JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Efficient Design of Truncation and Rounding Based Approximate Multiplier

[1]M. Harika Reddy, [2]S. Santhoshi Bhavani
[1]P G Scholar, [2] Assistant Professor
[1, 2] Department of Electronics and Communication Engineering
[1, 2] Avanthi Institute of Engineering and Technology, Cherukupally (V), Bhogapuram (M), Vizianagaram, Andhra Pradesh, India

Abstract: The increasing complexity of the DSP systems demands higher computational performance in its architecture. But the traditional DSP arithmetic has limits in terms of speed of calculations. Moreover, in some applications, speed is more important than accuracy. To further enhance performance, approximate arithmetic circuits are designed with some loss of accuracy to reduce power consumption and increase speed. Furthermore, these approximate circuits have been considered for error-tolerant applications. In this paper, we propose a truncation and rounding-based approximate multiplier. In this multiplier, the operands are rounded to the nearest exponent of two and adder, and shifters are used at the final multiplication stage. This proposed multiplier simplifies multiplication operation, thus reducing area, power, and increasing speed. Simulation results reveal that the proposed approximate multiplier improves delay, area, and power consumption up to 9.3%, 37%, and 6.6%, respectively, compared to the ROBA multiplier. The proposed multiplier was synthesized and simulated using Xilinx Vivado 2016.4 environment.

Index Terms – Truncation, Rounding, Adders, Shifters, Approximate Computing, DSP.

I. INTRODUCTION

Electronics are continuously upgraded, particularly in digital signal processing and multimedia applications. Nowadays, they are growing day by day. In this DSP and Multimedia applications, it requires many arithmetic multiplications. Different types of multipliers perform these multiplication operations. Multipliers play a vital role in most high-performance systems. When considering digital signal processors and microprocessors, a multiplier is a crucial component for logical operation. In DSP methods, we utilize Fourier Transformations and discrete cosine transformations in DWT. Multiplication is more delayed and power-consuming as we evaluate other arithmetic operations. The main challenge in designing the multiplier is reducing the delay and minimizing the area with efficient power usage. To accomplish this test by decreasing the number of partial products. Even if the number of partial products is reduced by utilizing a higher radix booth encoder, the number of complex multipliers that are expensive to generate increases concurrently. By utilizing the Canonic Sign Digit representation (CSD), we can encode minimum non-zero digit coefficients [1]. Canonic Sign Digit multipliers give fewer non-zero unfinished products that diminish their switching action. In CSD, encoding additionally has different constraints. Folding technique that limits silicon zone by utilizing time-multiplexing action into single practical units, for instance, adders, multipliers is not conceivable as the Conic Sign Digit based multipliers are hard-wired to particular co-efficient.

CSD-based programmable multiplier arrangement was specially planned for social events of pre-chosen coefficients that give particular features. The degree of ROM is utilized to accumulate the get-together of coefficients. It diminishes the area and power usage of the circuit. In any case, this multiplier design is not versatile. Since the fractional items age unit is intended for a specific gathering of coefficients and can not be reutilized for different gatherings, also, this method is difficult to extend massive groups of predetermined coefficients accomplished simultaneously for high efficiency [2]-[7]. Modified Booth (MB) encoding solves the previously mentioned limitations and decries the count of the partial products to half. We can minimize the area with efficient power usage and delay by reducing the partial products. In [8], Kim et al. anticipated a procedure like laying out more capable MB multipliers for social affairs of pre-chosen coefficients with measure up to obstructions indicated in the before section. In [9]-[11], multipliers are incorporated in butterfly units of FFT processors utilize coefficients which standard to put away in ROMs.

Energy minimization is one of the main design requirements in almost any electronic system, especially portable ones such as smartphones, tablets, and different gadgets. Therefore, achieving this minimization with minimal performance (speed) penalty is highly desired. However, Digital signal processing (DSP) blocks the re-key components of these portable devices for realizing various multimedia applications. The computational core of these blocks is the arithmetic logic unit, where multiplications have the greatest share among all arithmetic operations performed in these DSP systems. Therefore, improving the speed and power/energy-efficiency characteristics of multipliers plays a key role in improving the efficiency of processors. Furthermore, many DSP cores implement image and video processing algorithms where final outputs are images or videos prepared for human consumption. This fact enables us to use approximations for improving speed/energy efficiency.

