Journal of Engineering Sciences
Vol 11, Issue 6,June/2020
ISSN NO:0377-9254

AN APPROXIMATE UNSIGNED MULTIPLIER WITH SIGNIFICANT LOGIC BASED ERROR RECOVERY

S. Mehathab[1] , O.Homa Kesav[2]

[1]*Dept.of ECE , Annamacharya Institute of Technology& Sciences Kadapa, Andhra Pradesh, India.*

[2] *Dept. of ECE , Annamacharya Institute of Technology& Sciences kadapa ,Andhra Pradesh ,India.*

***Abstract*-In approximating, the requirement for accurate results is ignored because of some other applications with better performance in terms of area or delay. Multipliers are key arithmetic circuits in many of these applications including digital signal processing (DSP). Here in this paper we propose an approximate multiplier circuit with two various architectures where one is designed by modifying the design of the circuit by adding AND-OR logic approximation in the partial product generation stage and a dual quality adder at the final stage of the multiplication. The other one is by adding the same AND-OR logic approximation in the partial product reduction stage where it is applied tow only least significant part and accurate adders are used in the remaining for final outcome. The proposed approximate multipliers have been shown that both have a lower area and one with better accuracy other with delay than an exact Wallace multiplier and existing approximate designs. Functional analysis has shown that on a statistical basis, the proposed multipliers have considerable error distances and thus, they achieve a high accuracy and better area. To show the effectiveness of the work we had implemented and image processing application with the help of Xilinx System generator and Matlab. The total designs are done and implemented in Xilinx ISE 14.7 with Verilog HDL coding.**

Keywords: Approximate circuit, AND-OR logic, Dual quality adder , Accuracy.

1 .INTRODUCTION

The pervasive, portable, embedded and mobile nature of present age computing systems has led to an increasing demand for low power along with reduced area and delay and high performance. Approximate computing (AC) is a nascent computing paradigm that allows us to achieve these objectives by compromising the arithmetic accuracy. Many systems used in domains, like multimedia and big data analysis, exhibit inherent tolerances to a certain level of inaccuracies in computation, and thus can benefit from AC.

Approximate multipliers have been mainly designed using three techniques, Approximation in partial products generation, Approximation in partial product tree, and Approximation in partial products summation. Jiang et al.[7] compared the characteristics of different approximate multipliers, implemented in VHDL based on these different techniques. In the proposed approximate multiplier, a simple tree of the approximate adders is used for partial product accumulation and the error signals are used to compensate errors for obtaining a better accuracy.

Applications such as multimedia, recognition and data mining are inherently error-tolerant and do not require a perfect accuracy in computation. For digital signal processing (DSP) applications, the result is often left to interpretation by human perception. Therefore, strict exactness may not be required and an imprecise result may suffice due to the limitation of human perception. For these applications, approximate circuits may play an important role as a promising alternative for reducing area, power and delay in digital systems that can tolerate some loss of accuracy, thereby achieving better performance in energy efficiency.

In Digital Signal Processing (DSP), which performance improved with loss of accuracy for application of approximate circuits. In existing system, a approximate multiplier with low power and small critical path is deliver for high performance DSP applications. An approximate adder is developed in previous only with Luminance (Luma) concept. AM1 and AM2 are described for error recovery, which error has hand over from approximate adder and then multiplexing. In proposed system, to avoid the error correcting block, by applying bit weight based compression technique we designed a multiplier By using this method we reduce the area of the unsigned multiplier circuits with better accuracies. The performance, hardware

complexity and power consumption associated with multiplier design depend largely on the maximum height of the accumulation tree. The proposed approach decreased the number of vertical product terms in the PPM with the aim of reducing the height of the critical column in the accumulation tree. This can be achieved by following two major steps.

