Optimized Design and Implementation of an Iterative Logarithmic Signed Multiplier

Sanjeev kumar Patel, Vinod Kapse

Abstract- We have present a new architecture for multiplication of signed numbers. The objective of this paper work was to design a 16 bit signed logarithmic multiplier. The coding is done for 16 bit multiplications using verilog.For the design entry, we used the Xilinx ISE 13.2 - Web-PACK and the design was synthesized with the Xilinx XST Release 13.2 for Windows. When using Xilinx xc3s1500-5fg676 device

Keywords- S-BOX, SOC, LNS, MA

1 INTRODUCTION

Digital arithmetic operations are very important in the design of digital processors and application-specific systems. Arithmetic circuits form an important class of circuits in digital systems. Multiplication is especially relevant since other arithmetic operators, such as division or exponentiation, which they usually utilize multipliers as building blocks. Hardware implementation of arithmetic operations has been oriented typically to use VLSI circuits. Among the arithmetic operations, the multiplication is widely used in applications such as graphics and scientific computation. The use of alternative number systems to optimize the realization of arithmetic blocks, maintaining high performance without incurring prohibitive area and power increases. One such number system is the Logarithmic Number System in base two. Utilizing this system has the potential to result in highly optimized realizations of functions such as multiplication, division and square root. We have two base papers [1] & [2] for our work comparison. Ref [1] has proposed architecture for unsigned logarithmic multiplier; This paper presents comparison of required number of slices,4 input lut's and power consumption at 25 MHz of logarithmic number system multipliers. The target device of implemented multipliers is Xilinx Spartan 3 xc3s1500-5fg676 FPGA chip. We also propose iteration method to achieve accuracy.

2 COMPLEX MULTIPLICATIONS

2.1 The Digital Down Converter (DDC)-It can be viewed as a complex mixer, is used to shift a signal from one frequency range to another. A complex mixer includes a Direct Digital Frequency Synthesizer (DDFS) and a complex multiplier, where the complex multiplier is a major building block of the complex mixer. The complex mixer consists of one DDFS, four multipliers, one adder, and one subtractor.

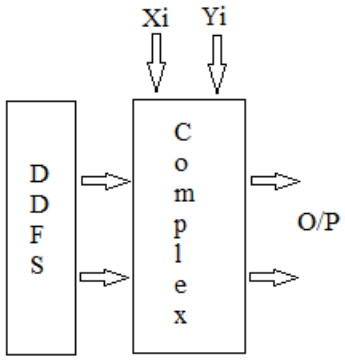

Fig 1: shows the block diagram of the complex mixer.

2.2 Logarithmic Multiplier Architecture The logarithmic multiplier is the main component of the complex multiplier. It consists of two logarithmic converters, one adder and one antilogarithmic converter, as shown in Figure 2. The complex multiplier needs four multiplications, one addition, and one subtraction in order to generate the real and imaginary outputs. An alternative approach suggests the use of three real multipliers instead of four, and the details are shown in Equations (A) and (B). Effectively, the trade off involves two extra additions and an increased multiplier word length. This approach is rejected for our proposed LNS based complex multiplier because of the increased complexity of performing logarithmic addition.

$$X_o = AC - BD \quad \text{(Real)} \quad \text{(A)}$$

$$Y_o = (A + B)(C + D) - AC - BD \quad \text{(Imaginary)} \quad \text{(B)}$$

From Equation the complex multiplier real component can be expressed as:

$$X_o = AC - BD = \log^{-1}(\log A + \log C) - \log^{-1}(\log B + \log D) \quad (C)$$

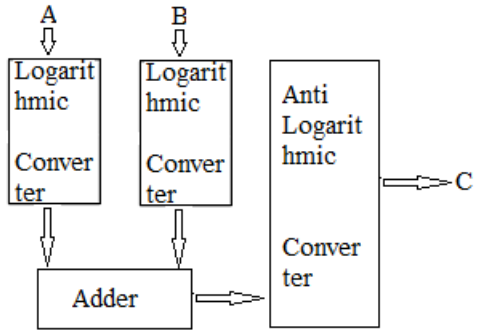

Fig 2: Logarithmic multiplier system diagram

To implement Equation (C) in hardware requires four logarithmic converters to convert the inputs, two adders to perform the logarithmic multiplications, two antilogarithmic converters to convert the products from LNS to binary formats, and a subtractor to subtract BD from AC in binary format. Similarly, the imaginary component described in Equation (B) can be expressed as:

