International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

Approximate Multipliers with Configurable Error Recovery

Dr.T.Kowsalya M.E.,Ph.D[1]., Venkat sai kumar E[2], Nikhil Kumar G[3], Reddy Prasad K[4]

Assistant Professor, Department of Electronics and Communication Engineering, Muthayammal Engineering College, Tamilnadu, India[1]

UG Scholar, Department of Electronics and Communication Engineering, Muthayammal Engineering College, Tamilnadu, India[2]

UG Scholar, Department of Electronics and Communication Engineering, Muthayammal Engineering College, Tamilnadu, India[3]

UG Scholar, Department of Electronics and Communication Engineering, Muthayammal Engineering College, Tamilnadu, India[4]

ABSTRACT: Approximate circuits have been considered for applications that can tolerate some loss of accuracy with improved performance and/or energy efficiency. Multipliers are key arithmetic circuits in many of these applications including digital signal processing (DSP). In this paper, a novel approximate multiplier with low power consumption and a short critical path is proposed for high performance DSP applications. This multiplier leverages a newly designed approximate adder that limits its carry propagation to the nearest neighbors for fast partial product accumulation. Different levels of accuracy can be achieved by using either OR gates or the proposed approximate adder in a configurable error recovery circuit. The approximate multipliers using these two error reduction strategies are referred to as AM1 and AM2, respectively. Both AM1 and AM2 have a low mean error distance. Compared with a Wallace multiplier optimized for speed, an 8×8 AM1 using four most significant bits for error reduction shows a 60% reduction in delay (when optimized for delay) and a 42% reduction in power dissipation (when optimized for area). In a 16×16 design, half of the least significant partial products are truncated for AM1 and AM2.

KEYWORDS: Approximate Circuits, Digital Signal Processing (DSP), Error Reduction (AM1& AM2)

I. INTRODUCTION

Approximate computing has emerged as a potential solution for the design of energy-efficient digital systems. 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 play an important role as a promising alternative for reducing area, power and delay, thereby achieving better performance in energy efficiency.

As one of the key components in arithmetic circuits, adders have been extensively studied for approximate implementation. As the typical carry propagation chain is usually shorter than the width of an adder, the speculative adders use a reduced number of less significant input bits to calculate the sum bits [. An error detection and recovery scheme has been proposed in to extend the scheme for a reliable adder with variable latency. A reliable variable-latency adder based on carry select addition has been presented. As a number of approximate adders have been proposed, new methodologies to model, analyze and evaluate them have been discussed.

A multiplier usually consists of three stages: partial product generation, partial product accumulation and a carry propagation adder (CPA) at the final stage. In the under designed multiplier (UDM), approximate partial products are computed using inaccurate 2 × 2 multiplier blocks, while accurate adders are used in an adder tree to 2 accumulate the approximate partial products, approximate 4 × 4 and 8 × 8 bit Wallace multipliers are designed by using a carry-in prediction method. Then, they are used in the design of approximate 16 × 16 Wallace multipliers,

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

referred to as AWTM. The AWTM is configured into four different modes by using a different number of approximate 4 × 4 and 8 × 8 multipliers. The use of approximate speculative adders has been discussed in for the final stage addition in a multiplier. The error tolerant multiplier (ETM) of [16] is based on the partition of a multiplier into an accurate multiplication part for most significant bits and a non-multiplication part for least significant bits.

The static segment multiplier (SSM) utilizes a similar partition scheme. In an n ×n SSM, an m ×m accurate multiplier(m _ n/2) is used to multiply the m consecutive bits from the two input operands. Whether the (n− m) MSBs of each input operand are all zero determines the selection of the inputs for the accurate multiplier. These approximate multipliers are designed for unsigned operation. Signed multiplication is usually implemented by using a Booth algorithm. Approximate designs have been proposed for fixed width Booth multipliers.

