International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

Design of LNS based Approximate Multiplier using Mitchell's Algorithm and Modified Operand Decomposition Technique

Shanmugapriya. A
M,E.VLSI Design,
Department of ECE,
Jeppiaar Engineering College,
Chennai,India.

G. Balachandran
Assistant Professor,
Department of ECE,
Jeppiaar Engineering College,
Chennai,India.

***Abstract*-In signal processing, multiplication is an important operation but slow and complex leading to more time consumption. LNS provides a suitable alternative for implementing multiplication operation. The LNS approximate logarithmic multiplier converts multiplications to additions by taking approximate logarithm and achieves significant improvement in reducing complexity and delay .Mitchell's algorithm(MA) offers a simplest way of determining the logarithm and antilogarithm however it suffers from certain loss of accuracy. The experimental results indicate that the value obtained by MA algorithm has an error percentage of 6.66%. Our proposed decomposition design shows a significant improvement in terms of accuracy over the previous work that has been applied to logarithmic multiplication.**

***Keywords- Signal processing, LNS, MA , operand decomposition*.**

I. INTRODUCTION

Multiplication is a significant process in DSP applications,these algorithms involve repetitive multiplications which require more time. In DSP applications, time is a crucial factor than accuracy. The quality of images which is crucial in digital image processing applications like medical imaging, satellite imaging ,biometric trait images etc., can be improved by multipliers. Lot of research is going on to optimize the multipliers in terms of speed, area and power or a combination of these parameters. The traditional multipliers use large amount of hardware and are power hungry. One of the alternate solutions is the implementation of LNS in multipliers.

In all arithmetic operations the most hardware consuming components is multiplication due to partial products generation .This partial product generation can be avoided by LNS based Multiplication [9]. In LNS, repeated addition is the basis for multiplication. So adders are the building blocks for LNS. Multiplication is realized by computing logarithms of numbers represented in binary notation and then adding those values. Antilog value of the obtained sum is computed. The logarithmic multiplication is performed in three steps: (1) conversion of binary numbers into the logarithmic numbers, (2) addition operation, and (3) the antilogarithmic conversion of logarithmic numbers [14].

LNS multipliers are mainly classified into two types. They are look up table (LUT) based approach and Mitchell Algorithm (MA) based approach. MA approach is more popular than LUT due to less complex hardware which conserves area. Piecewise linear approximation of the log curve introduces significant percentage of error into the system. This error percentage increases relatively with the increase of number of '1' bits in mantissa

Mitchells algorithm for multiplication of two operands is simple. The logarithm of the input values are added then the antilogarithm is taken for the sum which is the final product. The accuracy degrades in this method due to the technique used to determine the log and antilog . The error in logarithm of Mitchells algorithm is in the fractional part. When fraction part of log is zero MA log value is same as actual log value . The error due to MA lies in the range of 0 to 0.086 and the error is maximum when fractional part is 0.44

Several methods have been proposed to reduce error in Mitchells algorithm.In this paper , a new design based on modified operand decomposition technique is proposed to improve accuracy . The operand decomposition technique reduces the number of 1 bits in the decomposed operands which reduces the carry bits from mantissa to the integer during addition of log values.

II. EXISTING SYSTEM

A. *LOGARTHMIC MULTIPLICATION BASED ON MITCHELLS ALGORITHM :*

Mitchell proposed a very low cost , simple method to determine logarithmand antilogarithm using piecewise straight line approximation of the log curve. The steps involved in Mitchell algorithm are i) calculation of log values by shifting and counting operations, ii) adding of the two log values, iii)determining the antilog of the two summed values. A zero detector is placed initially to check if any input value is zero to avoid the further computations. The drawback of MA is that the maximum worst case error is around 11.12% and average error is around 3.79%[2]. However there are certain advantages of MA which are reduction in power and area. It also enhances speed when applied in high speed applications.