The proposed multiplier organizes the partial product terms using different sizes of significant-driven logic clusters. Each logic cluster targets a group of columns containing two or more bits starting from the least significant bits in successive partial products. Using an array of OR gates in each logic cluster reduces the partial products by half. And reduced set of pre-processed partial products are to be accumulated by applying any convenient scheme of addition. Our proposed approach consists of two major steps. In the first, lossy compression is carried out through logic clustering. The resulting compressed terms are then remapped using their commutative properties.

2. RELATED WORK

A novel approximate multiplier design is referred using a simple, yet fast approximate adder. This newly designed adder can process data in parallel by cutting the carry propagation chain. It has a critical path delay that is even shorter than a conventional one-bit full adder. Albeit with a high error rate, this adder simultaneously computes the sum and generates an error signal; this feature is employed to reduce the error in the final result of the multiplier. In the proposed approximate multiplier, a simple tree of the approximate adders is used for partial product accumulation and the error signals are used to compensate error for obtaining a better accuracy.This multiplier can be configured to two designs by using OR gates and the proposed approximate adders for error reduction, referred to as approximate multiplier 1 (AM1) and approximate multiplier 2 (AM2), respectively. Different levels of error recovery can also be achieved by using a different number of MSBs for error recovery in both AM1 and AM2. Compared to the traditional Wallace tree, the proposed multipliers have significantly shorter critical paths. Functional and circuit simulations are performed to evaluate the performance of the multipliers. Image sharpening and smoothing are considered as approximate multiplication based DSP applications.

Experimental results indicate that the proposed approximate multipliers perform well in these error tolerant image processing applications. The proposed designs can be used as effective library cells for the synthesis of approximate circuits. The nxn approximate multiplier with k-MSB OR-gate based error reduction is referred to as an n/k M1, while an nxn approximate multiplier with k-MSB approximate adder based error reduction is referred to as an n/k AM2. The structures of AM1 and AM2 are shown in Fig. 1

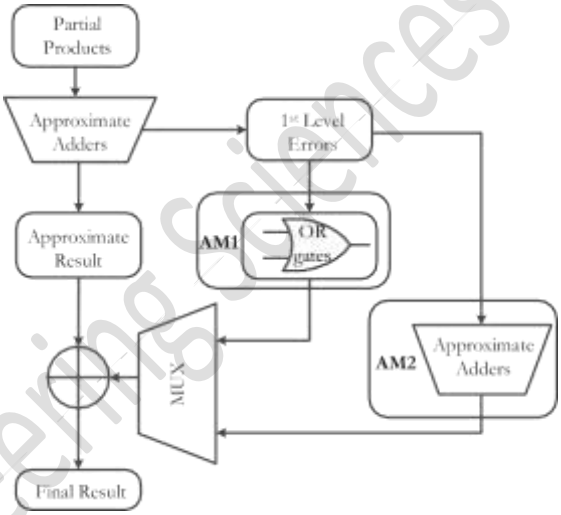


Fig 1 : Block diagram of approximate multiplier

A. THE APPROXIMATE ADDER

The design of a new approximate adder is shown. This adder operates on a set of preprocessed inputs. The input pre-processing (IPP) is based on the commutativity of bits with the same weights in different addends. For example, consider two sets of inputs to a 4-bit adder: i) A = 1010, B = 0101 and ii) A = 1111, B = 0000. Clearly, the additions in i) and ii) produce the same result. In this process, the two input bits Ai Bi = 01 is equal to Ai Bi = 10 (with i being the bit index) due to the commutativity of the corresponding bits in the two operands. The basic rule for the IPP is to switch Ai and Bi if Ai = 0 and Bi = 1 (for any i), while keeping the other combinations (i.e., Ai Bi = 00, 10 and 11) unchanged. By doing so, more 1's are expected in A and more 0's are expected in B. If A˙i B˙ i are the ith bits in the pre-processed inputs, the IPP functions are given by:

