International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 24 Issue 2, FEB, 2024

Floating Point FFT algorithm using Parallel Prefix Multiplier

Prabha A. G, Sweatha V, Vanisha R and Poornisha L[4]
Dept. of ECE
Rajalakshmi engineering College, Chennai, India

***ABSTRACT*: The communication system operations, mainly deal with signal processing, are largely depend on efficient Fast Fourier Transform (FFT) algorithm. There are number of FFT algorithms developed in recent years. However, there is a need for a high throughput and low latency at less cost FFT co-processor for signal processing applications. The basic components of any FFT algorithm are complex multipliers and complex adders in the form of radix-2 or radix-4 butterfly structure. Both the fundamental structures operate very fast while dealing with fixed point arithmetic, but slow in the case of floating point (FP) arithmetic which is more prevalent now. To combat the slowness effect of FFT, this paper propose a modified FP butterfly architecture with parallel prefix (PP) multiplier which provide fast operation in floating point arithmetic. Also the FP butterfly FFT with PP multiplier gives better performance than unmodified FP FFT.**

Index terms: Butterfly, complex number, Fast Fourier Transform floating point, Parallel prefix multiplier.

I. INTRODUCTION

There are several consecutive multipliers and adders over complex numbers in Fast Fourier transform (FFT) circuitry. So number representations which are appropriate must be chosen wisely. The fixed- point arithmetic is used in most of the FFT architectures. Recently, FFTs depends on floating-point (FP) operations are growing today[1][2]. The wide dynamic range is the main advantage of FP over fixed - point arithmetic; but it is at the expense of higher cost. Moreover, use of IEEE-
754-2008 standard for FP arithmetic allows for an FFT coprocessor in collaboration with general purpose processors [3].

The Main drawback of the FP operations is their slowness in comparison with the fixed-point counterparts. A way to speed up the FP arithmetic is to merge several operations in a single FP unit. Hence delay, area, and power consumption are reduced. Redundant number system is other well-known way of overcoming slowness of FP, where there is no word-wide carry propagation within the intermediate operations. A number system, defined by a radix r and a digit-set $[\alpha, \beta]$, is redundant iff $\beta - \alpha + 1 > r$.[4].

The transformation from non-redundant, to a redundant format is a carry-free operation, hence, the reverse conversion requires carry propagation. This makes redundant representation more useful where several consecutive arithmetic operations are performed prior to the final result. This proposes a butterfly architecture utilizing redundant FP arithmetic, useful for FP FFT coprocessors and provides to the applications of digital signal processing. While, there are other works on the usage of redundant FP number systems but they are notoptimized for butterfly architecture in which both redundant FP multiplier and adder are required. The novelties and techniques used in the proposed design include the following.

1) Design of all the significands are represented in binary signed digit (BSD) format and the corresponding carry-limited adder.
2) Design of FP constant multipliers for operands with BSD significands.
3) FP three-operand adders for operands with BSD significands are designed.
4) FP fused-dot-product-add (FDPA) units for operands with BSD significands are designed.

II. LITERATURE REVIEW

FFT is one of the important but computationally intensive operations in any radio communication and multimedia system. It occupies a major part of the baseband processor for operations. Shashank

Mittal et. al. , attempted to find, analyze and design efficient FFT computation units to reduce area, computational delay and power consumption in [5].Bertil Schmidt and Manfred schimmler presented a concept of the parallel computer model and the architectural details of the processor design in which they proposed that the computer model that allowed for efficient implementation of parallel prefix computations. In their finding they have depicted that how parallel prefix computations can be used as key operations in for deriving efficient implementations on the KPROC [6].Clyde P. Kruskal, et. al., presented a parallel algorithm to solve the prefix computation problem with linear speed up and used only shared memory location exclusively read or written [7.NishaLaguri and K. Anusudha proposed an efficient algorithm for Fast Fourier Transform using distributed arithmetic parallel prefix adder where all the complex multiplications and additions were performed by distributed arithmetic and parallel prefix adder respectively [8].J. H. Min et. al. proposed very fast FP multiplier by merging the fused FFT butterfly architecture into multiple constant multipler [9].