Approximation of the logarithm and anti -logarithm is important and essential in determining the final output using MA. It can be determined from final representation of numbers[3].

$N=2^{p}(1+\sum_{j=n}^{p-1} 2^{j-p}.Y_j)$

$=2^{p}(1+x)$

Where p=place of the most significant bit with the value of 1,Y_j is the bit value at the j-th position, x is mantissa and n depends on numbers precision.

By logarithm of the product computation,

$log_2(N_1.N_2)=p1+p2+log_2(1+x1)+log_2(1+x2)$

$log_2(1+x1)$ is approximated with x1, and log of the 2 numbers product is expressed as a sum of characteristic numbers and mantissas.

$log_2(N_1.N_2)\approx p1+p2+x1+x2.$

Antilog also uses the similar approximation, The final MA approximation for multiplication depending on one carry bit from sum of mantissas is given by:

$$(N1.N2)_{MA}=\begin{cases} 2^{p1+p2}(1+x1+x2), x1+x2<1 \\ 2^{p1+p2+1}(x1+x2), x1+x2\geq 1 \end{cases}$$

MSB of product is determined by the sum of characteristic &sum of mantissa completes the final result.

B. MITCHELL ALGORITHM:

```
A,B:n-bits,P:2n -bits approximate product
   1. $K_A$ =LOD(A),$K_B$=LOD(B)
   2. $X_A$=A<<(n- $K_A$ -1),$X_B$=B<<(n- $K_B$-1)
   3. L =(‘0’&$K_A$& $X_A$[n-2...0])+(‘0’&$K_B$&$X_B$[n-2…..0])
   4. Charac = L[n+log(n)-1…….n-1], mant = L[n-2…0] , S = Charac [log(n)]
   5. If S==‘1’ THEN// Large characteristic
        D=(‘1’&mant)<<
        ((‘0’&Charac[log(n)- 1…..0])+1)
      ELSE // Small characteristic
        D = (‘1’&mant)>>(~char[log(n)-1……0])
  6. If Is-Zero(A,B) THEN// A or B are zero
        P = 0
     ELSE
        P=D
```

An example for multiplication of two numbers using Mitchell algorithm is given below:

Let us consider two values ten (1010) and six(0110) n=4 by following the above explained Mitchell algorithm multiplication is carried out.

A=1010 ; B=0110 ;n=4

1. K_A=0011;K_B=0010
2. X_A=1010<<(n-K_A-1)
   =1010<<(4-3-1)
   =1010
   X_B=0110<<(n-K_B-1)
   =0110<<(4-2-1)
   =1100

3.L = (0&K_A&X_A[n-2….0])+(0&K_B&X_B[n-2….0])
= (0&11&1010[2:0])+(0&10&1100[2:0])
=(011010)+(010100)
=101110

4.Charac = L[n+log(n)-1…n-1]
=101110[4+(2-1)….3]
=101110[2:0]
=101
mant= L[n-2…..0]
=101110[2:0]
=110

S = Charac[log(n)]
=101[2]
= 1

5. S==1;
D =(1&mant)<<((‘0’&Charac[log(n)-1…0])+1)
=(1110)<<(0101[1:0]+1)
=1110<<(01+1)
=111000 (56)

The example of Mitchells algorithm given above indicates the result is approximate , in order to improve the accuracy modified operand decomposition can be used.

III. PROPOSED METHOD

The existing does not provide anacceptable a trade-off between the accuracy, speed, and complexity. The Improved Operand Decomposition (IOD) is proposed to quantify the trade-off.

A.MODIFIED OPERAND DECOMPOSITION TECHNIQUE:

Consider two n-bit numbers *X* and *Y* of the form:

$X=x_{n-1}x_{n-2}.............x_2\,x_1\,x_0$ and

$Y=y_{n-1}y_{n-2}.............y_2y_1y_0.$

The operands X and Y are compared and the greater number is considered as X1and the other operand is considered as X2.

