ASIC Implementation of Inexact Floating Point Adder

S. Kalaiselvi [1]
Department of Electronics and communication
Dr. Mahalingam college of engineering & technology
Pollachi, Tamilnadu, India

N. Sheela [2]
Department of Electronics and communication
Dr. Mahalingam college of engineering & technology
Pollachi, Tamilnadu, India

M. Hariesh Kumar[3]
Department of Electronics and communication
Dr. Mahalingam college of engineering & technology
Pollachi, Tamilnadu, India

V. Gowthaman [4]
Department of Electronics and communication
Dr. Mahalingam college of engineering & technology
Pollachi, Tamilnadu, India

Abstract:- **In deep submicron CMOS technology, power is a primary design constraint and it has skyrocketed due to more on-chip transistor count and high clock frequencies. To reduce power consumption, approximate circuit implementations have been considered as a potential solution in which strict exactness is not required. This paper presents the design and analysis of the inexact floating - point adder in TSMC 180nm technology. The proposed adder uses an inexact addition algorithm for mantissa addition and high speed circuitry for normalization. Performance comparisons with Exact Floating Point Adder reveal that the proposed Inexact Floating Point Adder(IFPA) occupies less area and consumes less power. The proposed IFPA is designed using Cadence Virtuoso 64 design environment and simulated using Cadence Spectre. The proposed inexact FPA finds extensive applications in image processing and error-tolerant applications where accuracy is not a prime concern..**

Keywords : Lower part Or addition ,CMOS, floating point adder, Dot product unit, mantissa.

1 INTRODUCTION

In present day portable applications, speed and power dissipation are the main important consideration in submicron CMOS design. The existing floating point adder requires more number of Transistors and consumes more power to get an accurate result. Floating-point addition is the most frequent floating-point operation and accounts for almost half of the scientific operation. Therefore, it is a fundamental component of math coprocessor, DSP processors, embedded arithmetic processors, and data processing units. These components demand high numerical stability and accuracy and hence are floating- point based. Floating-point addition is a costly operation in terms of hardware and timing as it needs different types of building blocks with variable latency.The approximate multiplier with a lower power consumption and a shorter critical path than traditional multipliers is proposed for high-performance DSP applications was investigated in the paper[1] was proposed by Cong Liuaetall. Then the conventional digital hardware computational blocks with different structures are designed to compute the precise results of the assigned calculations was proposed by H. R. Mahdianietall [3]. Cong Liuaetall [2] proposed that the Approximate adders have been considered as a potential alternative for error-tolerant applications to trade off some accuracy for gains in other circuit-based metrics, such as power, area and delay. Existing approximate adder designs have shown substantial advantages in improving many of these operational features. However, the error characteristics of the approximate adders still remain an issue that is not very well understood

Fang Fanget all (2011) proposed a a lightweight FP design flow that can optimize the bit-width configuration.To enable floating-point (FP) signal processing applications in low-power mobile devices, we propose a lightweight FP design flow that can optimize the bit-width configuration. The optimization considers both the hardware cost and the numerical precision. In addition, promising results on some real-world applications such as video coding and speech recognition demonstrate that lightweight FP signal processing will find more and more applications in low-power devices.[4]

This paper is organized as follows.Section 2 provides background on Fixed point and floating point unit .Section 3 presents the design of the exact and inexact FP adders. Section 4 provides the information about the performance analysis of both inexact and exact floating point adder. Section 5 provides the application of the designed inexact floating point adder in the dot product unit. Finally, a conclusion is provided in Section 6.

2 BACKGROUND

2.1 FIXED POINT UNIT

A fixed point number has a specific number of bits (or digits) reserved for the integer part (the part to the left of the decimal point) and a specific number of bits reserved for the fractional part (the part to the right of the decimal point). No matter how large or small your number is, it will always use the same number of bits for each portion. For example, if your fixed point format was in decimal IIIII.FFFFF then the largest number you could represent would be 99999.99999 and the smallest would be 00000.00001. Every bit of code that processes such numbers has to have built-in knowledge of where is the decimal point.

