APPROXIMATE MULTIPLIER DESIGN USING NOVEL 4: 2 COMPRESSORS

D. SWETHA [1], B. NARENDRA [2].

[1] PG STUDENTS, ECE Dept. ECE DEPT. MJR COLLEGE OF ENGINEERING, PILER, ANDHRA PRADESH, INDIA. Email: devaganiswetha7@gmail.com.

[2] ASSISTANT PROFESSOR, ECE DEPT. MJR COLLEGE OF ENGINEERING, PILER, ANDHRA PRADESH, INDIA. Email: boggulanarendra@gmail.com.

Abstract: In this paper, we are going to design pipelined based approximate multiplier by applying pipelining concept to the proposed multiplier and following the previous implementations such as the proposed approximate compressor with high accuracy compared to existing approximate compressors. Multipliers consume a lot of power, as they perform mathematical operations repeatedly in the most applications. An approximate multiplier is a new way for reducing critical path delay in error-tolerant applications. The primary requirement is high-speed computations, which establish a trade-off between area, power, and accuracy with computational delay. We use pipelining to implement the suggested approximate multiplier, as well as an approximate 4-2 compressor with high accuracy and an adjustable approximate multiplier that may dynamically truncate partial products to meet different accuracy needs. In addition, we offer a simple error compensation circuit for reducing error distance. The proposed approximate multiplier can adjust the accuracy and power required for multiplications at run-time based on the users' requirement. Experimental results show that the compared to existing accurate Wallace multiplier, the proposed adjustable approximate multiplier can be reduced in parameter values. And compared to pipelined implementation and non-pipelined approximate multiplier design performance will be improved. The synthesis and simulation of the proposed designs can be implemented using Xilinx Vivado 2018.2.

Keywords: Approximate computing, pipelining concept, approximate multiplier, deep learning, high precision, reconfigurable approximate design

INTRODUCTION

Multipliers are one of the most important computational blocks, and are frequently used in DSPs. Growth in multipliers like these can be seen in fields like computer graphics, science calculation,

and image processing. Multiplier speed affects how quickly processors are running and designers emphasise high-speed over low energy consumption. The multiplier architecture is a partial production, reduction, and addition stage. Most of the time, power, and range of multiplication are assigned to the partial product reduction stage. Compressors normally implement this stage because they help reduce partial products, as well as reduce the critical path that is important for maintaining the circuit's performance.

The structure 3-2, 4-2, 5-2 is used to accomplish this. Full adder cell: also known as a 3-2 compressor circuit. Improving the design of these compressors will lead to improved system performance as a whole. The internal compressor structure includes XOR-XNOR and multiplexers. Arithmetic systems, multipliers, compressors, parity control systems, and other circuits all use XOR-XNOR circuits. Optimized XOR-XNOR gates improve the multiplier circuit performance. The work here proposes a new XOR-XNOR module and uses it to implement a 4-2 compressor. Reduced transistor and power consumption, providing partial product accumulation.

As an approximation for full adder cells, full adder calculation has been extensively studied. Liang et al. compared these adders and provided several new measures for the assessment of approximate and probabilistic supplements for unified figures. In a circuit, the error distance (ED) is the arithmetic distance between the wrong and the right output. However, approximation multipliers have not received enough attention. The use of approximate additional products in the design of a multiplier is not viable, as it is hard, complicated, and wasteful with regard to performance metrics. Several applications received approximate multipliers. These designs use a truncated method of multiplication, estimating the constant sized columns of partial products. This neural network error happens when partials are multiplied using an improper array multiplier (and thus removing some adders in the array). a truncated multiplier with a correction constant

LITERATURE SURVEY

Previous approximate multiplier work is reviewed in this section. Broken-array multiplier (BAM) was used to propose two approximate multipliers. Approximate signed Booth multiplier was calculated using the BAM approximation method. The approximate multiplier results in savings ranging from 28% to 58.6% in power consumption and reductions in area ranging from 19.7% to 41.8% for different word lengths. a 2×2 set of inaccurate building blocks saved the power by approximately 31.8% to 45.4% An approximate 32-bit signed multiplier design was created for

pipelined processors. Full adder-based tree multipliers gave a 20% speed increase while having a 14% probability of error.

In a correctable multiplier, which correctly divided the multiplication, an error-tolerant multiplier was created. This reports bit width accuracy. A 50% power savings was reported in the 12-bit multiplier case. Using approximate multipliers instead of exact multipliers in image processing applications has been widely discussed in the literature. a resilience-enhancing architecture (ACRE) included an accurate-configurable multiplier (ACM). ACMA used carry-in prediction, which involved pre-computation logic, to increase the throughput. using the proposed approximate multiplication, the critical path length was reduced by around 50% Bhardwaj et al created an approximate Wallace tree multiplier (AWTM). It relied on carry-in predictions to shave time off the critical path. AWTM was used in a real-time benchmark image application where power and area reductions of about 40% and 30% were seen, with no impact on image quality.