The operands X1 and X2 are decomposed into A and B by the formula given by

A=X1&X2

B=~X1&X2

The product is then computed from the decomposed operands by using the following equation:

X*Y={(X1*A)+(X1*B)}

By using the modified OD the accuracy of Mitchell’s algorithm can be improved with an improvement in the area, delay, Area Delay Product (ADP) and power consumption.

B.ALGORITHM FOR LOGARTHMIC MULTIPLICATION USING MODIFIED OPERAND DECOMPOSITION TECHNIQUE:

Let X and Y be the inputs of n-bits

Step1: Compare X and Y and assign the greater number as X1and smaller number as Y1

Step2: Calculate A and B

Where A=X1&Y1; B=~X1&Y1

Step 3: Calculate Ox-leading 1 bit in input x.

Step 4: Calculate Oa and Ob -leading 1 bit in decomposed operands A and B respectively. Ox,Oa,Ob determines the characteristic part of X,A,B.

Step 5: Assign Mx-bit next to leading 1 bit in x

Step 6: Assign Ma, Mb -bit next to leading 1 bit in A and B. Mx, Ma, Mb determines the mantissas of X, A and B.

Step 7: Log X=Ox.Mx Log A=Oa.Ma Log B=Ob.Mb

Step 8: Sum 1= Log X+ Log A
Sum 2= Log X+ Log B

Step 9:Assign Fxa=1 append mantissa of sum 1 after this bit, assign Fxb=1

append mantissa of sum 2 after this bit.
Step 10: If A or B is zero output is zero
Product of X and Y is Fxa+Fxb.

An example for multiplication of two numbers using Mitchell algorithm is given below:
Let us consider two values ten (1010) and six(0110) n=4 .
Step1:X1=1010; Y1=0110
Step 2: A=0010; B=0100
char and mantissa:
Ox=011 Oa=001 Ob=010 ; Mx= 010 Ma=000 Mb =000
Log values:

Log X=011.010 Log A=001.000 Log B=010.000

Sum 1=100.010 Sum 2= 101.010

Fxa=00010100 Fxb=00101000

Product=00111100(60)

C.ARCHITECHTURE FOR LOGARTHMIC MULTIPLIER USING MODIFIED OPERAND DECOMPOSITION:

The fig.1 has major blocks such as comparator, multiplexer , operand decomposition and Mitchell algorithm.
The comparator is accurate and determines the greater input value . The comparator output is used as selection line of the multiplexer. Then the operands are decomposed which consists of basic AND , NOT gates. Each MA block consists of leading one detector, Encoder , Left barrel shifter ,Right barrel shifter and zero detector. The zero detector checks if any input to MA is zero , if so then the output value of MA block is zero. Finally a ripple carry adder adds the values of the MA blocks to give final product.

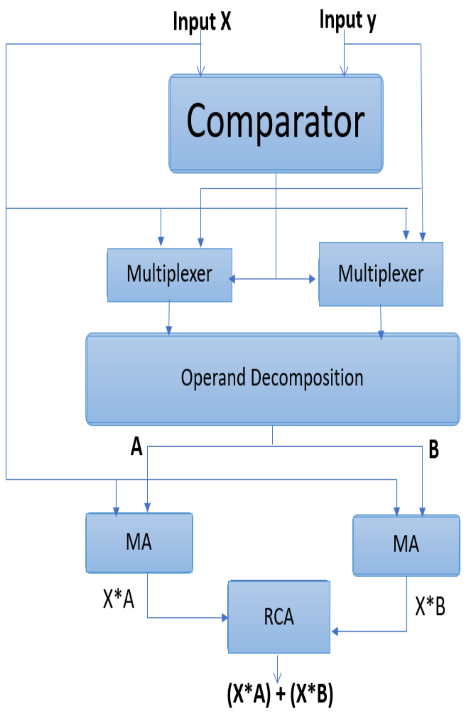
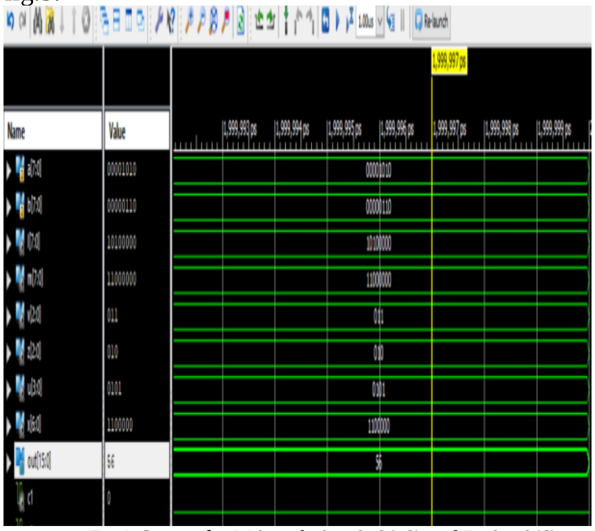