$$A_i = A_i + B_i \quad (1)$$

$$B_i = A_i B_i \quad (2)$$

Equations (1) and (2) compute the propagate and generate signals used in a parallel adder such as the carry lookahead adder (CLA). The proposed adder can process data in parallel by reducing the carry propagation chain. Let A and B denote the two input binary operands of an adder, S be the sum result, and E represent the error vector. Ai , Bi , Si and Ei are the ith least significant bits of A, B, S and E, respectively. A carry propagation chain starts at the ith bit when B˙i = 1, A˙i+1 = 1, B˙i+1 = 0. In an accurate adder, Si+1 is 0 and the carry propagates to the higher bit. In the proposed approximate adder, Si+1 is set to 1 and an error signal is generated as Ei+1 = 1. This stops the carry signal from propagating to the higher bits. Hence, a carry signal is produced only by the generate signal, i.e., Ci = 1 only when B˙i = 1, and it only propagates to the next higher bit, i.e., the (i +1)th position.The logical functions are given by

$$Si = B˙i{-}1 + B˙i\, A˙i, \qquad (3)$$

$$Ei = B˙i\, B˙i{-}1\, A˙i\,. \qquad (4)$$

By replacing A˙i and B˙i using (1) and (2) respectively, the logic functions with respect to the original inputs are given by

$$Si = (Ai \oplus Bi) + Ai{-}1Bi{-}1, \qquad (5)$$

$$Ei = (Ai \oplus Bi)\, Ai{-}1Bi{-}1, \qquad (6)$$

where i is the bit index, i.e., i = 0, 1,⋯ , n for an n-bit adder. Let A−1 = B−1 = 0 when i is 0, thus, S0 = A0 ⊕ B0 and E0 = 0. Also, Ei = 0 when Ai−1 or Bi−1 is 0. Consider an n-bit adder, the inputs are given by A = An−1 ⋯ A1A0 and B = Bn−1 ⋯ B1B0, the exact sum is S = Sn−1 ⋯S1S0. Then, Si can be computed as Si + Ei and thus, the exact sum of A and B is given by

$$S = S + E. \qquad (7)$$

In (7) '+' means the addition of two binary numbers rather than the 'OR' function. The error E is always non-negative and the approximate sum is always equal to or smaller than the accurate sum. This is an important feature of this adder because an additional adder can be used to add the error to the approximate sum as a compensation step. While this is intuitive in an adder design, it is a particularly useful feature in a multiplier design as only one additional adder is needed to reduce the error in the final product.

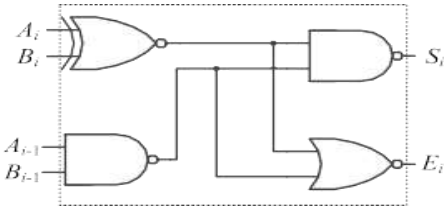


Fig 2 : Approximate Adder Cell

3. PROPOSED WORK

In Digital Signal Processing (DSP), which performance improved with loss of accuracy for application of approximate circuits. In existing system, a approximate multiplier with low power and small critical path is deliver for high performance DSP applications. An approximate adder is developed in previous only with Luminance (Luma) concept. AM1 and AM2 are described for error recovery, which error has hand over from approximate adder and then multiplexing. In proposed system, to avoid the error correcting block, by applying bit weight based compression technique we designed a multiplier By using this method we reduce the area of the unsigned multiplier circuits with better accuracies. The performance, hardware complexity and power consumption associated with multiplier design depend largely on the maximum height of the accumulation tree. The proposed approach decreased the number of vertical product terms in the PPM with the aim of reducing the height of the critical column in the accumulation tree. This can be achieved by following two major steps.

The proposed multiplier organizes the partial product terms using different sizes of significant-driven logic clusters. Each logic cluster targets a group of columns containing two or more bits starting from the least significant bits in successive partial products. Using an array of OR gates in each logic cluster reduces the partial products by half. And reduced set of pre-processed partial products are to be accumulated by applying any convenient scheme of addition

Our proposed approach consists of two major steps. In the first, lossy compression is carried out through logic clustering. The resulting compressed terms are then remapped using their commutative properties. These steps together with the variable compression method, are described below.