$$Y_o = BC + AD = \log^{-1}(\log B + \log C) + \log^{-1}(\log A + \log D)$$

2.2.1 Mitchell's Algorithm- One of the most significant multiplication methods in LNS is Mitchell's algorithm. An approximation of the logarithm and the antilogarithm is essential, and it is derived from a binary representation of the numbers.
The logarithm of the product is

$$\log_2(N_1.N_2) = k_1 + k_2 + \log_2(1 + m_1) + \log_2(1 + m_2)$$

The expression $\log_2(1 + m)$ is approximated with m and the logarithm of the two number's product is expressed as the sum of their characteristic numbers and mantissas:

$$\log_2(N_1.N_2) \approx k_1 + k_2 + m_1 + m_2 \quad (D)$$

The characteristic numbers k_1 and k_2 represent the places of the most significant operands' bits with the value of '1'. For 16-bit numbers, the range for characteristic numbers is from 0 to 15. The fractions m_1 and $m_2$are in range [0, 1].
The final MA approximation for the multiplication where $P_{true} = N_1.N_2$ depends on the carry bit from the sum of the mantissas and is given by:

$$P_{MA} = (N_1.N_2)_{MA} = \begin{cases} 2^{k_1+k_2}(1 + m_1 + m_2), & m_1 + m_2 < 1 \\ 2^{k_1+k_2+1}(m_1 + m_2), & m_1 + m_2 \geq 1 \end{cases}$$

The final approximation for the product above requires the comparison of the sum of the mantissas with '1'.The sum of the characteristic numbers determines the most significant bit of the product. The sum of the mantissas is then scaled (shifted left) by $2^{k_1+k_2}$ or by $2^{k_1+k_2+1}$, depending on the $m_1 + m_2$.
If $m_1 + m_2 < 1$, the sum of mantissas is added to the most significant bit of product to complete the final result. Otherwise, the product is approximated only with the scaled sum of mantissas. The proposed MA-based multiplication is given in Algorithm 1

Algorithm 1-

1.N_1, N_2: n-bits binary multiplicands, P_{MA}= 0:2 n-bits approximate product
2. Calculate k_1: leading one position of N_1
3. Calculate k_2: leading one position of N_2
4. Calculate m_1: shift N_1 to the left by $n - k_1$bits
5. Calculate m_2 : shift $N_2$to the left by $n - k_2$bits
6. Calculate $k_{12} = k_1 + k_2$
7. Calculate $m_{12} = m_1 + m_2$
8. IF $m_{12} > 2^n (i.e.\ m_1 + m_2 \geq 1)$
(a) Calculate $k_{12} = k_{12} + 1$
(b) Decode k_{12} and insert m_{12} in that position of P_{approx}
ELSE:
(a) Decode k_{12} and insert '1' in that position of P_{approx}
(b) Append m_{12} immediately after this one in P_{approx}
9. Approximate $N_1.N_2 = P_{MA}$

2.2.2 Error Estimation- Based on Equation (3.29), the error of the logarithmic-based multiplier can be determined as:

$$E_m = \frac{P_{approx} - P_{true}}{P_{true}} = \frac{P_{approx}}{P_{true}} - 1$$

$$E_{m1} = \frac{1 + m_1 + m_2}{(1 + m_1)(1 + m_2)} \quad for\ m_1 + m_2 < 1$$

$$E_{m2} = \frac{2(m_1+m_2)}{(1+m_1)(1+m_2)} \quad for\ m_1 + m_2 \geq 1$$

2.2.3 Error Correction-Mitchell analyzed this error and proposed the following analytical expression for the error correction.

$$(N_1.N_2)_{MAC} =$$
$$\begin{cases} P_{MA} + 2^{k_1+k_2}(m_1.m_2) \ , & m_1 + m_2 < 1 \\ P_{MA} + 2^{k_1+k_2}(1-m_1)(1-m_2), & m_1 + m_2 \geq 1 \end{cases}$$

Where $2^{k_1+k_2}(m_1.m_2)$ $2^{k_1+k_2}(1-m_1)(1-m_2)$ are the correction terms proposed by Mitchell.

