Design and implementation of Multiplier Accumulator (MAC) unit using Rounding Based Approximate (RoBA)

G. Indira[1*] and B.Srinivas[2]
[1]*Student,* [2]*Asst. Professor,*
[1,2]*Dept. of ECE, MVGR college of engineering (A), india-535005.*
[1]*induguri04@gmail.com,* [2]*srinivas.b@mvgrce.edu.in*

Abstract

In this paper, the implementation of the Multiplier-Accumulator (MAC) Unit is proposed by designing with Rounding-Based Approximate (RoBA) using Verilog HDL. The MAC unit is an important requirement for today's VLSI application like FIR filters, FFT, and Image processing applications. The computational complex part of the multiplication can be improved by rounding of the operand to the nearest exponent of two without compromising speed and energy consumption at a very small error rate. In line with, an efficient RoBA multiplier is designed, and using it, the proposed MAC unit performance has increased. The performance of the proposed MAC Unit using the RoBA Multiplier is calculated in terms of power, delay, and space area, and error rate and compared with other existing multipliers such as Baugh Wooley multiplier, Vedic multiplier, and RoBA multiplier. Among all the proposed MAC Unit using the RoBA multiplier has shown outperformance of delay of 1 ns, the power consumption of 5.3 mW, and a relative error rate of 0.29%.

Keywords: *Baugh Wooley multiplier, Vedic multiplier, RoBA Multiplier, Rounding-Based Approximate, MAC unit, MAC unit using RoBA Multiplier.*

1. Introduction

A multiplier is one in every of the foremost widely used arithmetic knowledge path operations in modern digital design. Multiplication is an important and computationally intensive operation. The multiplication operation is actually a gift in several elements of a digital system or computing device, most notably in signal process, graphics, and scientific computation. Booth algorithmic program could be a crucial improvement within the style of the signed binary multiplication MAC unit is an inevitable component in many digital signal processing (DSP) applications involving multiplications and/or accumulations. MAC unit [1] is used for high-performance digital signal processing systems. Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic determines the execution speed and performance of the entire calculation. Therefore, the functionality of the MAC unit enables high-speed filtering and other processing typical for DSP applications. A MAC unit consists of a multiplier and an accumulator containing the sum of the previous successive products. The design of a high-performance 64-bit Multiplier-Accumulator (MAC) is implemented using Verilog HDL [2] in this pa-per. Once partial products are generated, they need to be classified and supplemental during a systematic manner with intense less delay. The approximation may be performed using different techniques such as allowing some timing violations and function approximation methods or a combination of them [3], [4]. In the class of performing approximation

ways, a number of approximating arithmetic building blocks, such as adders and multipliers, at different design levels have been suggested. In [5], two approximate 4:2 compressors for utilizing in a regular Dada multiplier were designed and analyzed. In [6], and accuracy configurable multiplier architecture (ACMA) was suggested for an error-resilient system.

2. Literature Survey

In this section, I will reassess some of the earlier works in the discipline of approximate multipliers. In [7] by using a technique called Broken Array Multiplier (BAM) an approximate multiplier and an approximate adder were proposed. By applying the BAM approximation method of [7] to the conventional modified Booth Multiplier can be presented as a signed approximate Booth multiplier as [8]. The approximate multiplier, when compared with a regular Booth multiplier can have an advantage of less power consumption saving from 28% to 58.6% and from 28% to 58.6% in terms of area reduction. Kulkarni et al. [8] suggested an approximate multiplier consisting of a number of 2 × 2 inaccurate building blocks that saved the power by 31.8%–45.4% over an accurate multiplier.

In [9] for speculation purposes in Pipelined processors was designed by using an approximate signed 32-bit multiplier. While possessing a probability error of around 14%, it was 20% quicker than a full-adder based tree multiplier. In [3], an error-tolerant multiplier, which evaluated the imprecise outcome by splitting the multiplication into one precise and one imprecise part, was introduced, in which the accuracies for distinct bit-width were disclosed. A power efficiency saving of 50% was disclosed in the instance of a 12-bit multiplier. By using a regular Dadda Multiplier two approximate 4:2 compressors were designed and examined in [10]. Most of the approximate multipliers that are proposed earlier are dependent on either adjusting the structure or complexity reduction of a specific accurate multiplier. In [4] an approximate 4x4 WTM has been suggested which uses an inaccurate 4:2 counter. In inclusion, an error correction unit for correcting the results has been suggested. To design a larger multiplier, this 4x4 inaccurate Wallace multiplier can be in an array structure. In this paper, similar to [11], I proposed accomplishing the approximate multiplication through simplifying the performing operation. The differences between this paper and [11] is that, although the underlying principles in both works are similar to the unsigned numbers, The mean error of our proposed approach is smaller, In addition, when performing multiplication for signed number, I also propose some approximation techniques.