It originates from the limited perceptual abilities of human beings in observing an image or a video. In addition to the image and video processing applications, there are other areas where the exactness of the arithmetic operations is not critical to the system's functionality. Using approximate computing allows the designer to make trade-offs between accuracy, speed, and power/energy consumption. The approximation can be applied to the arithmetic units at different design abstraction levels, including circuit, logic, architecture levels, algorithm, and software layers. The approximation may be performed using different techniques such as allowing some timing violations (e.g., voltage over-scaling or over-clocking) and function approximation methods (e.g., modifying the Boolean function of a circuit) or a combination of them. In the function approximation method category, several approximating arithmetic building blocks, such as adders and multipliers, have been suggested at different design levels.

In this paper, we focus on proposing a high-speed low power/energy yet approximate multiplier appropriate for error-resilient DSP applications. The proposed approximate multiplier, which is also area efficient, is constructed by modifying the conventional multiplication approach at the algorithm level, assuming rounded input values. We call this ROunding-Based Approximate (ROBA) multiplier [17]. The proposed multiplication approach applied to unsigned multiplications is presented. The efficiencies of these structures are assessed by comparing the delays, power, and areas with ROBA multipliers. The contributions of this paper can be summarized as follows:

1) Presenting a new scheme for Rounding based approximate multiplier by modifying the existing ROBA multiplier.

2) Describing the hardware architectures of the proposed and existing approximate multipliers for unsigned operation.

II. BACKGROUND

The main goal of the paper organization is to combine the space between theoretical studies into the implementation of digital-level algorithms.

The Author C.Grabbe [10] presented in his paper about cryptography as well as its application areas. He also delivered drawbacks about finite field multiplication and other downsides like time-consuming operations. C.Grabbe designed as well as an implemented multiplier which is a high-performance multiplier named GF (S/sup233/) multiplier for FPGA realization. C.Grabbe implemented the multiplier by utilizing Karatsuba Algorithm. For achievement of superior performance, C.Grabbe utilized additional pipelining on 233-bit advanced finite field multiplier

C.Paar[15], given key implementation of effective parallel structure multiplier by considering Galois field. C.Paar implemented a multiplier working in the presence of composite fields GF. To achieve effective performance, he adopted the Karatsuba-of-man algorithm. Finally, C.Paar delivered a multiplier with minimum gate count and also succeeded in delivering a faster computational multiplier with less delay

C.Rebeirno[16] presented in his paper introduced the hybrid procedure that reduces the computational time. Rebeirno implemented a hybrid technique with a minimum number of gates for hardware realization. In a cryptography system, the main downside and key issue are SCA. Rebeirno delivered the hybrid technique by diminishing some notable demerits like the cost required for implementation, and his technique includes the utilization of LUTs.

A.Hasan [13] presented a procedure to reduce consumption time and implement a cryptographic scheme with minimum logic gates to achieve ultra-high-speed performance. Hasan also presented the key issue that diminishes the performance. Since millions of gates sometimes lead to faulty output because of malicious attacks. Hasan presented the procedure that automatically finds the incorrect outputs that occurred due to certain faults.

B.Sunar [12] presented the procedure to build a sub-quadratic multiplier even in the field. Sunar adopted the different algorithms for his implementation of the technique. The different algorithm includes convolution algorithm as well as syntax procedures like nesting. To achieve the fast convolution algorithm, he studied and adopted the context of polynomial multiplication. Sunar delivered the key concept by effectively utilizing the recursive procedure. From this procedure, he generated the six alternative constructions. One of them has similar aspects to the Karatsuba algorithm.

G.Zhou, H.Michalik, and L.Hinsenkamp[14] address the modification to upgrade the performance to the next level. G.Zhou analyzed and implemented the work on complexity analysis in Field Programmable Gate Array and application-specific integrated circuits using the Karatsuba-of-man multiplier. The author evaluated LUT complexity and delivered the point that his multiplier of bit-parallel devours the slightest resources to deliver high performance.

III. PROPOSED APPROXIMATE MULTIPLIER

It is a truncation and rounding-based approximate multiplier, which reduces the number of partial products by truncating each input operand based on their leading one-bit position. In the proposed design, multiplication is performed by shift, adding operations resulting in large improvements in energy consumption and area occupation compared to the ROBA multiplier. To improve the total accuracy, input operands of the multiplication part are rounded to the nearest odd number. Because input operands are truncated based on their leading one-bit positions, the accuracy becomes weakly dependent on the width of the input operands. Therefore, greater improvements in design parameters can be achieved as the input operand widths increase. Its design parameters are compared with the ROBA multiplier to evaluate the efficiency of the proposed approximate multiplier. The Block diagram of the proposed approximate multiplier is shown in Fig. 1. The proposed approximate multiplier's mathematical equation represents equation (1).