3 DESIGN & IMPLEMENTATION

Here a solution is given to simplify the logarithmic approximation introduced by Mitchell and introduces an iterative algorithm with various possibilities for achieving the multiplication error as small as required and the possibility of achieving the exact result. By simplifying the logarithm approximation introduced in (D), the correction terms could be calculated almost immediately after the calculation of the approximate product has been started. In such a way, the high level of parallelism can be achieved by the principle of pipelining, thus reducing the complexity of the logic required by (D) and increasing the speed of the multiplier with error correction circuits.

The proposed design works for 2's compliment number representation for 16 bit signed number. In the proposed design LOD is replaced by S-BOX the proposed design works for signed number by adding Ex-OR gate and adder in both inputs of reference design. So the input of two 16 bit signed numbers is converted into unsigned numbers. In the proposed design priority encoder, decoder and barrel shifter are used and same work as in reference design. The product of these two numbers is converted into signed number by using Ex-or gate and adder. The sign of the product is depend on the most significant bit of both input operands. A basic block (BB) of signed multiplier shown in figure 3 is a simple multiplier with no correction terms. The task of the basic block is to calculate one approximate product without pipelining; it will have maximum error and slow speed.

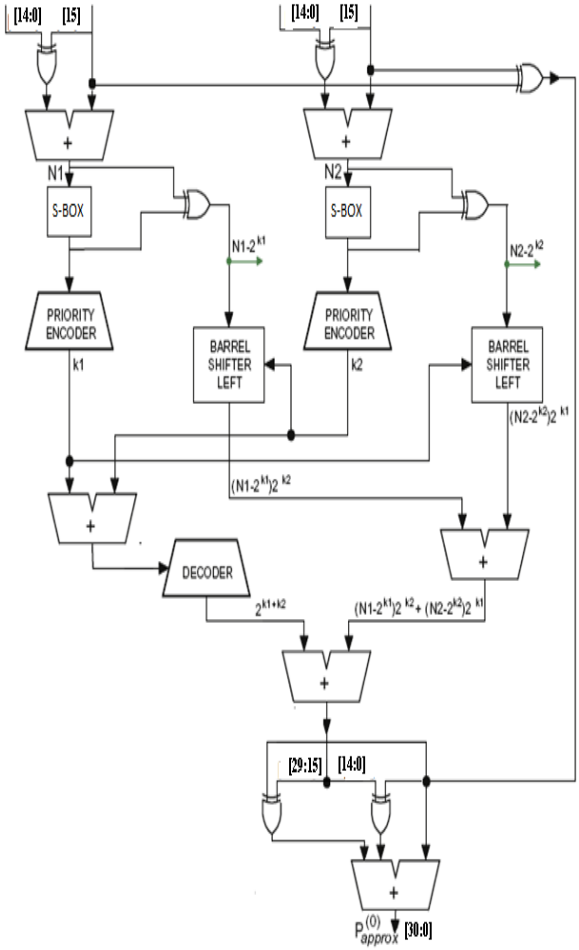

Fig 3: Basic Block of signed logarithmic multiplier

To decrease the maximum combinational delay in the basic block, we used pipelining to implement the basic block shown in figure 4. For pipelining registers are used in each segment the implemented design has 4 stage pipelining.

As in the pipelined 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 from the prior block is finished. The pipelined multiplier with two correction circuits is presented in Figure 5.