3. Proposed Method

3.1 Design of RoBA multiplier

The main idea of this proposed work is to design RoBA multiplier [12] to make use of simple operation to multiply the numbers when they are two to the power n (2n). The detailed operation of this multiplier is, first, to denote the rounded numbers of the input A and B by Ar and Br, respectively. The multiplication of A by B can be written as

$$A * B = (Ar - A) * (Br - B) + Ar * B + Br * A - Ar * Br \qquad (1)$$

The multiplications of Ar*Br, Ar*B, and Br*A can be performed by the shifting operation. The implementation of (Ar A)*(Br B) will become complex. So that I can exclude those terms and simplify the operation. Hence, the multiplication operation used is:

$$A * B \approx Ar * A + Br * A - Ar * Br \qquad (2)$$

This multiplication can be executed using three shifters, adder, and a sub tractor. By using this method, I have to obtain the nearest 2^n form of the inputs A and B. If the inputs (A or B) are in the form of $3 * 2^{p-2}$(where p is the positive integer greater than 2) has two nearest 2^n values with equal differences with 2^p and 2^{p-1}. Both 2^p and 2^{p-1} have the same result on the accuracy of the multiplier. By selecting the largest one(except p=2) results in the smaller hardware implementation. The only inconsistency is for three, in this case, two is considered as its nearest 2^n value. The final output of the RoBA multiplier[12] may be either larger or smaller than the exact result depending on the magnitudes of rounded values Ar and Br compared with A and B, respectively. RoBA multiplier gives the exact values only for the positive integers as the rounded values of negative numbers are not in the form of 2^n. So first you have to identify the sign and remove it before multiplying. After the completion of multiplication, you have to add the appropriate sign to the output.

Sign Detector: It detects the sign of the inputs and gives the sign for output of the multiplier. If the MSB is '0', then it will be considered as a positive integer. If MSB is '1', then it is a negative integer.

Rounding: It rounds the given input values to the nearest 2^n value.

Shifters: The shifters are used to determine the products Ar × B, Br × B, Ar × Br.

Adder: In order to calculate the summation of Ar × B and Br × A, a single 2^n-bit Kogge-Stone adder is used.

Sub tractor: The inputs of the sub tractor block are the output of the Kogge-Stone adder and the result of Ar × Br are the output of the absolute value of the multiplier output.

Sign Set: It sets the sign of the output depending on the input sign that is detected in the sign detector block.

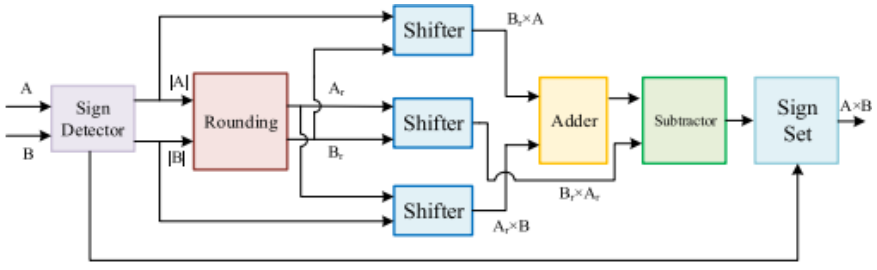

Figure 1: Block Diagram of RoBA Multiplier

3.2 MAC unit using RoBA Multiplier

A Digital signal Processor(DSP) is a significant block that executes various Signal processing applications such as Convolution, Fourier transform, Discrete Cosine transform(DCT),Discrete wavelet Transform(DWT),Filter and De- signs, Multimedia Information processing and so on. A MAC unit is present in every Digital Signal Processor. To perform the most crucial operations in Digital signal processing, A MAC unit[1] is accomplished to operations like multiplication and accumulations processes repeatedly. A MAC unit consists of clock and Reset options to Control its operations in a more versatile way.