2.2 FLOATING POINT UNIT

The FP format typically contains a sign bit, the exponent and the mantissa fields (commonly represented as a string from left to right). It offers a higher dynamic range than a fixed-point format to represent real numbers. However, the FP hardware is both more complex and consumes significant power. A floating point number does not reserve a specific number of bits for the integer part or the fractional part. Instead it reserves a certain number of bits for the number (called the mantissa or significand) and a certain number of bits to say where within that number the decimal place sits (called the exponent). So a floating point number that took up 10 digits with 2 digits reserved for the exponent might represent a largest value of 9.9999999 e+50 and a smallest value of 0.0000001 e-49.

Table 2.1 No. of Exponent and Mantissa Bits for the IEEE 754 Basic and Extended FP Types

Type	Sign Bit	Exp. Bits	Mant. Bits	Total	Mant. Bits/Total
Half	1	5	10	16	62.5%
Single	1	8	23	32	71.9%
Double	1	11	52	64	81.2%
Extended	1	15	64	80	80.0%
Quad	1	15	112	128	87.5%

3 FLOATING POINT ADDERS

An adder, also called summer, is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic units, but also in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators, and similar operations. Although adders can be constructed for many numerical representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or one'scomplement is being used to represent negative numbers, it is 5 trivial to modify an adder into an adder–subtractor. Other signed number representations require more logic around the basic adder. The description of the exact floating point adder are as follows:

- Both the mantissa and the exponent are unsigned
- Assume both inputs are normalized .before performing the actual addition of the two mantissa they must aligned.if the difference between the exponent is n,then the mantissa of the smaller exponent must be shifted right by n,thus the bits being added will have the same weight.
- Note that the exponent of the result is not the sum of the two input exponents. Adding one exponent to another results in multiplication.
- If after adding the mantissa there is overflow,the exponent must be adjusted accordingly and the 5 most significant bits of the mantissa must be retained.
- Building a saturating adder,meaning that if the result of the addition , meaning that if the result of the addition exceeds the maximum allowed number. It should output the maximum(1111 1111) .

3.1 DESIGN OF EXACT FLOATING POINT ADDER

The existing floating point adder requires more number of Transistors and consumes more power to get an accurate result. Floating-point addition is the most frequent floating-point operation and accounts for almost half of the scientific operation. Therefore, it is a fundamental component of math coprocessor, DSP processors, embedded arithmetic processors, and data processing units.These components demand high numerical stability and accuracy and hence are floating- point based. Floating-point addition is a costly operation in terms of hardware and timing as it needs different types of building blocks with variable latency. In floating-point adder implementation, latency is the overall performance bottleneck. A lot of work has been done to improve the overall latency of floating-point adders.

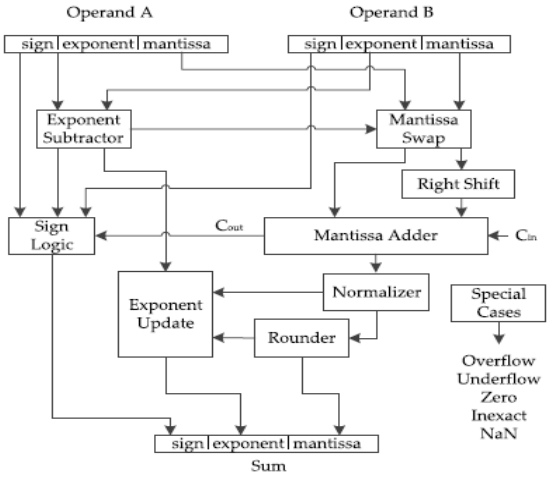

Fig 3.1 Exact floating point adder architecture

3.2 DESIGN OF INEXACT FLOATING POINT ADDER

Traditional designs apply fully accurate computing to all types of applications; however, error-tolerant applications involving human intervention do not require full accuracy. So, it is possible to perform computation with inexact circuits; in these cases, inexact computing is an attractive approach to save power and area, it replaces the mantissa swap circuit with the 2x1
multiplexers. so the number of gates required for the inexact is very less and power consumption also very small and the Accurate floating point adder (FPA) has a rounder to round off the value at the end but in proposed inexact floating point adder (FPA) the rounder block was eliminated