Discuss about Vedic multipliers based on Vedic Mathematics. Vedic multipliers are presently under focus due to their high speed and low power consumption. They propose a design of 8 and 16bit multipliers using fast adders (carry save adder, Brent- Kung adder and carry- select adder) to minimize the power delay product of multipliers intended for high-performance and low-power applications. Implementation results demonstrate that the proposed Vedic multipliers with fast adders really achieve significant improvement in delay, and power-delay product when compared with the conventional multipliers.

The proposed different signed 16×16 bit approximate radix- 8 Booth multiplier designs. Initially, an approximate 2-bit adder consisting of a 3-input XOR gate is proposed to calculate the triple of binary numbers. The error detection, compensation and recovery circuits of the approximate 2-bit adder are also presented. The 2-bit adder is then employed to implement the lower part of an approximate encoding adder for generating a triple multiplicand without carry propagation; it overcomes the issue commonly found in a radix-8 scheme. In the proposed signed approximate radix-8 Booth multipliers, referred to as ABM1 and ABM2, a truncation technique is employed to further save power and time. The parallel processing by a Wallace tree is then employed to speed up the addition of partial products. The simulation results show that the proposed approximate recoding adders (ARA8, ARA8-2C and ARA8-2R) are more suitable (in terms of hardware efficiency and accuracy) for a radix-8 Booth multiplier than other approximate adders.

II. EXISTING SYSTEM

Generally, a multiplier consists of stages of partial product generation, accumulation and final addition. The commonly used partial product accumulation structures include the Wallace, Dadda trees and a carry-save adder array. In a Wallace tree, log2(n) layers are required for an n-bit multiplier. The adders in each layer operate in parallel without carry propagation, and the same operation repeats until two rows of partial products remain. Therefore, the delay of the partial product accumulation stage is O(log2(n)). Moreover, the adders in a Wallace tree can be considered as a 3:2 compressor and can be replaced by other counters or compressors (e.g. a 4:2 compressor) to further reduce the delay. The Dadda tree has a similar structure as the Wallace tree, but it uses as few adders as possible. For a carry-save adder array, the carry and sum signals generated by the adders in a row are connected to the adders in the next row. Adders in a column operate in series. Hence the partial product accumulation delay of an n-bit multiplier is approximately O(n), longer than that of the Wallace tree. However, an array requires a smaller area and thus a lower power dissipation due to the simple and symmetric structure. Three methodologies are applicable to approximate a multiplier: i) approximation in generating the partial products, ii) approximation (including truncation) in the partial product tree and iii) using approximate designs of adders, counters and/or compressors to accumulate the partial products. Following this classification, existing designs of approximate multipliers are briefly reviewed next.

III. SYSTEM IMPLEMENTATION

Approximation in generating partial products

The under designed multiplier (UDM) utilizes an approximate 2 × bit multiplier block obtained by altering a single entry in the Karnaugh Map (KMap) of its function. In this approximation, the accurate result “1001” for the multiplication of “11” and “11” is simplified to “111” to save one output bit. Assuming the value of each input bit is equally likely, the error rate of the 2 × 2 bit multiplier block is (1 2) 4 = 1 16 . Larger multipliers can be designed based on the 2 × 2 bit multiplier. This multiplier introduces an error when generating partial products, however the adder tree remains accurate.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

Approximation in the partial product tree

A bio-inspired imprecise multiplier referred to as a broken array multiplier (BAM) is proposed. The BAM operates by omitting some carry- save adders in an array multiplier in both horizontal and vertical directions. The error tolerant multiplier (ETM) is divided into a multiplication section for the MSBs and a non-multiplication section for the LSBs. A NOR gate based control block is used to deal with two cases: i) if the product of the MSBs is zero, then the multiplication section is activated to multiply the LSBs without any approximation, and ii) if the product of the MSBs is nonzero, the nonmultiplication section is used as an approximate multiplier to process the LSBs, while the multiplication section is activated to multiply the MSBs. The static segment multiplier (SSM) [10] was further proposed using a similar partition scheme. Different from ETM, no approximation is applied to the LSBs in the SSM.

