An efficient VLSI architecture for Iterative Logarithmic Multiplier

Durgesh Nandan[1]
Department of Electronics and Communication Engineering
J.U.E.T
Guna, Madhya Pradesh, India
prof.durgeshnandan@gmail.com

Jitendra Kanungo[2]
Department of Electronics and Communication Engineering
J.U.E.T
Guna, Madhya Pradesh, India
jitendra.kanungo@juet.ac.in

Anurag Mahajan[3]
Department of Electronics and Communication Engineering
J.U.E.T
Guna, Madhya Pradesh, India
anurag.mahajan@juet.ac.in

Abstract— Logarithmic Number System (LNS) based multiplier plays a significant role in the fields of Digital Signal Processing (DSP), Image processing and Neural network which needs a lot of arithmetic operation. In all arithmetic operations, the multiplication is most hardware consuming component. Here, we give a possible solution to that problem by using an efficient VLSI architecture of Mitchell's algorithm based iterative logarithmic multiplier with seamless pipelined technique. The proposed work is based on the hardware minimization at the same error cost than of previously reported architectures. We use VHDL to design the existing and proposed Mitchell's algorithm based iterative logarithmic multiplier. Both multipliers design are evaluated with the Synopsys design compiler by using 90 nm CMOS technology and compared the results in terms of Data Arrival Time (DAT), Area, Power, Area Delay Product (ADP), and EPS (Energy per Sample). The proposed design involves 30.99 %, 31.10 %, and 20.84 % ADP, 5.12 %, 15.48%, and 23.55 % less EPS in comparisons of existing Mitchell's algorithm based iterative logarithmic multiplier for 8 bit,16 bit, and 32 bit operations respectively.

Keywords— Iterative logarithmic multiplier, Logarithmic number system, Mitchell method, Operand decomposition, Seamless pipelined.

I. INTRODUCTION

Logarithmic computation is broadly used in the field of Digital DSP, neural network, and image processing applications. An arithmetic operation (like addition, subtraction, multiplication, and division) is the broadly used operation in Digital Signal Processing (DSP) applications in a real time domain. Logarithm provides an option for digital designers in place of binary arithmetic because logarithm operation has fast and less area consuming in comparisons of binary arithmetic. Due to an efficient hardware implementation of logarithmic operations is a good choice in place of binary arithmetic operations.

In all arithmetic operations, multiplication is the most hardware consuming component due to the generation and processing of huge numbers of partial products. The generation of partial products can be avoided by using a Logarithmic Number System (LNS) [1]-[4]. Logarithmic operations can be converted multiplications operation into additions, and divisions into subtractions, respectively, which can save a lot of computation efforts [5].

The logarithmic multiplication has performed in three steps: (1) conversion of binary numbers into the logarithmic numbers by the help of logarithmic converter, (2) addition operation, and (3) the antilogarithmic conversion of logarithmic numbers by the help of antilogarithmic converter [3]. A Simple block diagram of logarithmic based multiplication is shown below in fig.1. [2].The process of implementation of logarithmic and antilogarithmic converters play a major role in deciding the accuracy and performance of LNS based circuits.

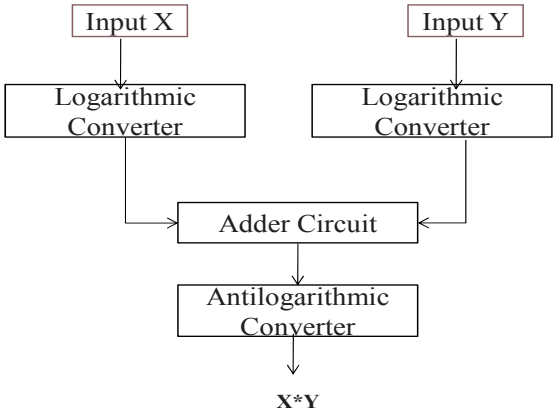

Figure1. Block diagram of logarithmic based multiplication

An example of logarithmic multiplication of two numbers 20 and 60 for easy understanding of operation is given below [8].