The rest of this paper is organized as follows. Section III is devotedto the existing redundant FP multiplier with carry propagation adder while the proposed model of redundant FP multiplier with parallel prefix multiplier is discussed in Section IV. Finally, the result and conclusion is drawnin Section V and IV respectively, followed by references.

III. REDUNDANT FP MULTIPLIER

Redundant Floating-Point Multiplier:

The Existed multiplier, contains two major steps, partial product generation (PPG) and PP reduction (PPR). However, contrary to the conventional multipliers, our multiplier keeps the product in redundant format and hence there is no need for the final carry-propagating adder. The exponents of the input operands are taken in the same way as done in the conventional FP multipliers.However,normalizationand rounding are left to be done in the next block of the butterfly architecture i.e., three-operand adder.

***a*) Partial Product Generation:**

The PPG step of the proposed multiplier is completely different from that of the conventional one because due to its representation of the input operands. Moreover, given that *Wre* and *Wim* are constants, the multiplications in Fig. 1 (over significands) can be computed through a series of shifters and adders.

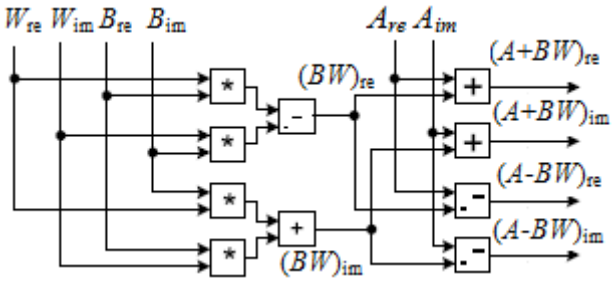

Fig. 1Butterfly architecture of FFT

With the intention of reducing the number of adders, we store the significand of *W* in modified Booth encoding. Fig. 2 shows the required circuitry for the generation of $PP_i$where each PP consists of $(n + 1)$ digits i.e., binary position[10].

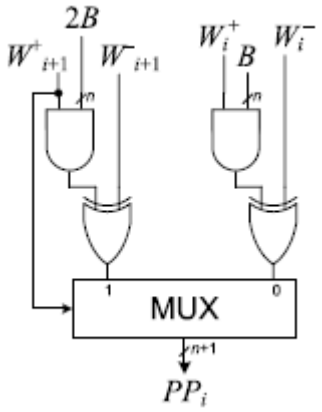

Fig. 2 Generation of i^{th} PP

***b*) Partial Product Reduction:**

In this PPR step major component is the carry-limited addition over the operands which are represented in BSD format.The length of the final product may be more than $2n$.Figure 3 shows the redundant FP multiplier which comprises three operand adder [11]-[14].

IV. PP MULTIPLIER

In Parallel Prefix adders the execution of an operation is in parallel. This is done bysegmentation the operation in smaller pieces which are computed in parallel. The output is depends on the initial inputs. Parallel Prefix Adder (PPA) is equivalent to carry look ahead

adder (CLA). A Carry look ahead adder is a type of adder used in digital logic. CLA is designed to overcome the latency introduced by repelling effect of carry bits in RCA. A CLA improves speed by reducing carry bits. It calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of larger bit value.
CLA uses the concept of generating (G) and propagating (P) carries. These two are differ in the way their carry generation block is implemented. The main advantage of PPA is the carry reduces the number of logic levels by essentially generating the carries in parallel. PPA fastest adder with focus on design time and is the choice for high performance adder in industry.
The proposed work focuses on two new techniques for reducing the hardware overhead and increasing the error correction capability. The technique analyzed in the previous work has certain limitation due to the complexity of handling larger number of FFTs. Partial summation is used for calculating its parity at the input and the output side of the FFT. It sums all possible node values of 4-point FFT along with the twiddle factors.