Either the MSBs or the LSBs of each of the operands is accurately multiplied depending on whether its MSBs are all zeros. has shown that a small improvement in accuracy and hardware cost is achieved compared to the ETM, thus this design is not considered further in the comparison study. A power and area-efficient approximate Wallace tree multiplier (AWTM) is based on a bitwidth aware approximate multiplication and a carry-in prediction method. An n bit AWTM is implemented by four n/2-bit sub- multipliers, and the most significant n/2-bit sub-multiplier is further implemented by four n/4-bit sub multipliers. The AWTM is configured into four different modes by the number of approximate n/4-bit sub-multipliers in the most significant n/2-bit sub multiplier. The approximate partial products are then accumulated by a Wallace tree.

Using approximate counters or compressors

In the inaccurate counter based multiplier (ICM), an approximate (4:2) counter is proposed for an inaccurate 4-bit Wallace multiplier [7]. The carry and sum of the counter are approximated as “10” (for “100”) when all input signals are ‘1’. As the probability of obtaining a partial product of ‘1’ is 1 4 , the error rate of the approximate (4:2) counter is (1 4) 4 = 1 256 . The inaccurate 4bit multiplier is then used to construct larger multipliers with error detection and correction circuits. In the compressor based multiplier, accurate (3:2) and (4:2) compressors are improved to speed up the partial product accumulation stage [12]. By using the improved compressors, better energy and delay characteristics are obtained for a multiplier.

IV. PROPOSED SYSTEM

The architecture of the proposed approximate multiplier is shown in below In the proposed design, the simplification of the partial product

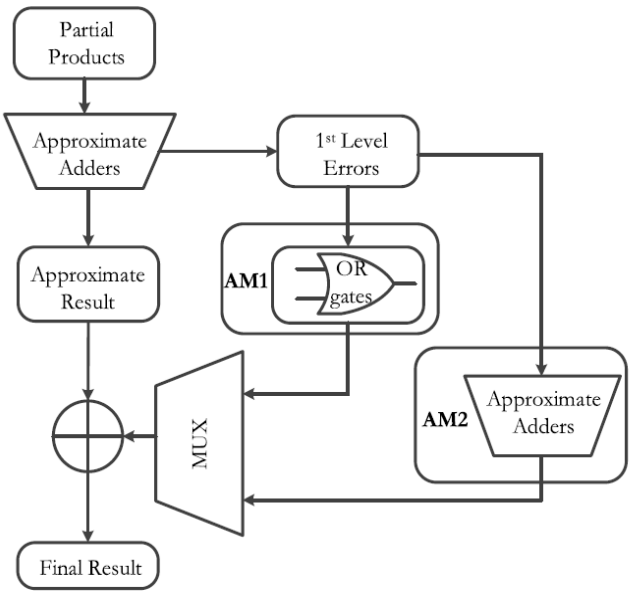

Fig 1 Proposed System Block Diagram

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

Basic Simulation Flow

The following diagram shows the basic steps for simulating a design in modelsim.