3.1 LOGIC COMPRESSION

Parallel multiplication design is generally divided into three consecutive stages: partial product

formation, accumulation, and carry propagation adder. In an (N ×N) multiplier, N^2 AND gates are utilized in parallel to generate the partial product bit-matrix. This matrix is then column-wise accumulated to generate the final product by using carry propagation adders. The proposed approach begins by generating all partial products using the same number of AND gates, similar to conventional multiplication. Before proceeding to the accumulation stage, the number of bits in the partial product matrix is reduced by performing lossy logic compression. The aim is to reduce the number of rows in the partial product matrix, thereby achieving low-complexity hardware before proceeding to accumulation.

3.2 COMMUTATIVE REMAPPING

The logic compression step reduces the number of partial product terms. This reduction can be leveraged to reduce number of rows prior to the accumulation stage. This can be achieved by remapping the partial product terms based on the commutative property of the bits, i.e., bits with the same weight are gathered in the same column. Due to the reduced number of rows, the critical path delay is drastically reduced Figure below demonstrates how the size of the partial product bit matrix in the case of an (8 × 8) multiplier is reduced using cluster mapping and in next stage it is converted to wallace tree structure.

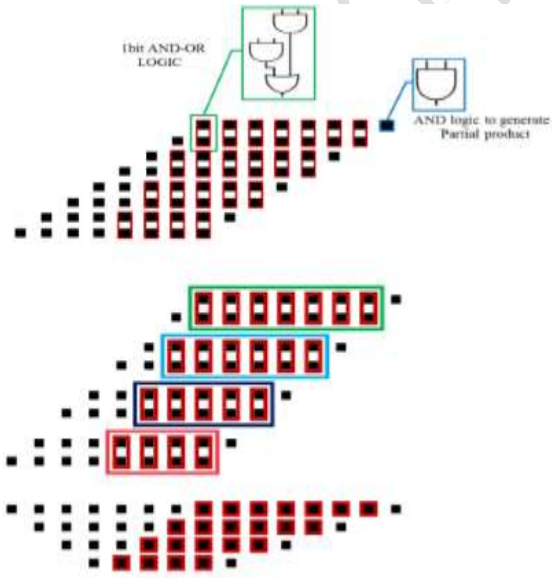


Fig 3: Proposed Multiplier Architecture

A Partial product is generated by using an AND gate these and gate generated outputs are then approximated Using an array of OR gates by half. A reduced set of pre processed partial product matrix is thus ready to be accumulated by applying any convenient scheme of multiplication, such as carry-save array, Wallace and Dadda tree. In theory, a two-input OR gate is sufficient to sum up two bits, i.e., ‘0’+‘1’=‘1’+‘0’=‘1’ ‘0’OR‘1’=‘1’OR‘0’=‘1’ and also ‘0’+‘0’= ‘0’OR‘0’=‘0’. However, the OR gate fails to give an accurate sum if the two inputs are high, i.e., ‘1’+‘1’=‘1’OR‘1’, the adder returns ‘10’ and OR outputs ‘1’.

In the Partial product accumulation stage we can add the partial products by a Dual quality adder thus we can have a choice to select the output with 2 combinations