Fig.1 Architecture of logarithmic multiplier using modified operand decomposition technique

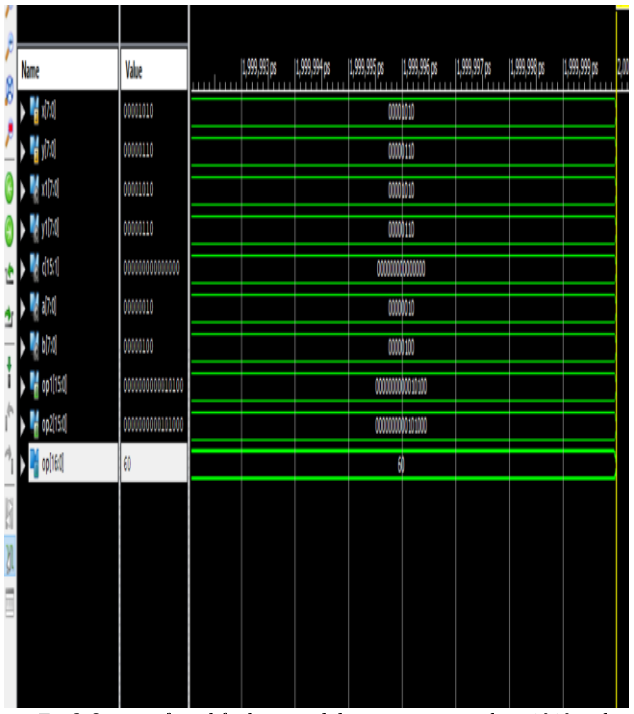
IV. RESULTS AND DISCUSSION:

The accuracy of MA has been significantly improved by adopting modified decomposition technique.
The proposed operand decomposition based Mitchell algorithm is implemented on FPGA spartan 6 and the algorithm is described by using verilog HDL language .xilinx ISE and model sim have been used for synthesis and simulation. The simulation result of Mitchell algorithm by adopting operand decomposition is shown in fig .2 and fig.3.



Name	Value
a[7:0]	00001010
b[7:0]	00000110
l[7:0]	10100000
m[7:0]	11000000
v[2:0]	011
z[2:0]	010
u[3:0]	0101
x[6:0]	1100000
out[15:0]	56
c1	0

Fig.2 Output for MA with A=1010(10) and B=0110(6)



Name	Value
x[7:0]	00001010
y[7:0]	00000110
x1[7:0]	00001010
y1[7:0]	00000110
c[15:1]	000000000000000
a[7:0]	00000010
b[7:0]	00000100
op1[15:0]	0000000000010100
op2[15:0]	0000000000101000
op[16:0]	60

Fig.3 Output of modified operand decomposition with x=1010 and y=0110

The error can be analysed by APE average error percentage AEP can be calculated by,
AEP=$\sum_{i=1}^{N} EPi\ /N$
where Epi is the error percent and N is the no of multiplications performed
The AEP for N=8 in Existing MA is about 3.77 % and the AEP for N=8 in proposed decomposition technique is

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 05, May-2019

about 1.6 % which clearly indicates the improvement in accuracy. The average error percentage for different multiplicand widths is given in table below which indicates the reduction of error in the proposed operand decomposition technique

AEP	Mitchells algorithm	Modified OD
4 - bit	---	1.627
8 - bit	3.76	1.65
16 - bit	3.91	2.10

Table I AEP for different multiplication widths.