$$A \times B \approx 2^{K_1+K_2} \times (1 + A_t + B_t + ((A_{apx_r} \times B_{apx}) + (B_{apx_r} \times A_{apx}) - (A_{apx_r} \times B_{apx_r}))) \quad (1)$$

Where A, B are the inputs

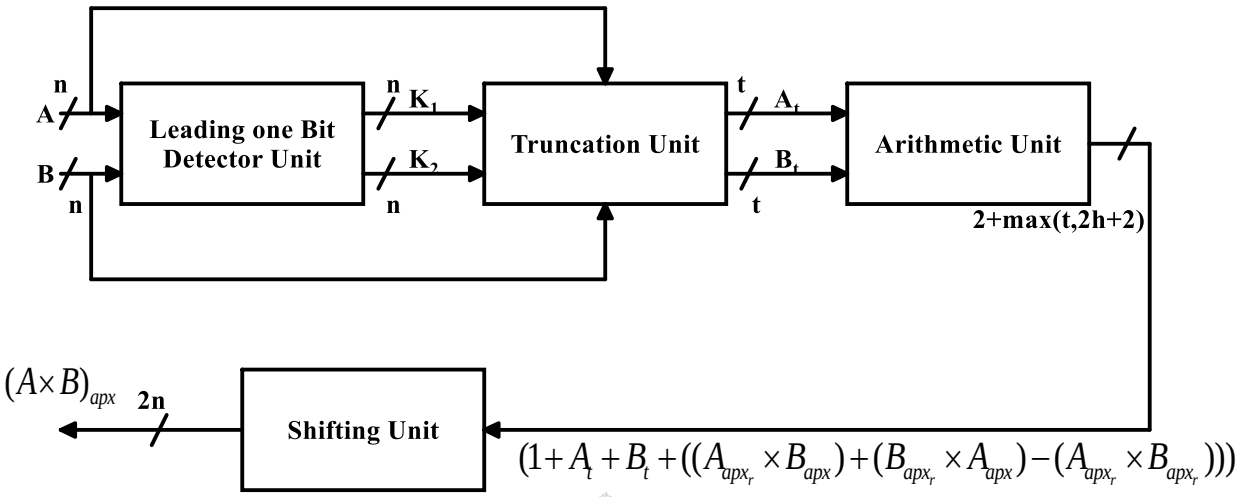

Fig. 1 Block Diagram of proposed approximate Multiplier

The architecture of the Proposed approximate Multiplier consists of a lead one detector, truncation unit, arithmetic unit, shift unit, and sign and zero detector unit. These individual units are explained below:

The LOD is used in logarithmic converters to find the position of the leading 'one' bit in the integral, and the fractional parts of a logarithm operation are determined with the help of LOD. Leading-One Detector (LOD) is important as they are used for the normalization process in a floating-point multiplication, logarithmic multiplication, and the logarithmic converter. An efficient and low-power LODs is a demand for the logarithmic converter to perform a DSP operation.

When we truncate a number, we find an estimate for the number without doing any rounding. To truncate a number, we miss off digits past a certain point in the number, filling-in zeros if necessary, to make the truncated number approximately the same size as the original number. To truncate a number to 1 decimal place, miss all the digits after the first decimal place. To truncate a number to 2 decimal places, miss all the digits after the second decimal place. To truncate a number to 3 significant figures, miss all the digits after the first 3 significant figures (the first non-zero digit and the next two digits). Fill in any spaces with zeros to make the number approximately the same size as the original value.

An arithmetic logic unit represents the fundamental building block of the central processing unit of a computer. It consists of add and shifter operations. For example, in the Shift Unit, the output of the Arithmetic Unit is shifted to the left by $K_1 + K_2$.

Rounding a number means replacing it with a different number that is approximatel4 equal to the original but has a shorter, simpler, or more explicit representation. Now, if we do this rounding using a two 4bits ad, we get a rounded 4-bit output.

Rounding is often done to obtain a value that is easier to report and communicate than the original. Rounding can also be important to avoid misleadingly precisely reporting a computed number, measurement, or estimate.

On the other hand, rounding of exact numbers introduces some round-off error in the reported result. Rounding is almost unavoidable when reporting many computations – especially when dividing two numbers in integer or fixed-point arithmetic, when computing mathematical functions such as square roots logarithms and sines, or when using a floating-point representation with a fixed number of significant digits. In a sequence of calculations, these rounding errors generally accumulate, and in certain ill-conditioned cases, they may make the result meaningless.

Accurate rounding of transcendental mathematical functions is difficult because the number of extra digits that need to be calculated to resolve whether to round up or down cannot be known in advance. This problem is known as "the table maker's dilemma".