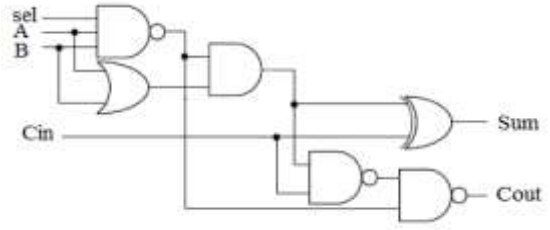
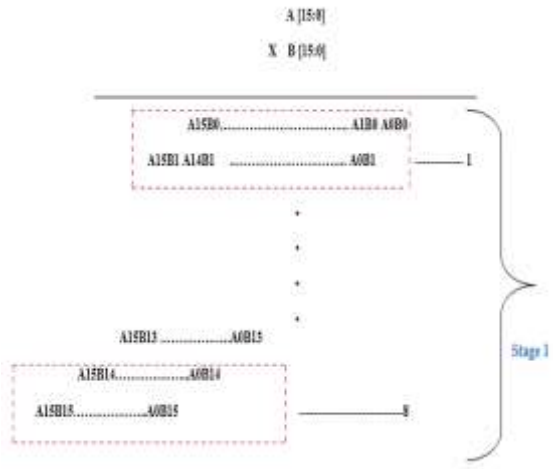


Fig 4: Dual Quality Adder cell

To Reduce the Delay of the Existing Circuitry we implemented another method where we have less area and delay compared to existing technique of approximation. And is done in two stages called partial product generation with Reduction and Final summation. The partial products generated are reduced by OR gate logic Clustering and in the second stage the reduced partial products are to be added in hybrid logic where right or least significant half is fully approximated part and the left part is added with accurate adder thus we can have better accuracy of the output.



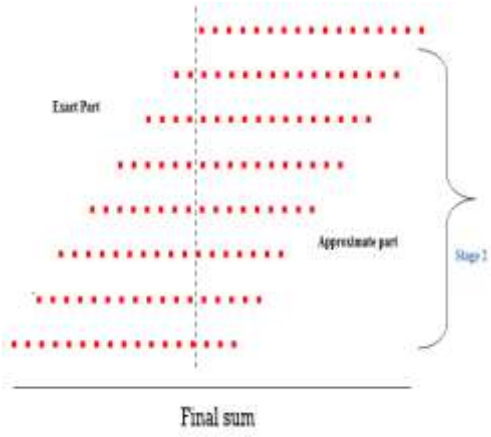


Fig5: Proposed Approximate Algorithm

The final sum is calculated using parallel prefix adder and in that we induce kogge stone adder in carry generation stage . In this way we reduce the area and delay of the overall multiplier.

4. SIMULATION RESULTS

The total designs are done and implemented in Xilinx ISE 14.7 with Verilog HDL coding

The hybrid logic used in the proposed multiplier method where the partial product accumulation is done using kogge-stone adder in the second stage of parallel prefix adder ,which helps in reducing the delay as we operate parallel.

Table I: Comparison of Existing and Proposed Multipliers

Parameters	Existing Method	Proposed Method
Area (LUT's)	571	226
Delay(ns)	20.493	16.534
Accuracy(%)	69.91	96.87

Where as in the existing Approximate multiplier the generation and reduction of error uses individual adders which increases delay as well as area.In the existing approximate multiplier in order to obtain the area and delay the accuracy is sacrificed.The Approximate multiplier when simulated for 16 bits it exhibits the accuracy comparably low,so in order to obtain better accuracy in proposed system, we avoid the error correcting block, by applying bit weight based compression technique we designed a multiplier as shown in fig (3) By using this method we reduce the area of the unsigned multiplier circuits with better accuracies or nearly accurate results.

5. APPLICATION

To showcase the advantage of this concept we consider an image processing application like Foreground-background separation is a method, where the goal is to separate the image into foreground and background. In semi-interactive settings, the user marks some pixels as "foreground", a few others as "background", and it's up to the algorithm to classify the rest of the pixels. Here, we are using 16-bit multiplier for foreground separation based on their threshold value which is related to that gray image. The results for the image processing can be done through Xilinx system generator and results are shown in below figures.
The PSNR value of the image multiplication using the proposed method is obtained as 33.98.