V. CONCLUSION:

Thus a modified operand decomposition approach , algorithm for modified operand decomposition and an architecture for Mitchells algorithm based multiplier is proposed. The results indicate an improvement in area , power , and delay. The advantage in terms of accuracy is also observed. Applying operand decomposition as pre-processing step improves accuracy. Operand decomposition , does not require addition of correction term hence it can be easily combined with other methods to improve accuracy.

VI. REFERENCES:

[1] Siti Zarina Md Naziri, Rizalafande Che Ismail, Ali Yeon Md Shakaff, The Design Revolution of LogarithmicNumber System Architecture, IEEE international conference on Electrical, Electronics and System Engineering,(2014), pp. 5-10.

[2] A. Klinefelter, J. Ryan, J.Tschanz , B.H. Calhoun, Error-Energy Analysis of Hardware Logarithmic Approximation Methods for Low Power Applications, IEEE International Symposium on Circuits and Systems,(May(24-27))(2015), pp.2361-2364.

[3] L. K. Yu, D. M. Lewis, A 30-b Integrated Logarithmic Number System Processor, IEEE Journal of Solid StateCircuits, 26(October)(1991), pp. 1433–1440.

[4] F.J. Taylor, R. Gill, J. Joseph, A 20 Bit Logarithmic Number System Processor, IEEE Trans. Computers,37(February)(1988), pp. 190-200.

[5] I. Kouretas, C. Basetas, V. Paliouras , Low-power Logarithmic Number System Addition / Subtraction andThseir Impact on Digital Filters, IEEE Transactions on Computers, 62 (November)(2013), pp. 2196–2209.

[6] J.N. Mitchell, Computer Multiplication and Division using Binary Logarithms, IRE Trans. Electronic Computers, 11(August) (1962), pp. 512-517.

[7] D. K. Kostopoulos, An Algorithm for the Computation of Binary Logarithms, IEEE Transaction on Computers,40(November) (1991), pp. 1267–1270.

[8] H. Fu, O. Mencer, W. Luk, FPGA Designs with Optimized Logarithmic Arithmetic, IEEE Transactions onComputers, 59, 7(July) (2010), pp. 1000–1006.

[9] 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.

[10] Patricio Bulic, ZdenkaBabic , AleksejAvramovic. An Iterative Logarithmic Multiplier. Microprocessors and Microsystems 35 (2011) 23 – 33 , Elsevier

[11] JérémieDetrey, Florent de Dinechin, A VHDL Library of LNS Operators, The Thrity-Seventh AsilomarConference on Signals, Systems & Computers, 2(Nov. (9-12)) (2003), pp. 2227 – 2231.

[12] K. Johansson, O. Gustafsson L. Wanhammar, Implementation of Elementary Functions for LogarithmicNumber Systems, IET Computer & Digital Techniques, 2, 4(April) (2008), pp. 295–30.

[13] Julien Le Maire, Nicolas Brunie, Florent De Dinechin, Jean-Michel Muller, Computing floating-pointlogarithms with fixed-point operations, IEEE 23rd Symposium on Computer Arithmetic, Santa Clara, United States(July)(2016).

[14] V. Mahalingam, N. Rangantathan, Improving Accuracy inMitchell's Logarithmic Multiplication UsingOperand Decomposition, IEEE Trans. Computers, 55(December)(2006), pp. 1523-1535.