A MAC unit operates absolutely independent of the Central Processing Unit, as a result there is very less load on the CPU to handle. A Digital Filter needs to perform Multiplication and accumulation operations repeatedly, for this reason I use a MAC unit for High-Speed filtering operations. The Applications of DSP which are related to Optical Communication needs an extremely High speed Processing of

Large amounts of Digital Data. And operations like Fast Fourier transform also accomplishes operations like Addition and Multiplication. The Computational Capacity and Entire Power Efficiency of the Digital Signal Processor can be estimated by using this MAC unit .So, The most pivotal task it to design a MAC unit which is Power Efficient, Sufficiently speed in per- forming tasks and Occupies less Area. The inputs to the MAC are taken from the Address of the Memory and Supplied to the Multiplier Block of a MAC after performing Multiplication The result is passed to an Adder and Then eventually the Final result will be stored into Target Memory.The different blocks present in the MAC multiplier are

Multiplier: It multiplies the two inputs that are given to the multiplier. This unit uses the conventional multiplier unit, which consists of multiplication of multiplier and multiplicand based on adding the generated partial products and to compute the final multiplication.

Accumulator: Accumulator stores the output of the adder in the register for further addition. The Accumulator should be fast enough in order to implement along with the fast adders.

Adder: It adds the multiplier output and the value stored in the accumulator.

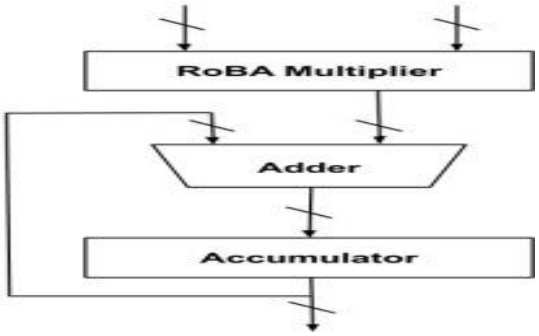

Figure 2: Block Diagram of MAC unit using RoBA Multiplier

4. Proposed Method Results and Simulations

4.1 RoBA Multiplier results:

The Register Transfer Level(RTL)[13] and Simulated waveforms of the RoBA Multiplier is shown in the Figures.3,4 respectively. These are generated using Verilog HDL in Xilinx ISE. Figure.3 consists of Sign Detector, Rounding, Shifters, Adder, Sub tractor and Set Sign blocks.

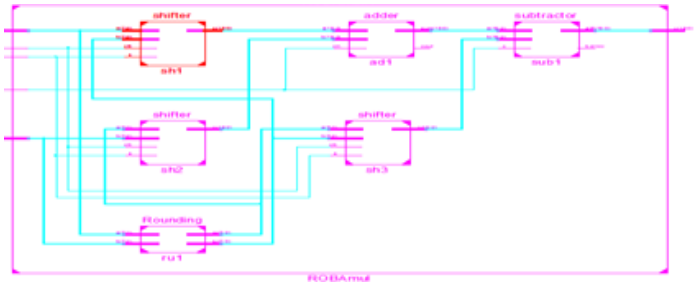

Figure 3: RTL of RoBA Multiplier

Figure 4: RoBA Multiplier simulated results

4.2 MAC unit using ROBA multiplier results

By using MAC unit and RoBA multiplier together improves the speed and reduces the power consumption. Fig.5 represents the RTL and Fig.6 represents the Simulated waveform of the MAC unit using RoBA Multiplier.