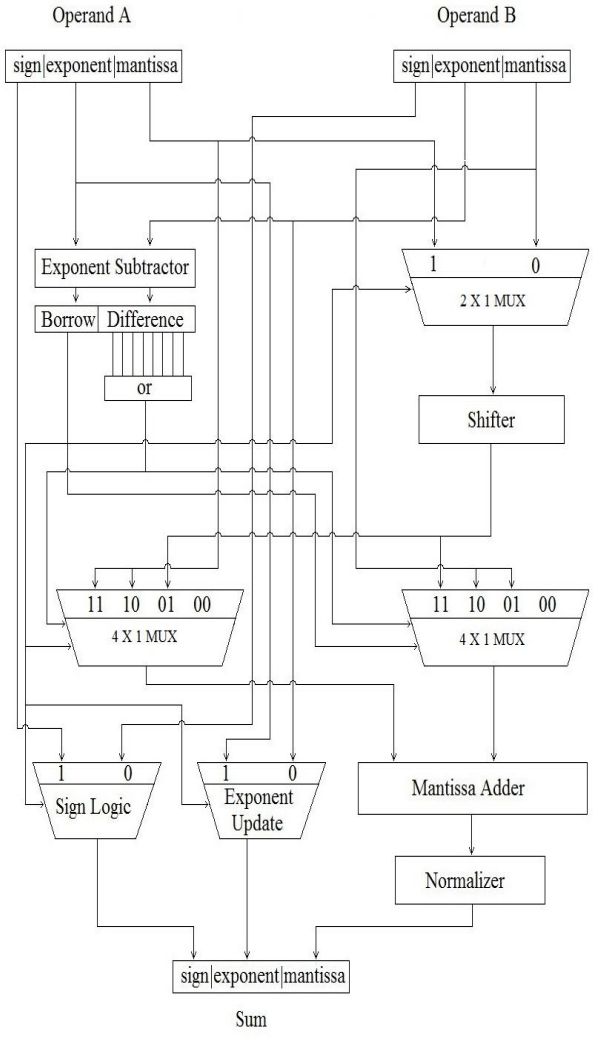

Fig 3.2 Inexact floating point adder architecture

3.2.1 MANTISSA SWAP

Instead of mantissa swapping circuit in the accurate floating point adder architecture, some changes has been made in proposed architecture which contains 2x1 multiplexer, 4×1 multiplexer and shifter. The output of the difference will contain 8 bits. The last 4 bits i.e., Least significant bits(LSB) will act as a selector signal for the shifter which is designed with twenty three 4×1 multipliers of 4 stage. This shifter will shift up to 15 bits in the mantissa .The 8 bits in the difference will get added by using OR gate and changed as a single control signal which will act as a selector for those two 4x1 multiplexer in the proposed architecture.

1.If A is less than B, then the borrow will be 0 and difference will be some value. In this case operand A's mantissa bits will get shift and loaded into corresponding multiplexer. Finally the multiplexer output will go to mantissa adder.

2.If A is greater than B,then the borrow will be 1 and difference will be some value. In this case operand A's mantissa bits will get shift and loaded into corresponding multiplexer. Finally the multiplexer output will go into mantissa adder

3.If A=B, then the borrow and difference will become zero. Then no shift will takes place.

