adder size is reduced to f+1 bits. The rest of lower bits (f+7) are passed to rounding. By normalizing the significand pair prior to the significand addition, the round position is fixed so that the significand addition and rounding can be performed in parallel, which significantly reduces the latency of the critical path.

4. THREE-INPUT LZA AND SIGNIFICAND COMPARISON

Since the normalization is performed prior to the significand addition, the LZA and normalization is on the critical path. To use a traditional two-input LZA, the three significands need to be reduced to two using a 3:2 CSA, which increases the delay. The three-input LZA encodes the three inputs at once to skip the delay of the 3:2 CSA. The three-input LZA can be implemented by extending the traditional two-input LZA. Like most of the LZAs, the three-input LZA consists of two parts: 1) Pre-encoding indicator vectors and 2) Leading Zero Detection (LZD) logic for generating the leading zero count.

5. COMPOUND ADDITION AND ROUNDING

The normalized significand pair is passed to the significand addition and rounding. As described above, the upper significand bits are used for the addition and the lower bits are used for the rounding.

In the two-term adder, only one significand is shifted for the alignment so that there is no carry propagation for the lower part.

The compound addition determines the upper f bits including possible two overflow bits, and the rounding determines the rest of three LSBs and the round decision. The upper f+1 bits are passed to the compound addition, which produces sum and sum+1 simultaneously. Two significand sums are right shifted by up to 2 bits based on the overflow bits. Then, one correct significand sum is selected based on the round decision.

6. BRENT KUNG ADDER

The earlier fused floating point adders used Kogge-Stone adder. In this paper, the Kogge-Stone adder is replaced with the Brent-Kung adder inorder to reduce the adder size Brent- Kung is a parallel prefix form of the carry look ahead adder. It takes less area to implement than the other prefix adders such as Kogge- Stone adder and it also has less wiring congestion. Instead of using a carry chain to calculate the output, the method shown in Fig.2 is used.

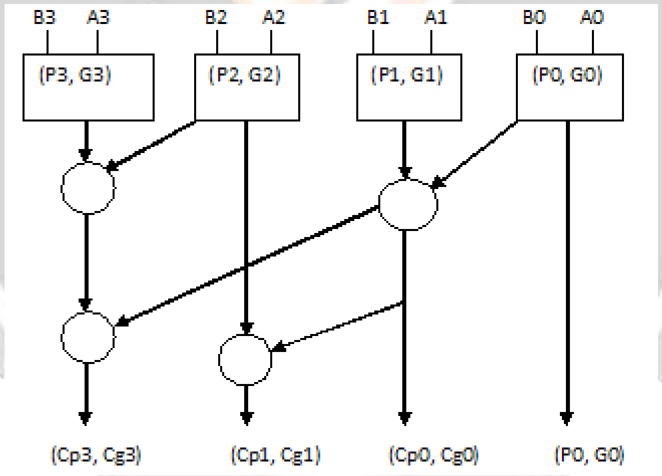

Fig.2 4-bit Brent-Kung Adder

This will reduce the delay without compromising the power performance of the adder. Replacing the Kogge-Stone adder with the above structure of Brent-Kung adder in the part of addition we can see that there is significant reduction in area.

RESULTS AND COMPARISON

The proposed design is implemented for single precision in Verilog-HDL and synthesized with the Nangate 45 nm CMOS technology standard cell library. In order to evaluate the improvement of the proposed design, the area and delay are compared with the previous designs. Depending on the target frequency, the implementations are synthesized with different area and delay. Table 1 compares the logic utilization of Kogge-stone adder and Brent-kung adder. Even though the delay for both adder is same, the area covered is smaller for Brent-kung adder. Thus the power consumption also become lower. The proposed design has a smaller significand adder size compared to the traditional designs.

Table 6.1 Logic utilization comparison

Logic utilization	**Kogge-Stone adder**	**Brent-Kung adder**
Number of slice registers	59	59
Number of slice LUTs	1028	923
Number of fully used LUT-FF pairs	59	59
Number of bonded IOBs	1658	1658
Delay	14.387 ns	14.387

CONCLUSION

The improved architecture design and implementation for a fused floating-point three-term adder has been presented. There are several critical design issues for the fused floating-point three-term adder: 1) Complex exponent processing and significand alignment, 2) Complementation after the significand addition, 3) Large precision significand adder, 4) Massive cancellation management, and 5) Complex round processing. To resolve those issues, several algorithms and optimization techniques are applied. For further improvement, Kogge-stone adder is replaced with Brent-kung adder which reduces the area.

REFERENCES

[1] Jongwook Sohn, Earl E. Swartzlander Jr.,(2014) "A Fused Floating-Point Three-Term Adder", *IEEE transactions on circuits and systems*: regular papers, vol. 61, no. 10, october

[2] Pappu P. Potdukhe, Vishal D. Jaiswal(2015) *"Review of carry select adder by using brent kung adder"* IJARECE Volume 4, Issue 10, October.

[3] J. Sohn and E. E. Swartzlander, Jr., (2013)"Improved architectures for a floating-point fused dot product unit," in *Proc. 21st Symp. Computer Arithmetic*, pp. 41–48

[4] J. Sohn and E. E. Swartzlander, Jr.,(2012) "Improved architectures for a fused floating-point add-subtract unit," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 10, pp. 2285–2291, Oct

[5] A. Tenca, (2009)"Multi-operand floating-point addition," in *Proc. 21st Symp. Computer Arithmetic*, pp. 161–168.

[6] H. H. Saleh and E. E. Swartzlander, Jr., (2008)"A floating-point fused addsubtract unit," in *Proc. 51st IEEE Midwest Symp. Circuits Syst.*, pp. 519–522.

[7] H. H. Saleh and E. E. Swartzlander, Jr.,(2008) "A floating-point fused dotproduct unit," in *Proc. IEEE Int. Conf. Computer Des.*, pp. 427–431.

[8] T. Lang and J.D. Bruguera,(2004) "Floating-point fused multiply-add with reduced latency," *IEEE Trans. Computers*, vol. 53, pp. 988–1003.

[9] P. M. Seidel and G. Even,(2004) "Delay-optimized implementation of IEEE floating-point addition," *IEEE Trans. Computers*, vol. 53, no. 2, pp. 97–113, Feb.

[10] S. F. Oberman, H. Al-Twaijry, and M. J. Flynn, (1997) "The SNAP project: Design of floating point arithmetic units," in *Proc. 14th IEEE Symp.Computer Arithmetic*, pp. 156–165.

[11] E. Hokenek, R. K. Montoye, and P. W. Cook,(1990**)** "Second-generation RISC floating point with multiply-add fused," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1207–1213, X.

[1**2] R. K. Montoye, E. Hokenek, and S. L. Runyon,** (1990) "Design of the IBM RISC system/6000 floating-point execution unit," *IBM J. Res. Develop.*, vol. 34, pp. 59–70.

[13] M. P. Farmwald,(1981) "On the Design of High Performance Digital Arithmetic Units," Ph.D. dissertation, Computer Science, Stanford University, Stanford, CA, USA.