Let A=b' (00010100) =d'20; and B=b' (00111100) =d'60;
Log A =0100.0100; and Log B=0101.11100;
Sum = Log A + Log B
Sum = (0100.0100) + (0101.11100) =1010.00100;
Antilog (Sum) = Antilog (Log A + Log B);
Antilog (Sum) =0000100100000000;
X*Y= Antilog (Sum) =1152;

978-1-5090-2797-2/17/$31.00 ©2017 IEEE

Rest of the paper has been organized as the systematic development of LNS is described in Section II. Proposed method of logarithmic multiplication and new approach of seamless pipeline has been explored further in Section III. Section IV gives hardware complexity, synthesis, and simulated results. And finally, explores the applications of LNS and the conclusion are concluded in Section V.

II. REVIEW OF LOGARITHMIC MULTIPLICATION METHODS

Systematic developments of logarithm multiplication methods after evolution are clear understandable for anyone by the help of block diagram given in Fig. 2. LNS based multipliers can be divided into two categories: (1) Look-Up Tables (LUT) and interpolations and (2) Mitchell's algorithm [3-11]. Mitchell's algorithm (MA) can be subdivided into four sub- categories namely as a) divided approximation (DA) b) correction term based c) operand

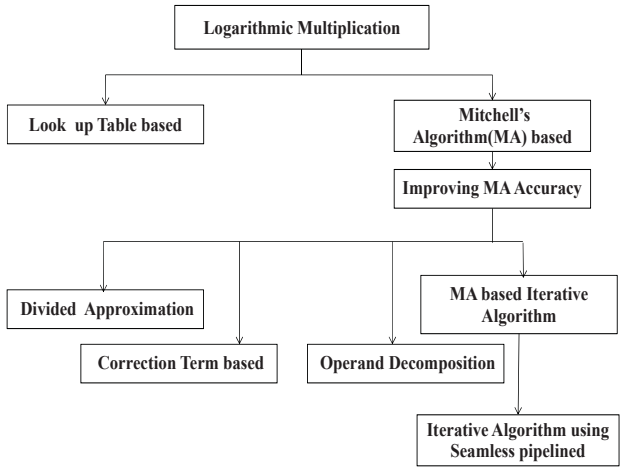

Fig.2 Organization model of Logarithmic Multiplication

decomposition d) MA-based iterative algorithm and e) Iterative algorithm using seamless pipelined. In 1962, Mitchell proposed an algorithm for multiplication using straight line approximation, with an error in the accuracy of approx 12% during multiplication operation. [8]. Interpolation based LNS multiplier gives high accuracy, but at high design cost. In 1970, Hall *et al.* proposed multiple piecewise linear approximation, but accuracy was at the price of hardware complexity, high power consumption, and less speed [9]. In 1999, SanGregory's correcting algorithm was small and fast because it uses only mantissa's four MSB for adjustment of concatenated result [10]. In 2003, Abed and Siferd developed logarithmic/ antilogarithmic converter correction algorithm that fulfills the required trade-off [11-12]. In 2009 and 2011, Juang et.al proposed a two region bit level manipulation scheme to achieve efficient hardware implementation with high level of accuracy [13]-[14]. A similar approach for error minimization has been used for an antilogarithmic converter. In 2015, Juang *et.al* 2 region logarithm approximation ranged over 0 to 0.0319 and ranged over -0.60 to 1.72 for antilogarithm converter [15]. The operand decomposition approach improves the average error percentage and the error range of Mitchell algorithms at the cost of 100 % area expansion. It is equally applicable to all other methods like Divided Approximation based correction and correction term based methods [3]. In 2010, Bulic *et.al* proposed iterative logarithmic approximation. It based on iteration process where the correction terms of multiplication calculate immediate after the product which avoids the comparison of the sum of mantissa with '1. It requires fewer logic resources and increasing the speed of the multiplier with error correction circuits [16]. In 2013, Agrawal *et.al* also proposed similar type of work but gives ASIC implementation of iterative pipelined architecture [17]. We found that Mitchell's algorithm based iterative logarithmic multiplier suffers from poor hardware efficiency, now the challenge is to make it efficient [16-17].