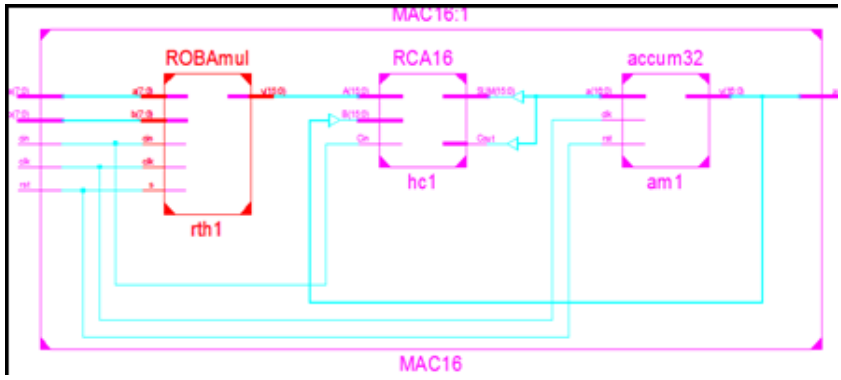

Figure 5: RTL of MAC unit using RoBA Multiplier

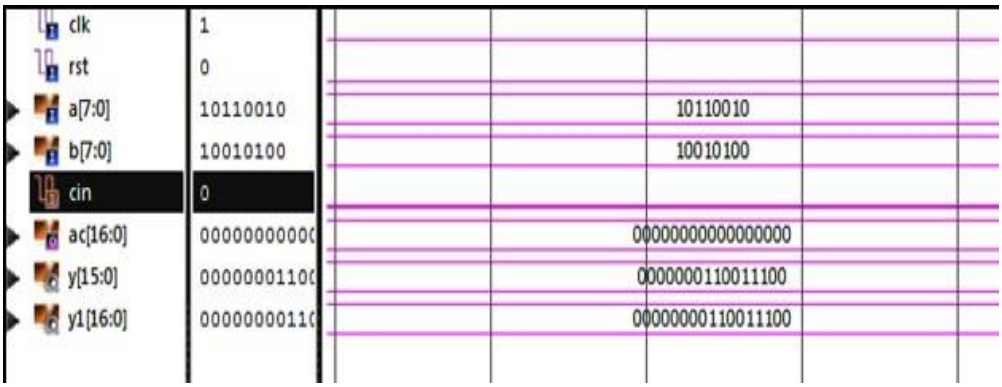

Figure 6: MAC unit using RoBA Multiplier with RST = 0

Figure 7: MAC unit using RoBA Multiplier with RST = 1

Table 1 shows the Design Parameters comparison of the RoBA and MAC unit using RoBA Multiplier. The considerable design parameters are Number of slices on the board, Number of Slice Flip Flops, Number of 4 input Luts, Number of bonded IOBs, Number of GCLKs. After the implementation, from this table I can conclude that MAC unit using RoBA Multiplier shows the best performance than RoBA Multiplier.

Logic Utilization	Used		Availabe	Util izati on	
	MAC unit using RoBA	RoBA		MAC unit using RoBA	RoB A
Number of slices	148	112	4656	3%	2%
Number of Slice FlipFlops	97	80	9312	1%	0%
Number of 4 input Luts	275	202	9312	2%	2%
Number of bonded IOBs	36	36	232	15%	15%
Number of GCLKs	1	1	24	4%	4%

Table 1: Utilization of MAC unit using RoBA and RoBA multiplier

5. Comparing other multipliers with proposed design

In this section I are comparing our proposed method with other kinds of multipliers like Baugh-Wooley and Vedic multipliers.

Figure 8:Baugh-Wooley multiplier for full precision

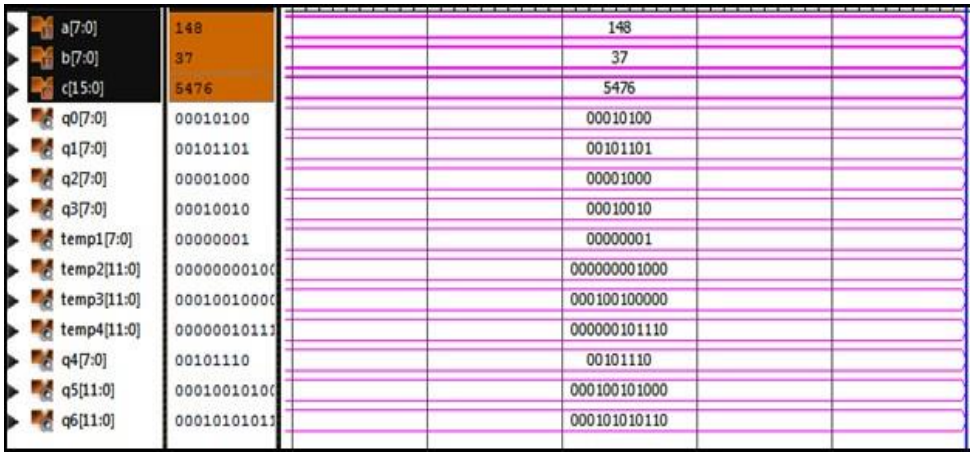

Figure 9: Vedic Multiplier

The Fig.8 shows the simulated results of Baugh-Wooley Multiplier and Fig.9 shows the simulated results of Vedic Multiplier. The comparisons are as follows:

1. **In terms of architectures**, Baugh-Wooley Multiplier contains 40 IOBs(area), Vedic Multiplier contains 50 IOBs(area), RoBA contains 35 IOBs(area), MAC unit using RoBA multiplier contains 36 IOBs(area)2. As compared to other multipliers, the architecture of the RoBA and MAC unit using RoBA Multipliers are smaller.

2. **In terms of performance(delay)** due to very large architectures, all the above-mentioned multipliers are computationally very slow meant for within a certain time interval they didn't compute a high number of multiplications. But this proposed method consists of a MAC unit so that it performs a very high number of multiplications with a delay of 1 ns only.