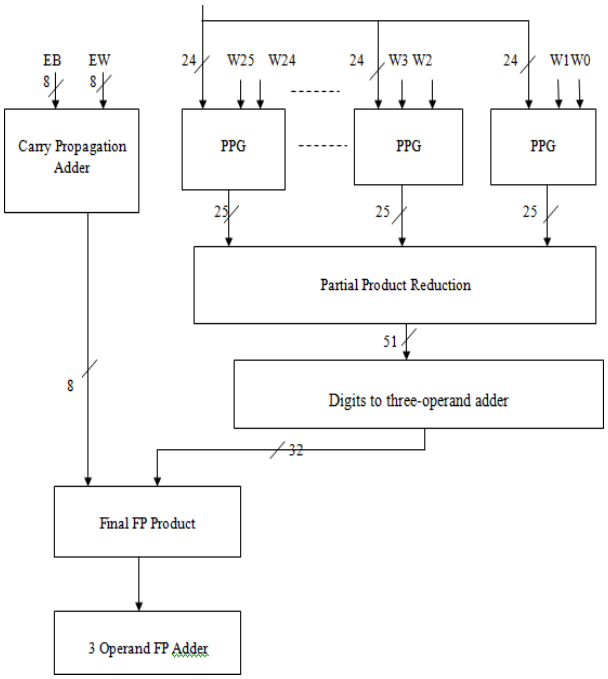

Fig. 3 Redundant FP multiplier with CP adder.

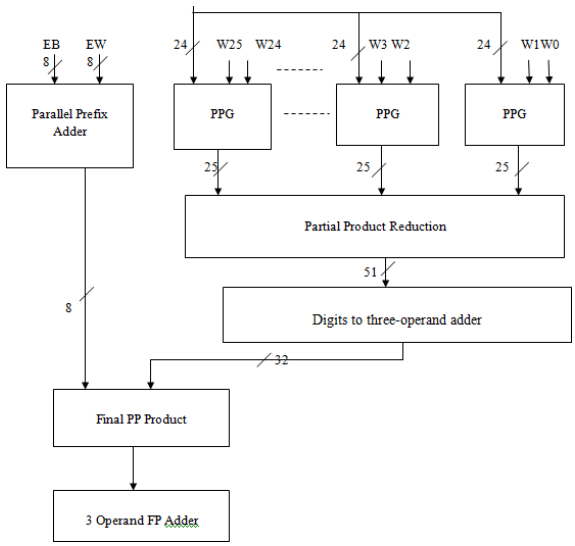

Fig. 4 Redundant FP multiplier with PP adder.

V. RESULTS

In Parallel Prefix adders the execution of an operation is in parallel. This is done by segmentation the operation in smaller pieces which are computed in parallel. The output is depends on the initial inputs. Parallel Prefix Adder (PPA) is equivalent to carry look ahead adder (CLA). A Carry look ahead adder is a type of adder used in digital logic. CLA is designed to overcome the latency introduced by repelling effect of carry bits in RCA. A CLA improves speed by reducing carry bits. It calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of larger bitvalue depicted in Fig. 5.

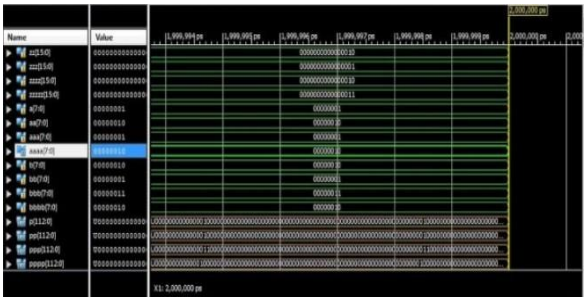

Fig. 5 RTL Schematic: Output waveform.

The proposed work focuses on two new techniques for reducing the hardware overhead and increasing the error correction capability. The technique analyzed in the previous work has certain limitation due to the complexity of handling larger number of FFTs. Partial summation is used,shown in Fig. 6, for calculating its parity at the input and the output side of the FFT. It sums all possible node values of 4-point

FFT along with the twiddle factors.