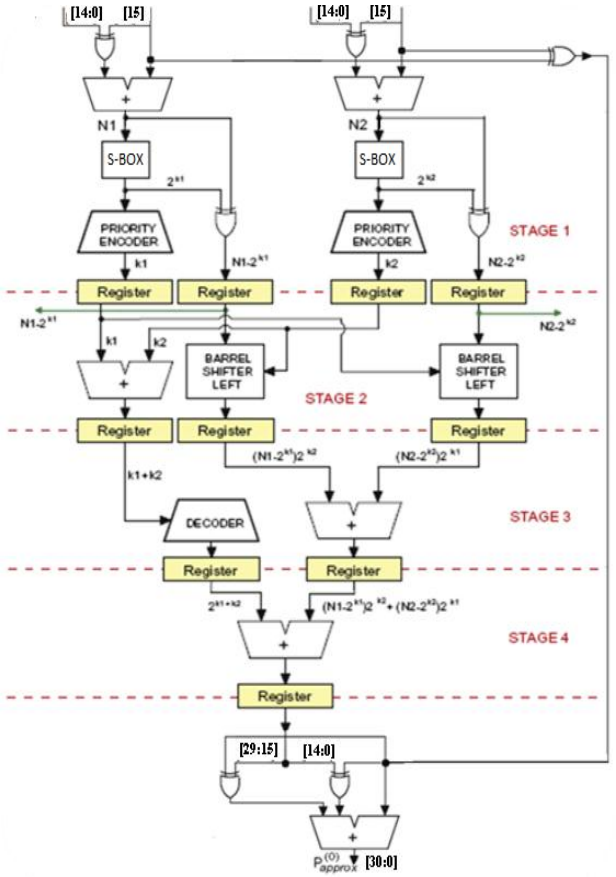

Fig 4: Pipelined Basic Block Signed Multiplier

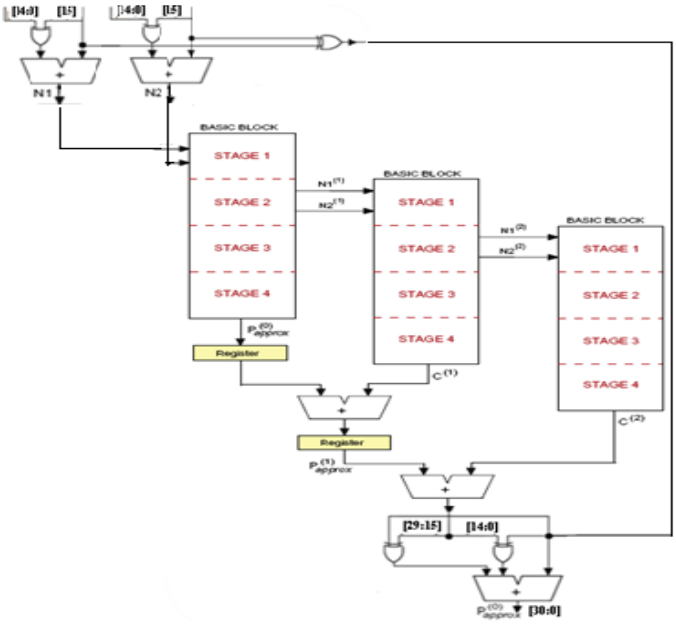

Fig 5: Two stage Iteration in pipelined signed multiplier

4 RESULTS

This chapter gives the output of each design method of the signed multiplier design. The comparison of the reference design's and the implemented design is also presented in this chapter. The parameters of comparison includes the number of slices, number of 4-input LUTs and power consumption by the module at 25MHz clock frequency.

The range of the implemented design in decimal is 65534 from -32768 to 32767.

We have use Xilinx ISE 13.2 web pack for synthesis our result for the all different designs. Table 1,table 2,table 3 shows the synthesis results of different design approaches used for signed multiplier. Tables show the comparative results with reference [1] & [2].

4.1 COMPARISON OF 16 BIT PIPELINED MULTIPLIER

MULTIPLIER		PBB	PBB+1ECC	PBB+2ECC	PBB+3ECC
NO. OF SLICES	Ref	216	427	635	824
	Our	214	414	608	790
4 INPUT LUT'S	Ref	404	803	1189	1546
	Our	392	757	1126	1462
PERCENTAGE REDUCTION (%)	Slices	0.92	3.04	4.25	4.12
	Lut's	2.97	5.72	5.29	5.43