EXISTING SYSTEM

VEDIC MULTIPLIER:

A 2X2, 4X4, and 8X8 Vedic bit multiplier model is shown below. In this passage, the sutra "Urdhva-Tiryagbhyam" (Vertically and crosswise) is used to describe the design of two-digit multiplication in terms of architecture. Partial product generation and additions can all be done concurrently with Vedic Multiplier. In this way, it is particularly well-suited to parallel processing. Binary multiplications will be facilitated by this feature. This is because it shortens delay, the prime motivation for this project.

VEDIC MULTIPLIER FOR 8x8 BIT MODULE:

In the previous section, we talked about implementing the 8x8 bit Vedic multiplier module using four 4x4 bit Vedic multiplier modules. Let's investigate eight-by-eight multiplications, like this: let A=A7A6A5A4A3A2A1A0 and B=B7B6B5B4B3B2B1B0. 16-bit output: S15, S14, S13, S12, S11, S10, S9, S8, S7, S6, S5, S4, S3, S2, S1, and S0 Let's decompose the 8-bit multiplier A into a pair of 4-bit AL-AH. Additionally, BH-BL is the total of B and BL. Vedic multiplication is the basis of the 16-bit product. If you use 4-bit multiplier blocks, you can conduct the multiplication. To obtain the final product, the outputs of 4x4 bit multiplication are added. The figure to the right shows how many 8-bit Ripple Carry Adder chips are required to implement this circuit.

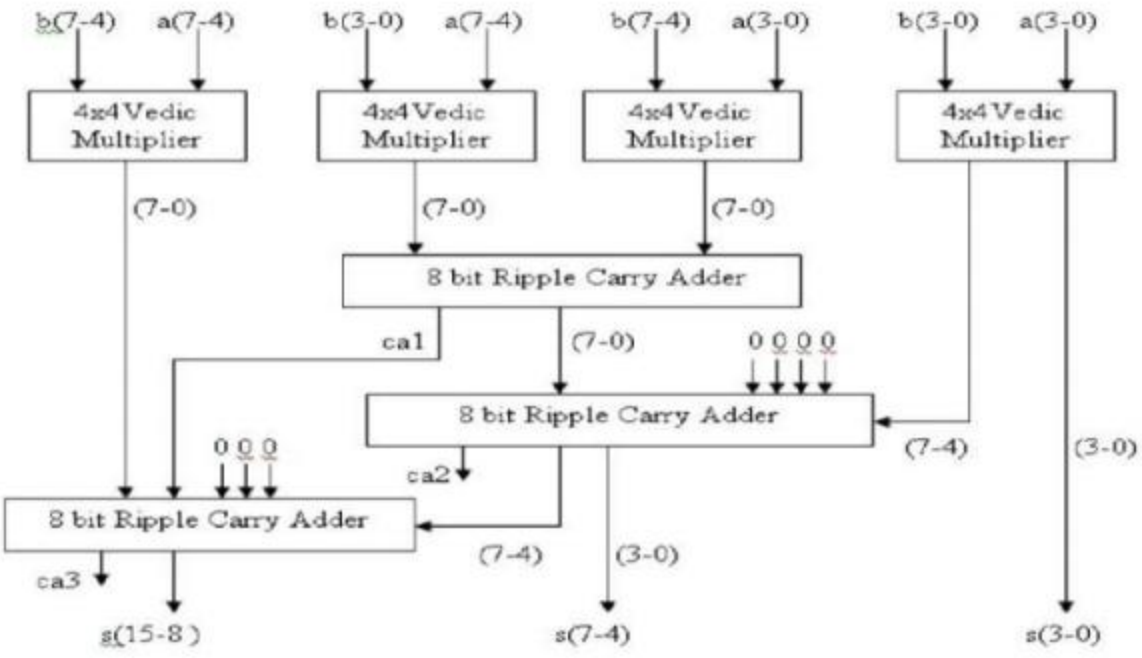

Fig:1 Block Diagram of 8x8 bit Vedic Multiplier

The process described above may be applied to any number of bits in input. Multiplication of two N-bit binary numbers must be done in the form of 2N. After we multiply N and N, the final result will be of (N + N) bits, because S = S(N + N). Step 1: Multiply the multiplicand A and the multiplier B, each consisting of a total of [N to (N/2)+1] bits and [N/2 to 1] bits, and place the results into two equal halves, where the first portion represents the MSB and the other represents the LSB. Step 2: Show how the different parts of A are represented as AM and AL, and the different parts of B are shown as BM and BL. Draw AM AL and BM BL as AM AL and BM BL. Step 3: In the format of Fig. 2, for A X B, we have

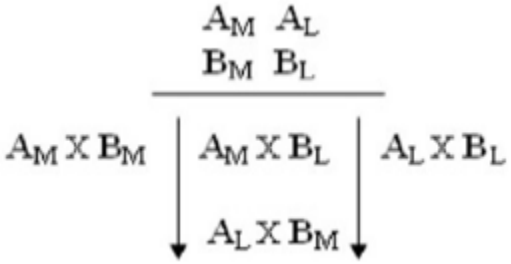

Fig. 2 General Representation for Vedic multiplication.