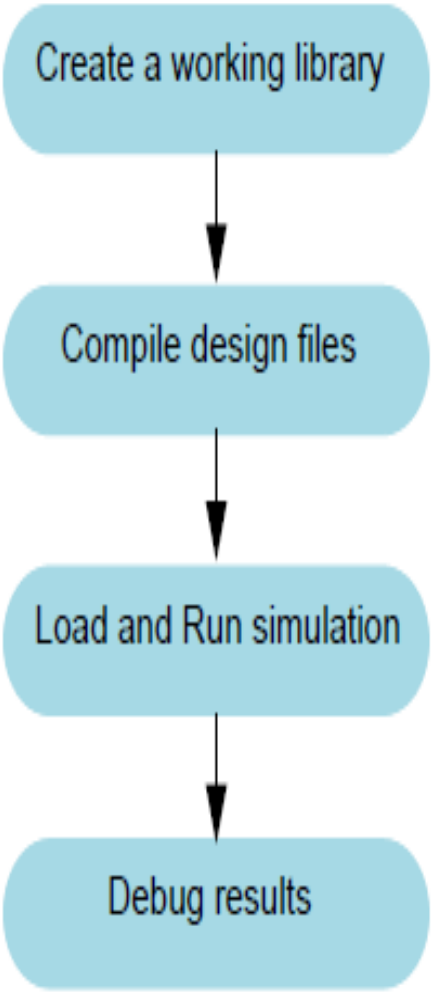

Fig :2 Basic Simulation Flow

- Creating the Working Library. In Modelsim, all designs are compiled into a library. You typically start a new simulation in Modelsim by creating a working library called "work," which is the default library name used by the compiler as the default destination for compiled design units.
- Compiling Your Design. After creating the working library, you compile your design units into it. The ModelSim library format is compatible across all supported platforms. You can simulate your design on any platform without having to recompile your design.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

Project Flow

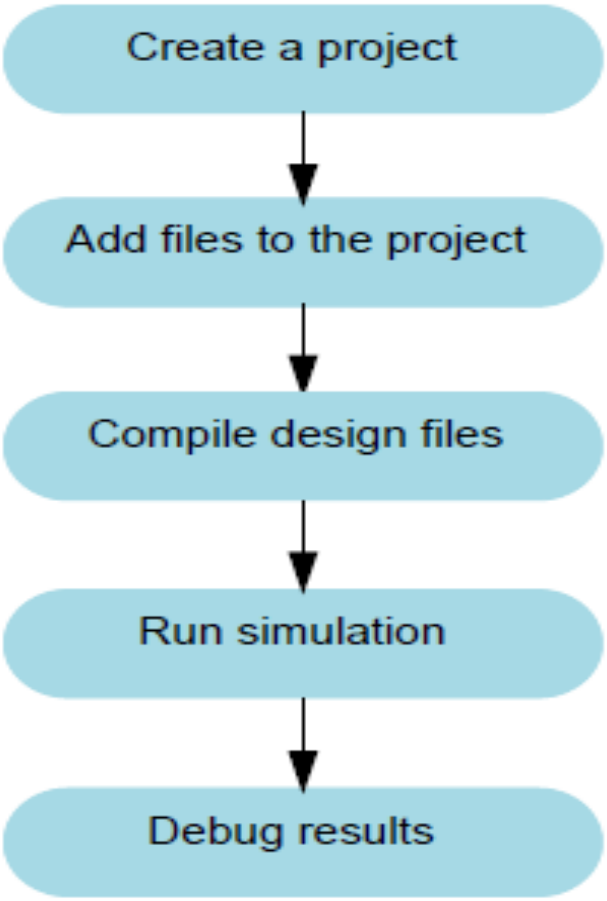

Fig 3 Project Flow

A project is a collection mechanism for an HDL design under specification or test. Even though you don't have to use projects in ModelSim.

Multiple Library Flow

The diagram above shows the basic steps for simulating with multiple libraries. A resource library is typically static and serves as a parts source for your design. You can create your own resource libraries, or they may be supplied by another design team or a third party (e.g., a silicon vendor). You specify which resource libraries will be used when the design is compiled, and there are rules to specify in which order they are searched.

A common example of using both a working library and a resource library is one where your gate-level design and test bench are compiled into the working library, and the design references gate-level models in a separate resource library.You can also link to resource libraries from within a project. If you are using a project, you would replace the first step above with these two steps: create the project and add the test bench to the project.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