3.2.2 MANTISSA ADDER

The revised LOA adder can be used in the mantissa adder for an inexact design. Further, the inexact design in the mantissa adder has a lower impact on the error than its exponent counterpart in the lower data range, because the mantissa part is less significant than the exponent part. Therefore, an inexact design of a mantissa adder is more appropriate. The total number of mantissa bits is 23. In this design the first 11 most significant bits (MSB) are added by using full adder and last 12 least significant bits (LSB) are added by using OR gates. The mantissa adder is also designed by using lower -part OR adder (LOA) method. The carry is generated from a_{11} and b_{11} by using AND gate and passed to a_{12} and b_{12} .By using this kind of method there is a possibility to reduce the area and power of the adder.

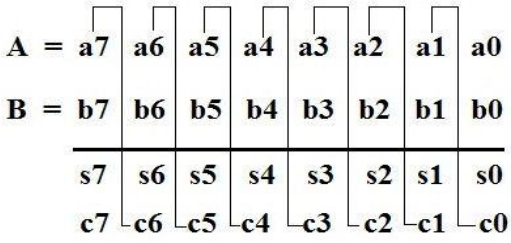

Fig 3.3 Exact Mantissa Addition

In exact floating point adder ,all the 23 bits are added using the full adder but in the exact mantissa adder,the last 11 bits are added using OR gate and the remaining bits are added using the full adder.So the requirement of the logic gates get reduces in inexact floating point adder.

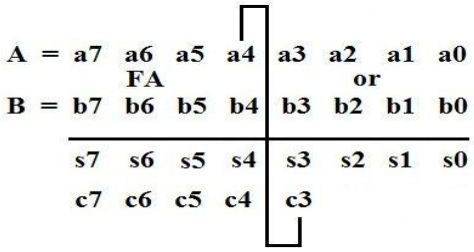

Fig 3.4 Inexact Mantissa Addition

4 PERFORMANCE

The proposed floating point adder is designed using CADENCE VIRTUOSO 64 TOOL with TSMC 180nm technology file .The performance results in terms of area, delay and power dissipation of proposed floating point adder are shown in Table 5.1

Table 4.1 performance table

Floating Point adder	Delay(ns)	Power(mw)	Area(μm²)	PDP
Exact	4.53	0.4132	8563.18	1.8717
Existing	4.38	0.2768	5485.23	1.2123
Proposed	3.28	0.2579	5154.58	0.8459

From the Table 4.1 it is seen that the proposed floating point adder demonstrates the delay reduction of 27.34 % and 25.56% compared to exact and exisisting floating point adder. This is due to the modification in the inexact floating point adder architecture and elimination of mantissa swap circuit, rounder which reduces the number of gates and delay.the last 11 bits of mantissa are get added up by using Or gate.so the number of gates required to build the or gate got reduced compared to full adder.therefore,it provides more efficiency compared to existing and exact FPA.

5 IMPLEMENTATION OF INEXACT FLOATING POINT ADDER IN DOT PRODUCT UNIT

The proposed inexact floating point adder is used in the design of calculating fused floating point four term dot product which is mostly used in the wide variety of graphics, digital signal processing (DSP) such as complex multiplication and fast Fourier transform (FFT) and discrete cosine transform (DCT) butterfly operations and statistical applications. The proposed design computes the four-term dot product in a single unit to achieve better performance and accuracy compared to a network of traditional floating-point multipliers and adders, which is very flexible. As a result, the area, latency and power consumption are reduced compared to the discrete design.

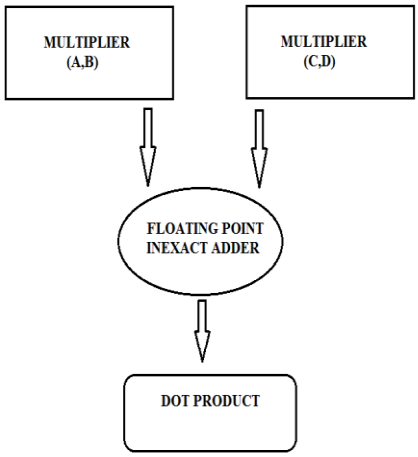

Fig 5 Block Diagram of Fused floating point four term Dot product unit

It demonstrates the use of the Multiply and Add functions to perform the dot product. The dot product of two vectors is obtained by multiplying corresponding elements and summing the products. The floating point four-term dot product unit takes eight floating point numbers and computes the sum or difference of two products as

$$Z = AB \pm CD \pm EF \pm GH$$

6. CONCLUSION

The proposed work developed an approach for design and implementation of high speed and low power approximate FPA suitable for digital image processing applications . Extensive comparisons shows that the proposed inexact floating point adder performs well .this suggests the suitability of proposed FPA for high speed portable VLSI implementation