III. PROPOSED METHOD

The existing MA based iterative algorithm for logarithmic multiplication approach does not provide a trade-off between the accuracy, hardware efficiency, and speed. We purpose the hardware efficient VLSI architecture of Mitchell's algorithm based iterative logarithmic multiplier using seamless pipelined technique to get further improvement in trade-off among various parameters.

A. Algorithm for iterative logarithmic multiplier

In this section, we understand the mathematical analysis of iterative algorithm with a possibility of achieving an exact result.

Consider two n-bit numbers N_1 and N_2 of the form.

$$X = x_{n-1}x_{n-2}.....x_2x_1x_0 \quad (1)$$

And

$$Y = y_{n-1}y_{n-2}.....y_2y_1y_0 \quad (2)$$

The final MA approximation for the multiplication depends on the carry bit form the sum of the mantissa and is given by:

$$P_{M.A} = 2^{K_1+K_2}(1+X_1+X_2) \text{ where } X_1+X_2 < 1 \quad (3)$$

$$P_{M.A} = 2^{1+K_1+K_2}(X_1+X_2) \text{ where } X_1+X_2 \geq 1 \quad (4)$$

The binary representation of multiplication of two numbers N_1 and N_2 is express as below:

$$P_{true} = N_1 * N_2 = 2^{k_1+k_2}(1+x_1+x_2)+2^{k_1+k_2}(x_1x_2) \quad (5)$$

For avoid approximation error, we take.

$$x * 2^k = N - 2^k \quad (6)$$

Putting the value of equation (19) in equation (18) and we find.

$$P_{true} = N_1 * N_2 = 2^{(k_1+k_2)} + (N_1 - 2^{k_1})2^{k_2} + (N_2 - 2^{k_2})2^{k_1} + (N_1 - 2^{k_1}) * (N_2 - 2^{k_2}) \quad (7)$$

Suppose that

$$P^{(0)}_{approx} = 2^{(k_1+k_2)} + (N_1 - 2^{k_1})2^{k_2} + (N_2 - 2^{k_2})2^{k_1} \quad (8)$$

Is the first approximation of the product and P_{true} is the

$$P_{true} = P^{(0)}_{approx} + (N_1 - 2^{k_1}) * (N_2 - 2^{k_2}) \quad (9)$$

The absolute error after the first approximation is

$$E^{(0)} = P_{true} - P^{(0)}_{approx} = (N_1 - 2^{k_1}) * (N_2 - 2^{k_2}) \quad (10)$$

If we repeated multiplication procedure till n correction terms, we can approximate the product as:

$$P^{(n)}_{approx} = P^{(0)}_{approx} + \sum_{j=1}^{n} C^{(j)} \quad (11)$$

Equations (1) to (11) show the complete mathematical analysis of iterative algorithm [16].

B. Seamless pipelined technique

Pipelining performs by inserting registers in combinational sections of the data path for accelerate digital signal processing (DSP) applications. But, it's hard to decide on the proper locations where pipeline registers should be inserted to derive the desired advantage. We see that at the architecture-level pipelining by injecting registers in between adjacent arithmetic circuits does not achieve a meaningful reduction in the critical path so that the advantage of pipelining is usually missed in DSP circuits. We need to pipeline at the architecture level, and pipeline within the arithmetic circuits are considered separately [18]. To derive greater advantage of pipelining, combinational logic is required to be partitioned seamlessly without restricting the locations of pipeline registers to only in between arithmetic circuits. We propose a seamless pipelining approach where the delay registers are placed across the signal path of a unified network of combinational components without restricting them to be placed in between two arithmetic components. It is different from conventional fine-grained pipelining which requires additional registers to be placed within the arithmetic circuits. The proposed pipelining approach takes a seamless view of combinational sections and decides on the appropriate locations where delay registers should be placed to reduce the critical path most effectively. For effective seamless pipelining, we obtain a precise estimate of the propagation delay of composite circuits to find the locations of pipeline registers to minimize the critical path. A conceptual diagram of seamless pipelining of 8 bit ripple carry adder circuit is shown in Fig. 3.