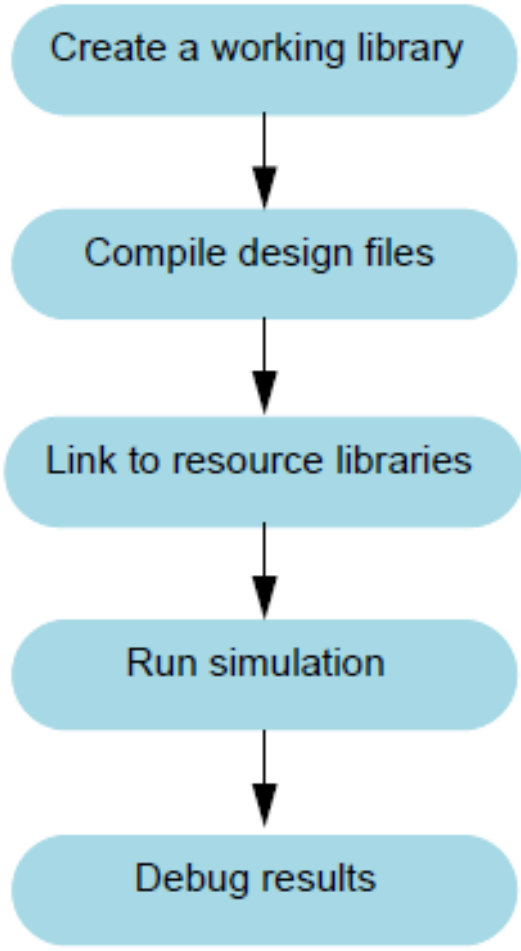

Fig 4 Multiple Library Flow

DOMAIN-SPECIFIC PLATFORM

The next layer in the targeted design platform hierarchy is the domain- specific platform. Released from three to six months after the base platform, each domain specific platform targets one of the three primary Xilinx FPGA user profiles (domains):the embedded processing developer, the digital signal processing (DSP) developer, or the logic/connectivity developer. This is where the real power and intent of the targeted design platform begins to emerge. Domain-specific platforms augment the base platform with a predictable, reliable, and intelligently targeted set of integrated technologies, including:

- Higher-level design methodologies and tools
- Domain-specific embedded, DSP, and connectivity IP
- Domain-specific development hardware and daughter cards
- Reference designs optimized for embedded processing, connectivity, and DSP
- Operating systems (required for embedded processing) and software.

A market-specific platform is an integrated combination of technologies that enables software or hardware developers to quickly build and then run their specific application or solution. Built for use in specific markets such as Automotive, Consumer, Mil/Aero, Communications, AVB, or ISM, market-specific platforms integrate both the base and domain- specific platforms and provide higher level elements that can be leveraged by customer-specific software and hardware designs. The market-specific platform can rely more heavily on third-party targeted IP than the base or domain-specific platforms.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

V. RESULTS AND DISCUSSIONS

A distinguishing feature of the proposed approximate multiplier is the simplicity to use approximate adders in the partial product accumulation. It has been shown that this may lead to low accuracy, because errors may accumulate and it is difficult to correct errors using existing approximate adders. However, the use of the newly proposed approximate adder overcomes this problem by utilizing the error signal. The resulting design has a critical path delay that is shorter than a conventional one-bit full adder, because the new n-bit adder can process data in parallel. The approximate adder has a rather high error rate, but the feature of generating both the sum and error signals at the same time reduces errors in the final product. An adder tree is utilized for partial product accumulation; the error signals in the tree are then used to compensate the error in the output to generate a product with a better accuracy.

The architecture of the proposed approximate multiplier is shown in Fig. 1. In the proposed design, the simplification of the partial product accumulation stage is accomplished by using an adder tree, in which the number of partial products is reduced by a factor of 2 at each stage of the tree. This adder tree is usually not implemented using accurate multi-bit adders due to the long latency. However, the proposed approximate adder is suitable for implementing an adder tree, because it is less complex than a conventional adder and has a much shorter critical path delay