(a)

(b)

Fig. 6 :(a) Input image (b) Output image

6. CONCLUSION

This paper proposes high-performance and area efficient partial product accumulation architectures for a multiplier using AND-OR gate and approximate adder based error reduction schemes are utilized, An approximate 16 × 16 multiplier with two partial product accumulations is compared with the proposed multipliers. The proposed approximate multipliers have been shown to have a lower area than an exact Wallace multiplier and existing approximate designs. The proposed approximate multiplier 1 had improved over previous approximate designs especially in area and accuracy and the proposed approximate multiplier 2 had improved over previous approximate designs especially in area and delay. While previous design focus on reducing both delay and power with often unsatisfying delay and accuracy, the proposed designs achieve excellent area reductions with a high accuracy where we are using approximate weight based logic compression using OR logic which is having better accuracy with reduced area and having better accuracy.

REFERENCES:

[1] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in Proc. 18th IEEE Eur. Test Symp., May 2013, pp. 1–6.

[2] S.-L. Lu, "Speeding up processing with approximation circuits," Computer, vol. 37, no. 3, pp. 67–73, Mar. 2004.

[3] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in Proc. Design, Automat. Test Eur., Mar. 2008, pp. 1250–1255.

[4] N. Zhu, W. L. Goh, and K. S. Yeo, "An enhanced low-power high-speed adder for error-tolerant application," in Proc. 12th Int. Symp. Integr. Circuits, Dec. 2009, pp. 69–72.

[5] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[6] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "IMPACT: IM Precise adders for low-power approximate computing," in Proc. IEEE/ACM Int. Symp. Low Power Electron. Design, Aug. 2011, pp. 409–414.

[7] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate N arithmetic designs," in Proc. Design Automat. Conf., Jun. 2012, pp. 820–825.

[8] K. Du, P. Varman, and K. Mohanram, "High performance reliable variable latency carry select addition," in Proc. Design, Automat. Test Eur. Conf. Exhib., Mar. 2012, pp. 1257–1262.

[9] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," IEEE Trans. Comput., vol. 62, no. 9, pp. 1760–1771, Jun. 2012.

[10] J. Huang, J. Lach, and G. Robins, "A methodology for energy-quality tradeoff using imprecise hardware," in Proc. Design Automat. Conf., Jun. 2012, pp. 504–509.

[11] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in Proc. Design, Automat. Test Eur. Conf. Exhib., Mar. 2014, pp. 1–4.

[12] B. Parhami, Computer Arithmetic. London, U.K.: Oxford Univ. Press, 2000.

[13] M. A. Breuer, "Intelligible test techniques to support error-tolerance," in Proc. 13th Asian Test Symp., Nov. 2004, pp. 386–393.

[14] N. Weste and H. David, CMOS VLSI Design: A Circuits and Systems Perspective, 3rd ed. London, U.K.: Pearson, 2005.

[15] V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," IEEE Trans. Comput., vol. 45, no. 3, pp. 294–306, Mar. 1996.

[28] K. C. Bickerstaff, E. E. Swartzlander, and M. J. Schulte, "Analysis of column compression multipliers," in Proc. 15th IEEE Symp. Comput. Arithmetic, Jun. 2001, pp. 33–39.

[16] K. C. Bickerstaff, M. J. Schulte, and E. E. Swartzlander, Jr., "Parallel reduced area multipliers," J. VLSI Signal Process. Syst. Signal, Image Video Technol., vol. 9, no. 3, pp. 181–191, 1995.

[17] Y.-K. Cheng, C.-H. Tsai, C.-C. Teng, and S.-M. Kang, Electrothermal Analysis of VLSI Systems. New York, NY, USA: Springer, 2002.

[18] E. J. King and E. E. Swartzlander, "Data-dependent truncation scheme for parallel multipliers," in Proc. 31st Conf. Rec. Asilomar Conf. Signals,
Syst. Comput., vol. 2, Nov. 1997, pp. 1178–1182.

[19] M. S. K. Lau, K.-V. Ling, and Y.-C. Chu, "Energy-aware probabilistic multiplier: Design and analysis," in Proc. Int. Conf. Compil., Archit., Synthesis Embedded Syst., 2009, pp. 281–290.
[20] H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.