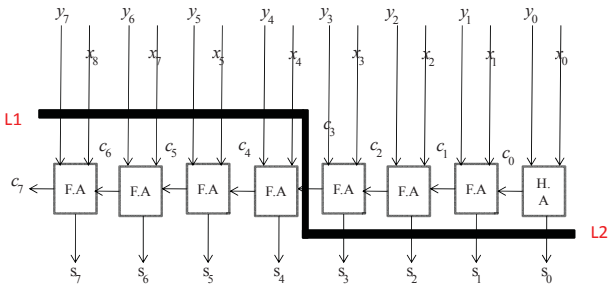

Fig.3. Block diagram of seamless pipelining of 8 bit ripple carry adder circuit

C. Functional Architecture of MA based iterative logarithmic multiplier using seamless pipelined architecture

The generalize diagram of the proposed iterative logarithmic multiplier using seamless pipelined is presented in Fig. 4. The MA-based iterative logarithmic multiplier using seamless pipelined architecture consisting of six major blocks: leading one detector (LOD), priority encoder, barrel shifter (left), adder circuit with seamless pipelined register, decoder, and register. Here in fig.4, PIPO stands for parallel in parallel out register which is here used for a general pipelining purpose, RCA with S.P stands for a ripple carry adder with seamless pipelined, and BSL stands for barrel shift register. Functionality of various blocks is discussed as: the LOD is simple to design

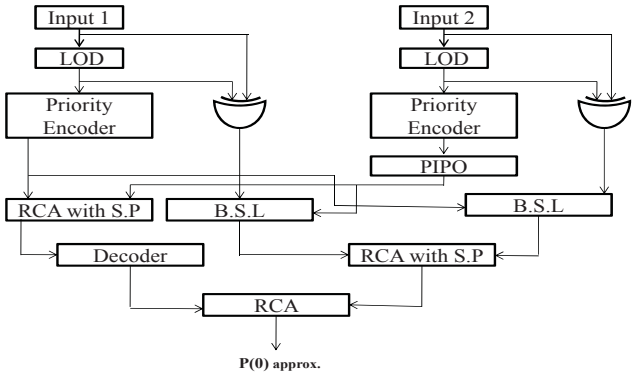

Fig.4. Generalize diagram of proposed seamless pipelined iterative logarithmic multiplier

and give a result which counts the position of LSB, priority encoder expands the LOD as a number, barrel shifter shift number on the left side. Seamless pipelined is a newly proposed pipeline technique; it reduced the delay of architecture without any increment of the area [18]. And finally, we remove unnecessary pipelined architecture. As in the seamless pipeline implementation of the basic block, the residues are available after the first stage. The correction circuit can now start to work immediately after the first stage of the prior block is finished [18].

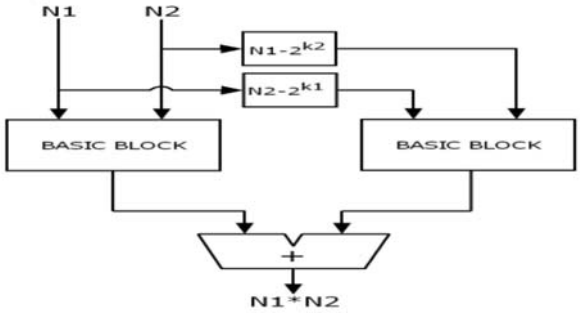

Fig.5. Block diagram of the proposed seamless pipelined multiplier with one error correction circuits

The seamless pipelined multiplier with one correction circuits is presented in figure 5.