3. **In terms of power consumption** due to large architectures require more number of components. If a design consists of a huge number of components it consumes very high power. The bus proposed method consumes 5.37 mW power.

4. **In terms of Error rate** of the above-mentioned multipliers, the error rate is getting higher if the number of bits of an operand is higher. This proposed design relative error rate is smaller than **1%** i.e., 0.29%.

5. **In terms of application**, the above-mentioned multipliers are used for only multiplications. But this proposed design is not only used for multiplication and also used for image processing This low power, but fast and area-efficient multiplier can also be used for FIR filter design, MAC design.

The Table.2 shows the design parameters of RoBA, MAC unit using RoBA , Baugh-Wooley and Vedic multipliers. The following table compares the Max Output time required(ns), Delay(ns), Power(mW), Area(IOBs) and Error rate among the four multipliers.

Types of Multipliers	Max Output time required(ns)	Delay (ns)	Power (mW)	Area (IOBs)	Error rate
MAC unit using RoBA	4.252	1.00	5.37	36	0.29
RoBA	11.436	1.54	9.21	35	1
Baugh-Wooley	15.181	1.59	17.42	40	1
Vedic	21.470	2.63	17.58	50	1

Table 2: Comparison of Design parameters

6. Conclusion

Multipliers for low power applications were enforced. Three basic algorithms namely Baugh-Wooley, Vedic multiplier, and ROBA multipliers were implemented using Verilog HDL in Xilinx14.2. The Power, Delay, and space area unit calculated for these algorithms. The proposed multiplier, which had high accuracy, was based on rounding off the inputs in the form of 2^n. This hardware consists of three implementations. One implementation of unsigned operations and the remaining two for signed operations. The design-related performance characteristics are computed for this proposed design and compared with other few multipliers. The results expressed that in every design aspect MAC unit using RoBA Multiplier architectures out-performed the corresponding other multipliers. This proposed method has a delay of **1 ns**, power consumption **5.37 mW** and a relative error rate is **0.29%**. Finally the proposed overcomes all.

7. Future Scope

Using the proposed design so far I have designed a multiplier that performs with less delay, high computational speed, and consumption of low power. If the number of components is reduced furtherly then resource utilization will decrease on the FPGA board. If I will use this design as an image processing unit, there will be a large scope to implement image processing algorithms. If machine learning techniques are included in the future these kinds of multiplication algorithms will be very useful to implement a few specific tasks.

References

[1] M. Sai Kumar, D. A. Kumar, and P. Samundiswary, "Design and performance analysis of multiply-accumulate (mac) unit," in *2014 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2014]*, pp. 1084–1089, 2014.

[2] M. D. Ciletti, "Advanced digital design with the verilog hdl," 2010.

[3] W. L. G. K. Y. Kyaw and K. S. Yeo, "Low-power high-speed multiplier for error- tolerant application,"

[4] A. R. V. Gupta, D. Mohapatra and K. Roy, "Low-power digital signal processing using approximate adders,"
IEEE Trans. Compute.-Aided Design Integer. Circuits Syst., vol. 32, no. 1, p. 1.

[5] K. Bhardwaj and P. S. Mane, "Acma: Accuracy-configurable multiplier architecture for error-resilient system- on-chip," *in Proc. 8th Int. Workshop Reconfigurable Common.-Centric Syst.-Chip*, p. 1, 2013.

[6] H.R.MylerandA.R.Weeks, "The pocket handbook of image processing algorithms in c," in *Englewood Cliffs*, (NJ, USA: Prentice-Hall), 2009.

[7] S. M. F. H. R. Mahdiani, A. Ahmadi and C. Lucas, "Bio-inspired imprecise computational blocks for efficient vlsi implementation of soft-computing applications,"

[8] M. S. A. F. Farshchi and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing,"

[9] B. J. P. D. R. Kelly and S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," *in Proc. Conf. Design Archit. Signal Image Process.*, p. 97, 2009.

[10] P. M. A. Momeni, J. Han and F. Lombardi, "Design and analysis of approximate compressors for multiplication," IEEE Trans. Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.

[11] J. N. Mitchell, "Computer multiplication and division using binary logarithms, IRE Trans. Electron. Comput., vol. EC-11, no. 4, pp. 512–517, Aug. 1962."

[12] M. K. Reza Zendegani, "Roba multiplier:a roundingbasedapproximate multiplier for high-speed yet energy- efficient digital signal processing," Inc.XILINX, "Rtl and technology schematic viewers tutorial," 2011.