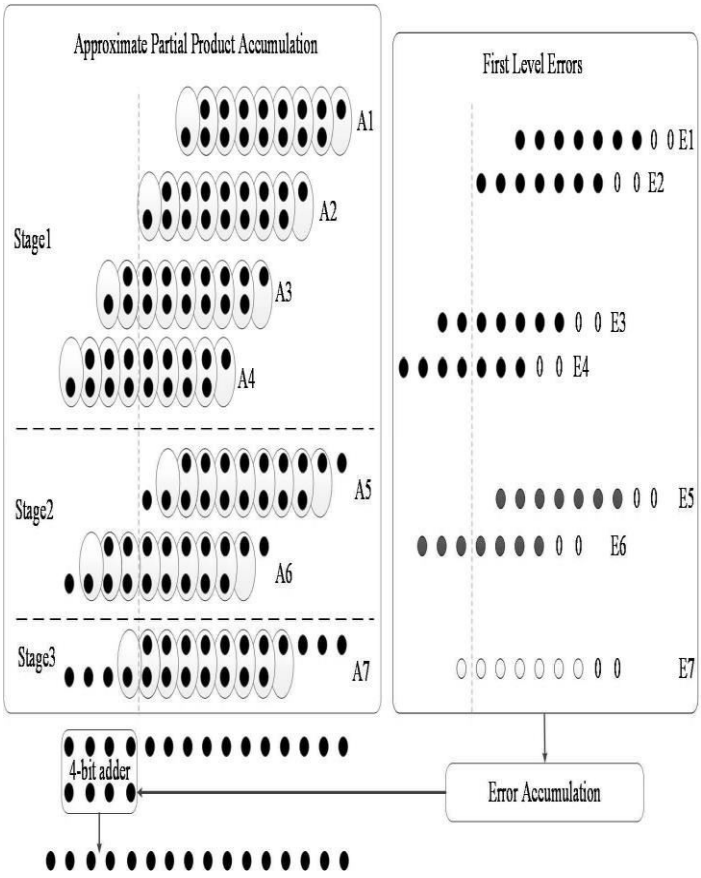

Fig 5 An approximate multiplier with partial error recovery

ERROR REDUCTION

The approximate adder generates two signals: the approximate sum S and the error E; the use of the error signal is considered next to reduce the inaccuracy of the multiplier. As (7) is applicable to the sum of every single approximate adder in the tree, an error reduction circuit is applied to the final multiplication result rather than to the output of each adder. Two steps are required to reduce errors: i) error accumulation and ii) error recovery by the addition of the accumulated errors to the adder tree output using a CPA. In the error accumulation step, error signals are accumulated to a single error vector, which is then added to the output vector of the partial product accumulation tree. Two approximate error accumulation methods are proposed, yielding the approximate multiplier 1 (AM1) and

approximate multiplier 2 (AM2). Fig. 2 shows the symbols for an OR gate, a full adder and half adder cell and an approximate adder cell used in the error accumulation tree.