IV. RESULTS AND DISCUSSION

The proposed structure has N input samples and produces a 2N output sample Where N is an equal number of bits used as the input bits. The proposed structure for logarithmic multiplication involves a LOD, XOR gate, priority encoder, barrel shift (left), RCA circuit with seamless pipelined, decoder circuit, and PIPO register.

A. Hardware complexity and Synthesis results

Our motive of this work is to implement the seamless pipelined technique to construct an MA based iterative logarithmic multiplier architecture which is efficient regarding hardware. We concentrate on minimizing the hardware with an equal accuracy. To fulfill this target, we apply seamless pipelined register in place of simple pipelining and remove additional pipeline whichever it is not required.

The theoretical analysis of area complexities in terms of transistors counted with proposed structures and reported structures of 8 bits are listed in Table 1. The proposed iterative logarithmic multiplier structure involves 74 less NOT gates, 74 less OR gate, 148 less AND gates and same XOR/XNOR gates in comparison of Existing.

We have synthesized the proposed iterative logarithmic multiplier using seamless pipelined architecture 90 nm CMOS technology node and performance evaluated for 8, 16, and 32 bits data inputs. We have also implemented and synthesizing the designs of the 8, 16, and 32 bits an ASIC-based logarithmic multiplier using iterative pipelined architecture [17]. The area, power, and timing constraints are compared with the reported ASIC-based logarithmic multiplier using iterative pipelined architecture [17] with Mitchell's algorithm based iterative logarithmic multiplier with seamless pipelined technique and without of the error correction circuit as listed in Tables 2. The ADP and EPS for the proposed and reported structures [17] are also listed in Tables 2. As shown in Table 1, the proposed Mitchell's algorithm based iterative logarithmic multiplier using seamless pipelined technique without the error correction circuit involve 8.21 %, 18.74 % and 12.68 % less DAT, 24.82 %, 15.21 %, and 9.34 % area, 30.99 %, 31.10 %, and 20.84 % ADP, 5.12 %, 15.48%, and 23.55 % less EPS for 8 bits, 16 bits and 32 bits architecture respectively as compared to reported an ASIC based logarithmic multiplier using iterative pipelined architecture [17].

Table1. General comparison of hardware complexities

Structure	Existing[17]	Proposed
NOT	205	131
OR	234	160
AND	493	345
EXOR/EXNOR	81	81

V. CONCLUSION

The seamless pipeline can reduce critical path better than the pipeline techniques. The pipeline can only reduce DAT up to 10 %, but Seamless pipeline reduced up to 20 % at same hardware cost. Our main motive to design the hardware architecture of logarithmic multiplier is to make it efficient for DSP applications where accuracy is not a big deal. Less hardware resource required for LNS multiplier while the binary multiplier takes a lot of hardware resources. This multiplier is an efficient and capable to use as an independent multiplier.

Table2. Synthesis results of proposed Logarithmic Multiplication using seamless pipelined architecture and reported structures [16].

Structure	Existing[17]			Proposed			% Gain		
	N=8	N=16	N=32	N=8	N=16	N=32	N=8	N=16	N=32
DAT(ns)	37.04	73.80	127.80	34.00	59.97	111.60	8.21 %	18.74 %	12.68 %
Area(μm²)	9557.47	22381.69	63586.93	7186.07	18977.67	57648.15	24.82 %	15.21 %	9.34 %
Power(μW)	103.63	263.36	754.80	142.48	223.91	660.89	-37.48%	14.98%	12.45%
ADP(μm²*μs)	354.008	1651.76	8126.40	244.32	1138.09	6433.53	30.99 %	31.10 %	20.84 %
EPS(n J)	3826.23	19435.96	96463.44	3630.96	16429.13	73755.32	5.12 %	15.48%	23.55 %

References

[1]. H. Fu, O. Mencer, and W. Luk, "FPGA Designs with Optimized Logarithmic Arithmetic." IEEE Transactions on Computers, Vol. 59, No. 7, pp. 1000–1006, July 2010.