Rounding is similar to the quantization that occurs when numbers or digital signals must encode physical quantities.

IV. RESULTS ANALYSIS

This section describes the performance of the proposed and existing approximate multipliers using Verilog simulated in Xilinx Vivado 2016.4. The simulated output of multipliers is shown in Fig. 2. From Table 1, it is revealed that the proposed multiplier reduces the power, delay, and area compared to the existing ROBA multiplier. A performance comparison between the proposed approximate Multiplier and ROBA multiplier is illustrated in Fig. 3. Results reveal that the proposed approximate multiplier improves delay, area, and power consumption up to 9.3%, 37%, and 6.6%, respectively, compared to the ROBA multiplier.

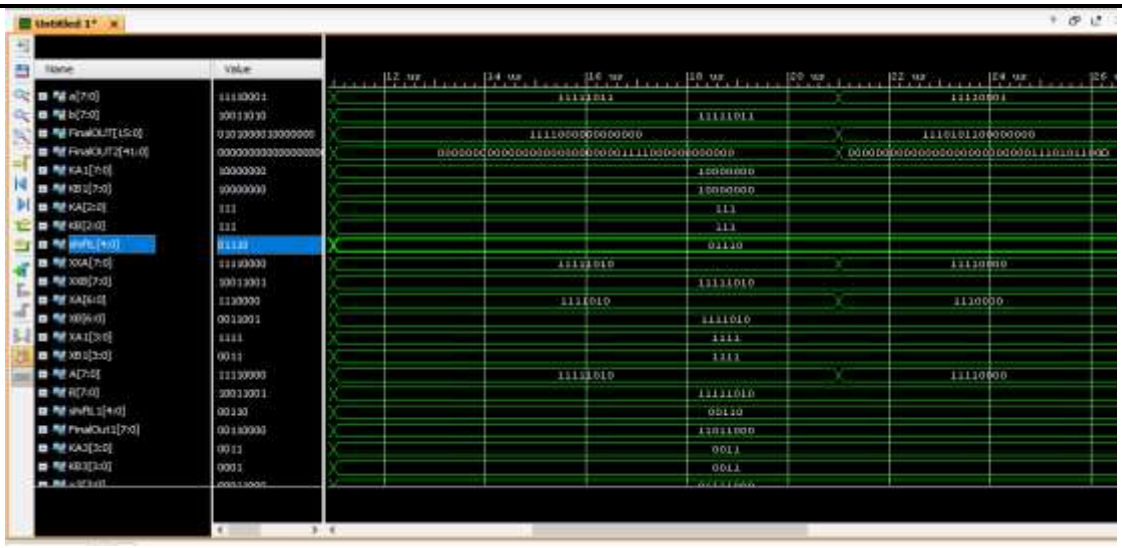

Fig. 2 simulation output of proposed approximate multiplier

Table 1 Comparison Table between the proposed approximate multiplier and existing ROBA multiplier

Name of the System	Proposed Multiplier	Existing ROBA Multiplier [17]
Number of LUTs	113	182
Power (*W*)	9.59	10.26
Delay (*ns*)	12.2	13.5

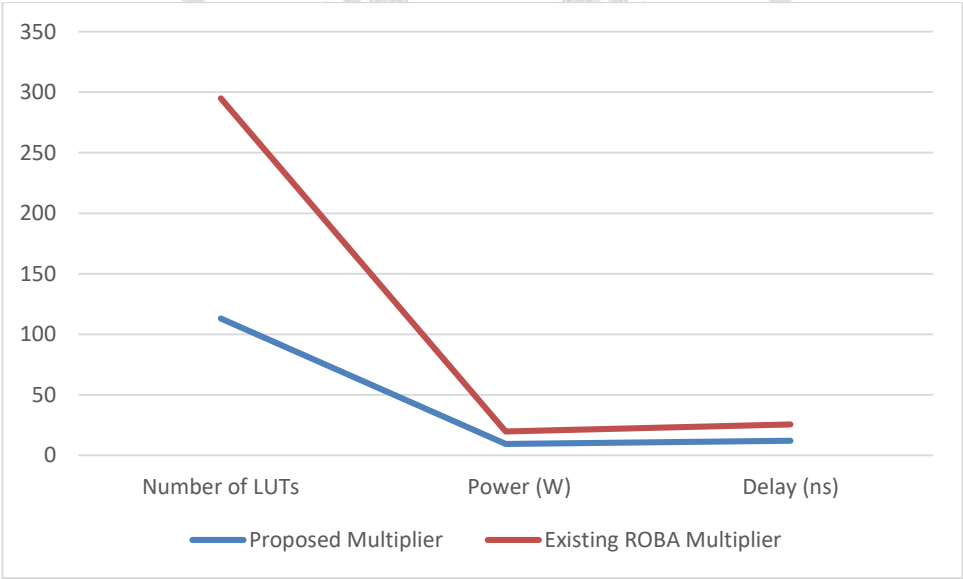

Fig. 3 performance comparison between proposed approximate Multiplier and ROBA multiplier