4.2 AVERAGE RLATIVE ERROR FOR 16 BIT PIPELINED MULTIPLIER [%]

MULTIPLIER	PBB	PBB+1ECC	PBB+2ECC	PBB+3ECC
Ref	9.412	0.9874	0.1070	0.0117
Our	6.57	0.495	0.05696	0.00295

4.3 ESTIMATED POWER CONSUMPTION AT 25 MHz FOR 16 BIT PIPELINED MULTIPLIER

MULTIPLIER		PBB	PBB+1ECC	PBB+2ECC	PBB+3ECC
LOGIC & SIGNALS (MW)	Ref	3.72	7.32	10.66	13.37
	our	2.21	3.58	5.44	6.37
IO BLOCKS (MW)	ref	51.26	51.62	51.67	51.93
	our	20.10	24.53	50.41	49.01
QUIESCENT (MW)	ref	152.06	152.66	153.03	153.42
	our	150	150	151	151
TOTAL POWER (MW)	ref	207.04	211.6	215.36	218.72
	our	174	180	213	215

4.4 SIMULATION WAVEFORM

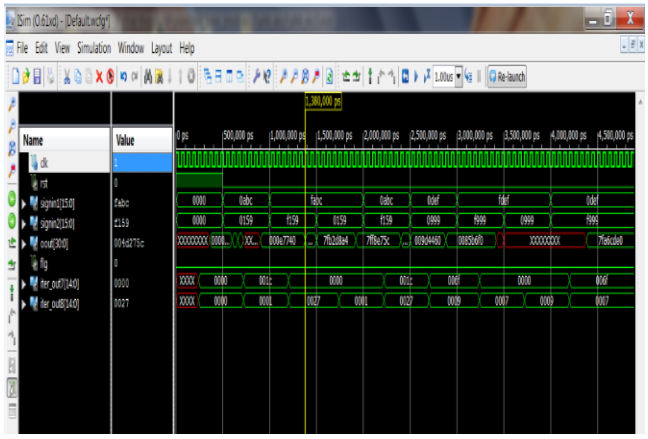

Fig 6:simulation waveform of pipelined basic block with three error correction circuit.

CONCLUSION

The implemented design reduced no of slices and 4 input lut's.The power consumption and average relative error also reduced.The correction circuit added combination delay increases with each added correction circuit, by pipelining this was significantly improved .

REFERENCES

[1] A Simple Pipelined Logarithmic Multiplier,Patricio Buli´c*, Zdenka Babi´c † and Aleksej Avramovi´c , Computer Design (ICCD), 2010 IEEE International Conference.

[2] An iterative logarithmic multiplier, Z. Babic´ a, A. Avramovic´ a, P. Bulic, Microprocessors and Microsystems 35 (2011) 23–33, Elsevier

[3] Saokar,S.S.Banakar,R.M.;siddamal,S,High speed signed multiplier for digital signal processing applications. 15-17 March 2012

[4] Swartzlander,E.E.,Jr.Alexopoulos,A.G, The signed/logarithm number system. Dec. 1975

[5] J.N. Mitchell, Computer multiplication and division using binary logarithms, IRE Transactions on Electronic Computers EC-11 (1962) 512–517.

[6] K.H. Abed, R.E. Sifred, CMOS VLSI implementation of a low-power logarithmic converter, IEEE Transactions on Computers 52 (11) (2003) 1421–1433.

[7] K.H. Abed, R.E. Sifred, VLSI implementation of a low-power leading one detector, IEEE Transactions on Computers (2003)

About the authors

Sanjeev Kumar Patel is currently a Research Scholar in M. Tech in Embedded System and VLSI Design in Gyan Ganga Institute of Technology and Sciences, Jabalpur. He did his graduation in Electronics and Communication in 2006 and his area of interest lies in field of VLSI Design.

Vinod Kapse born at Nagpur in India. He received the B.E. degree in Industrial Electronics from Amaravati University, Amaravati , India, in 1998, M. Tech. degree in Electronics Engg. From Nagpur University, Nagpur , India in 2007. In 1999, he joined the Srijan Control Drives in R & D department. In 2000, he joined the Sibar Software Services (India) Ltd. as a Design Engineer. In 2002, he joined as a Lecturer in Department of Electronics & Communication in Guru Ramdas Khalsa Institute of Science & Technology, Jabalpur (M.P.) India. He has been member of IEEE. He is currently a Asst. Professor in Gyan Ganga Institute of Technology & Science, Jabalpur (M.P.) India. His research interest includes VLSI Design, Fuzzy logic, Robotics.Email:kapse.vinod@rediffmail.com.