[2]. K. Johansson, O. Gustafsson and L. Wanhammar, "Implementation of Elementary Functions for Logarithmic Number Systems," IET Computer & Digital Techniques, Vol. 2, No. 4, pp. 295–304, April 2008.

[3]. V. Mahalingam, and N. Rangantathan, "Improving Accuracy in Mitchell's Logarithmic Multiplication Using Operand Decomposition," IEEE Transactions on Computers, Vol. 55, No. 2, pp. 1523-1535, December 2006.

[4]. P. Saha, A. Banerjee, A. Dandapat and P. Bhattacharyya, "High speed multiplier using high accuracy floating point logarithmic number system," Scientia Iranica Transactions D: Computer Science & Engineering and Electrical Engineering, Vol. 21, No. 3, pp. 826–841, 2014.

[5]. L. K. Yu, and D. M. Lewis., "A 30-b Integrated Logarithmic Number System Processor." IEEE Journal of Solid State Circuits, Vol. 26, No. 10, pp. 1433–1440,Oct. 1991.

[6]. E. Swartzlander and A. Alexopoulos, "The sign/logarithm number system," IEEE Transactions on Computers, vol. C, no 12, pp. 1238–1242, December 1975.

[7]. M. J. Schulte, and E. E. Swartzlander, "Hardware Designs for Exactly Rounded Elementary Functions." IEEE Transactions on Computers, Vol. 43, No. 8, pp. 964–973, Aug. 1994.

[8]. J.N. Mitchell, "Computer Multiplication and Division using Binary Logarithms," IRE Transactions on Electronic Computers, Vol. 11, No. 6, pp. 512-517, Aug. 1962.

[9]. E.L. Hall, D.D. Lynch, and S.J. Dwyer III, "Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications," IEEE Trans. Computers, vol. 19, no. 2, pp. 97-105, Feb. 1970.

[10]. S.L. San Gregory, R.E. Siferd, C. Brother and D. Gallagher, "Low-Power Logarithm Approximation with CMOS VLSI Implementation," Proc. IEEE Midwest Symp. Circuits and Systems, Aug. 1999.

[11]. K.H. Abed, R.E. Sifred, "CMOS VLSI Implementation of a Low- Power Logarithmic Converter," IEEE Transactions on Computers, Vol. 52, No. 11, pp. 1421-1433, November 2003.

[12]. K.H. Abed, R.E. Sifred, "VLSI Implementation of a Low-Power Antilogarithmic Converter," IEEE Transactions on Computers, Vol. 52, No. 9, pp. 1221-1228, September 2003.

[13]. T.B. Juang, S.H. Chen, and H.J.Cheng, "A Lower error and ROM-free Logarithmic Converter for Digital Signal Processing Applications." IEEE Transactions on Circuits and Systems II Vol. 56 (12):pp. 931–935, 2009.

[14]. T.-B. Juang, P. K. Meher, and K.S. Jan, "High-performance Logarithmic Converters Using Novel Two-region Bit-level Manipulation Schemes." IEEE International Symposium on VLSI Design, Automation and Test, pp.1–4, 25–28 April 2011.

[15]. Tso-Bing Juang, Han-Lung Kuo and Kai-Shiang Jan, "Lower-error and area-efficient antilogarithmic converters with bit-correction schemes," Journal of the Chinese Institute of Engineers, 2015.

[16]. P. Bulic, Z. Babic, and A. Avramovic, "A simple pipelined logarithmic multiplier", IEEE International Conference on Computer Design (ICCD), pp.235-240 (2010).

[17]. R. K Agrawal, and H. M. Kittur, "ASIC based logarithmic multiplier using iterative pipelined architecture," IEEE Conference on Information & Communication Technologies (ICT),pp.362-366 (2013).

[18]. P. K. Mehar, "seamless pipelining of DSP circuits," circuits system and signal processing, (4 July 2015), DOI 10.1007/s00034-015-0089-2.