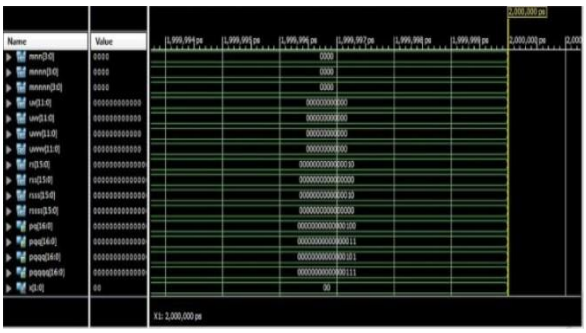

Fig. 6 Simulated output

VI. CONCLUSION

A high-speed FP butterfly architecture is proposed in this paper which is comparatively faster but at the high cost.This proposed FP co-processor does not require BSD representation of thesignificands which removes carry-propagation thus increases the speed. Also the PP multiplier for modified FP FFT comprises multipliers and adderswhich removes the requirement of LZD, normalization, and rounding units, thus further improve the speed of the operation. Parallel prefix (PP) multiplier is utilized in the proposed system which provides high speed and efficient performance.

REFERENCES

[1]. E. E. Swartzlander, Jr., and H. H. Saleh, "FFT implementation with fused floating-point operations," *IEEE Trans. Comput.*, vol. 61, no. 2, pp. 284–288, Feb. 2012.

[2]. J. Sohn and E. E. Swartzlander, Jr.,"Improved architectures for a floating-pointfused dot product unit," in *Proc. IEEE 21stSymp. Comput. Arithmetic*, Apr. 2013, pp. 41– 48.

[3]. *IEEE Standard for Floating-Point Arithmetic*, IEEE Standard 754-2008, Aug. 2008, pp. 1–58.

[4]. B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.

[5]. Shashank Mittal, Md. Zafar Ali Khan, M.B. Srinivas, "A Comparative Study of Different FFT Architectures for Software Defined Radio", Proc. of VII SAMOS Workshop, LNCS, Vol. 4599, pp. 375-384, Greece, July 2007.

[6]. Bertil Schmidt and Manfred schimmler, "KPROC – An Instruction Systolic Architecture For Parallel Prefix Applications", Scalable computing: practice and experience, Special issue: Unconventional Parallel Architecture, vol. 3, no. 2, 2000.

[7]. Clyde P. Kruskal, Larry Rudolph and Marc Snir, :The Power of Parallel Prefix", IEEE Transaction on Computer, vol. 34, pp. 965-968, 1985.

[8]. NishaLaguri and K. Anusudha "VLSI implementation of efficient split radix FFT based on distributed arithmetic",in Proceeding of International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE), 2000.

[9]. J. H. Min, S.-W. Kim, and E. E. Swartzlander, Jr., "A floating-point fused FFT butterfly arithmetic unit with merged multiple- constant multipliers," in *Proc. 45th Asilomar Conf. Signals, Syst.Comput.*, Nov. 2011, pp. 520–524

[10]. J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. *Comput.*, vol. 19, no. 90, pp. 297–301, Apr. 1965.

[11]. A. F. Tenca, "Multi-operand floating-point addition," in *Proc. 19th IEEE Symp. Comput. Arithmetic*, Jun. 2009, pp. 161–168.

[12]. Y. Tao, G. Deyuan, F. Xiaoya, and R. Xianglong, "Three-operand floating-point adder," in *Proc. 12th IEEE Int. Conf. Comput. Inf. Technol.*, Oct. 2012, pp. 192–196.

[13]. A. M. Nielsen, D. W. Matula, C. N. Lyu, andG. Even, "An IEEE compliant floating-point adder that conforms with the pipeline packetforwarding paradigm," *IEEE Trans. Comput.*, vol. 49, no. 1, pp. 33–47, Jan. 2000.

[14]. P. Kornerup, "Correcting the normalization shift of redundant binary representations," *IEEE Trans. Comput.*, vol. 58, no. 10, pp. 1435–1439, Oct. 2009.