7 FUTURE SCOPE

The future scope of the proposed architecture using inexact design can include following work.The proposed design consumes low power and reduced critical path ,so it will be implemented in real time Digital image processing application

REFERENCES

[1] C. Liu, J. Han, and F. Lombardi, "A low- power,high-performance approximate multiplier with configurable partial error recovery," in Proc. Design,Autom. Test Eur. Conf. Exhib., 2014, pp. 1–4.

[2] C. Liu, J. Han, and F. Lombardi, "An analytical framework for evaluatingthe error characteristics of approximate adders," *IEEE Trans. Comput.,vol. 64, no. 5, pp. 1268–1281, May 2015.*

[3] H. Mahdiani, A. Ahmadi, S. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft computing applications," *IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57,no. 4, pp. 850–862, Apr. 2010.*

[4] F. Fang, T. Chen, and R. Rutenbar, "Floating-point bit-width optimization for low-power signal processing

applications," in *Proc. IEEE Int. Conf.Acoust.*, Speech, Signal Process., 2002, vol. 3, pp. 3208–3211.

[5] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Trans. Comput., vol. 62, no. 9,pp. 1760–1771, Sep. 2013.*

[6] J.Y. Tong, D. Nagle, and R. Rutenbar, "Reducing power by optimizing thenecessary precision/range of floating-point arithmetic," *IEEE Trans. VeryLarge Scale Integr. Syst*., vol. 8, no. 3, pp. 273–286, Jun. 2000

[7] A. Gupta, S. Mandavalli, V. Mooney, K. Ling, A. Basu, H. Johan, and B. Tandianus, "Low power probabilistic floating-point multiplier design," in Proc. IEEE Comput. Soc. Annu. Symp. VLSI, 2011, pp. 182–187.

[8] J. Eilert, A. Ehliar, and D. Liu, "Using low precision floating point numbers to reduce memory cost for MP3 decoding", in Proc. 6th IEEE Workshop Multimedia Signal Process., 2004, pp. 119–122.

[9] W. Liu, L, Chen, C. Wang, M. O'Neill, and F. Lombardi, "Inexact floating-point adder for dynamic image processing," in Proc. 14th IEEE Conf. Nano-technol., 2014, pp. 239–243.

[10] IEEE Standard for Floating-Point Arithmetic, IEEE Std 754-2008, Aug. 29,2008.

[11] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs. London,U.K.: Oxford Univ. Press, 2009

[12] R. Mantiuk, K. Kim, A. Rempel, and W. Heidrich, "HDR-VDP-2: A cali-brated visual metric for visibility and quality predictions in all luminance conditions," ACM Trans. Graph., vol. 30, article 40, 2011.

[13] V. Gupta, D. Mohapatra, S. Park, A. Raghunathan, and K. Roy, "IMPACT: IMPrecise adders for low-power approximate computing," in Proc. Int.Symp. Low Power Electron. Des., 2011, pp. 1–3.

[14] Z. Yang, A. Jain, J. Liang, J. Han and F. Lombardi, "Approximate XOR XNOR-based adders for inexact computing," in Proc. 13rd IEEE Conf. Nano-technol., 2013, pp. 690–693.

[15] D. Mohapatra, V. Chippa, A. Raghunathan, and K. Roy, "Design of voltage-scalable meta-functions for approximate computing," in Proc. Des., Autom. Test Eur. Conf. Exhib., 2011, pp. 1–6.