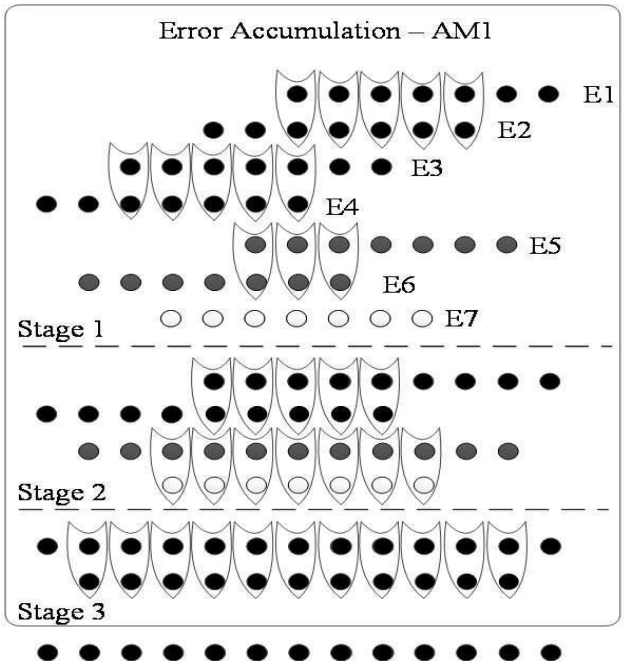

Fig 6 Error accumulation tree for AM1

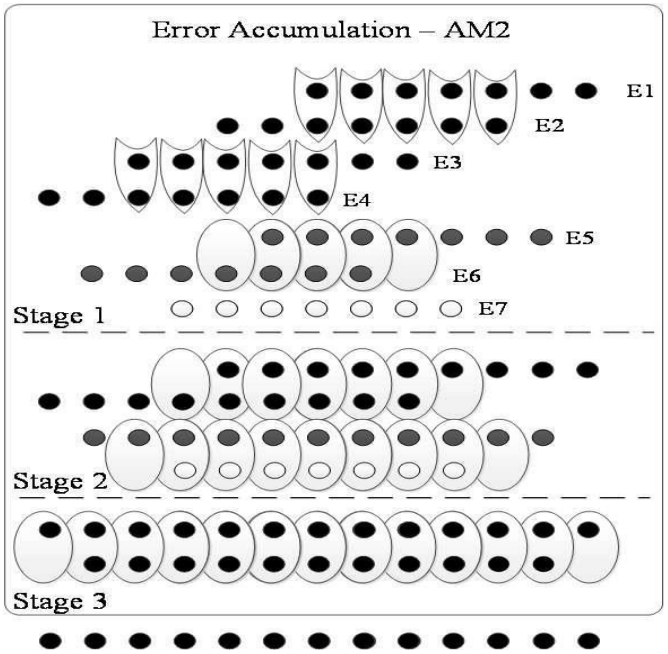

Fig 7 Error accumulation tree for AM2

VI. CONCLUSION

This proposes a high-performance and low-power approximate partial product accumulation tree for a multiplier using a newly designed approximate adder. The proposed approximate adder ignores the carry propagation by generating both an approximate sum and an error signal. OR gate and approximate adder based error reduction schemes are utilized, yielding two different approximate 8×8 multiplier designs: AM1 and AM2. Moreover, medications are made on the error reduction schemes for 16×16 multiplier designs, such that TAM1 and TAM2 are obtained by truncating 16 LSBs of the partial products. The proposed approximate multipliers have been shown to have a lower power dissipation than an exact Wallace multiplier optimized for speed. Functional analysis has shown that on a statistical basis, the proposed multipliers have very small error distances and thus, they achieve a high accuracy. Simulation has also shown that AM2 has a higher accuracy than AM1 at the cost of a longer delay and a higher power consumption. Truncation-based designs (TAM1 and TAM2) achieve a significant improvement in power and area with a small degradation in NMED. The proposed approximate multipliers improve over previous approximate designs especially in accuracy. While previous designs focus on reducing both delay and power with often unsatisfying accuracy, the proposed designs achieve excellent delay and power reductions with a high accuracy. The application of the proposed multipliers to image sharpening and smoothing has shown that the proposed designs are very competitive.

REFERENCES

[1] 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.
[2] 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.
[3] A. Kahng and S. Kang, “Accuracy-configurable adder for approximate arithmetic designs,” in Proc. Design Automat. Conf., Jun. 2012, pp. 820–825.
[4] 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.
[5] 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.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

||Volume 9, Issue 6, June 2020||

[6] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "IMPACT: IMPrecise 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 approxi- mate 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] J. Miao, K. He, A. Gerstlauer, and M. Orshansky, "Modeling and synthe- sis of quality-energy optimal approximate adders," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design, Nov. 2012, pp. 728–735.
[12] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.
[13] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classifi- cation, and comparative evaluation of approximate arithmetic circuits," ACM J. Emerg. Technol. Comput. Syst., vol. 13, no. 4, 2017, Art. no. 60.
[14] P. Kulkarni, P. Gupta, and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture," J. Low Power Electron., vol. 7, no. 4, pp. 490–501, 2011.
[15] K. Bhardwaj, P. S. Mane, and J. Henkel, "Power- and area-efficient approximate wallace tree multiplier for error-resilient systems," in Proc.15th Int. Symp. Qual. Electron. Design, Mar. 2014, pp. 263–269.