V. CONCLUSION

Truncation and the rounding-based approximate multiplier were proposed in the paper. The results show that the proposed multiplier performs better in terms of area, power, and delay. Moreover, reduced area resulted in less delay. The system shows a 9.3 % speed reduction, occupying only 38 % of the area and consuming 6.6 % of the existing ROBA multiplier power. The above results were obtained after simulating the proposed and existing multipliers using the Xilinx Vivado 2016.4 simulator at an operating voltage of 1.0V.

REFERENCES

[1] G. W. Reitwiesner, "Binary arithmetic," Advances in Computers, vol. 1, pp. 231–308, 1960.
[2] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. John Wiley & Sons, 2007.
[3] K. Yong-Eun, C. Kyung-Ju, J.-G. Chung, and X. Huang, "CSD based programmable multiplier design for predetermined coefficient groups," IEICE Trans. Fundam. Electron. Communication Computer. Sci., vol. 93, no. 1, pp. 324–326, 2010.
[4] O. Macsorley, "High-speed arithmetic in binary computers," Proc. IRE, vol. 49, no. 1, pp. 67–91, Jan. 1961.
[5] W.-C. Yeh and C.-W. Jen, "High-speed booth encoded parallel multiplier design," IEEE Trans. Comput., vol. 49, no. 7, pp. 692–701, Jul. 2000.
[6] Z. Huang, "High-level optimization techniques for low-power multiplier design," Ph.D. dissertation, Department of Computer Science, University of California, Los Angeles, CA, 2003.

[7] Z. Huang and M. Ercegovac, "High-performance low-power left-to-right array multiplier design," IEEE Trans. Comput., vol. 54, no. 3, pp. 272–283, Mar. 2005.
[8] Y.-E. Kim, K.-J. Cho, and J.-G. Chung, "Low power small area modified booth multiplier design for predetermined coefficients," IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol. E90-A, no. 3, pp. 694–697, Mar. 2007.
[9] C. Wang, W.-S. Gan, C. C. Jong, and J. Luo, "A low-cost 256-point FFT processor for portable speech and audio applications," in Int. Symp. on Integrated Circuits (ISIC 2007), Sep.2007, pp. 81–84.
[10] C. Grabbe, M. Bednara, J. Teich, J. Gathen, and J. Shokrollahi. FPGA Designs of Parallel High Performance GF (2233) Multipliers. In ISCAS (2), pages 268– 271, 2003.
[11] A. Weimerskirch and C. Paar. Generalizations of the Karatsuba Algorithm for Efficient Implementations. Ruhr-Universit¨at-Bochum, Germany. Technical Report, 2003.
[12] B. Sunar. A Generalized Method for Constructing Subquadratic Complexity GF (2k) Multipliers. IEEE Trans. Computers, 53(9):1097–1105, 2004.
[13] A. Reyhani-Masoleh and M. A. Hasan. Low Complexity Bit Parallel Architectures for Polynomial Basis Multiplication over GF (2m). IEEE Trans. Computers, 53(8):945–959, 2004.
[14] G. Zhou, H. Michalik, and L. Hinsenkamp., "Efficient and High-Throughput Implementations of AES-GCM on FPGAs". International Conference on Field- Programmable Technology (FPT), 2007, 185-192
[15] C. Paar. Efficient VLSI Architectures for Bit Parallel Computation in Galois Fields. PhD thesis, University at GH Essen, 1994.
[16] C.Rebeiro and D.Mukhopadhyay, Power attack resistant efficient FPGA architecture for karatsuba multiplier, in Proc. of Int. Conf. VLSI Des.,2008,pp.706-711 N.S.Kim, T.Mudge, and R.Brown, —A 2.3 Gb/s fully integrated and synthesizable AES rinjdael core, in proc. IEEE Custom Integrated Circuits Conf., 2003, pp.193-196.
[17] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, and M. Pedram (2017) RoBa multiplier: A rounding-based approximate multiplier for high-speed yet energy-efficient digital signal processing. IEEE Transaction Very Large Scale Integration (VLSI) Systems 25(2):393–401.