We would like to offer a novel and alternative multiplier for error robust DSP applications here. To make the standard multiplication methodology more accurate, this proposed approximate multiplier, which is also area efficient, is constructed. We can compare the structures' delays, power, and energy consumptions, as well as their power-delay products (PDPs) and areas, and select the ones with the biggest divergence from an approximate and accurate (exact) multiplier. This paper's contributions can be described as follows: Planning a new hardware architecture for the proposed approximate multiplication method for signs and unsigned operations.

PROPOSED SYSTEM:

MULTIPLIER USING 4-2 COMPRESSOR

We proposed three different accurate 4:2 compressor with using high speed gate level full adder design of XOR and Multiplexer circuit instead of conventional full adder design. Which have the flexibility of switching between the exact and approximate computing operating modes. In this approximate mode, these dual quality compressor provide high speed and low power consumptions at the cost of lower accuracy. Each of the compressor has its own level of accuracy in the approximation mode as well as different delay and power dissipations in the approximate and exact modes. Using these compressors in the Approximate multiplier, it will provide accuracies as well as the power and speed may change dynamically during the runtime operations. The proposed 4:2 compressor based approximate multiplier save few number of logic gates in partial product, and this proposed multiplier was evaluated in image processing application, such as image multiplication, image sharpening, and so on. This design was implemented in Verilog HDL, and synthesized in Xilinx S6LX9 FPGA and compared all parameters in terms of area, delay and power.

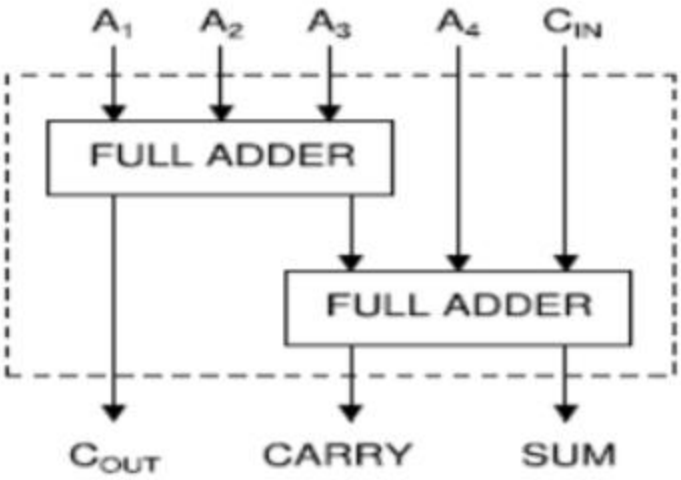

Fig: 3 conventional 4:2 compressor

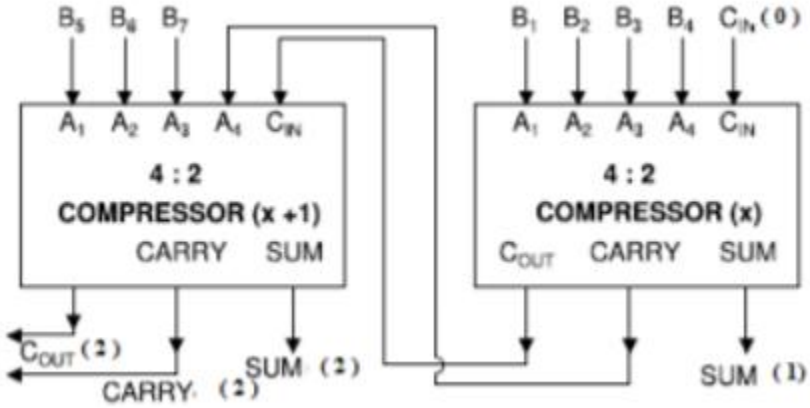

Fig: 4 4:2 compressor chain

In AI and DSP applications, one of most important arithmetic operation is multiplication. These applications require high speed multiplier architectures to involve high speed parallel operations with tolerable levels of accuracy. Introduction of approximation in multipliers leads to design of faster computations with minimal hardware complexity, delay and power, with accuracy in required levels. In multiplication process Partial product summation is the one of speed limiting operation cause the propagation delay in adder structure. In order to minimize the propagation delay, compressors are introduced. Compressors calculate the sum and carry at each stage simultaneously. The resultant carry is added with a higher significant sum bit in the next level. This process is continued until the final product is produced. A 4–2 compressor has five inputs (A1; A2; A3; A4; Cin) and three outputs (Sum; Carry; Cout). All the input bits and outputs have

the same binary weight. The compressor receives the input Cin from a previous block of order one binary bit lower in significance, and produces outputs Cout and Carry of order one binary bit higher in significance. The general block diagram of exact 4–2 compressor is shown in Fig.1.

SUM = A1 XOR A2 XOR A3 XOR A4 XOR Cin ---------- (1)

CARRY= Cin(A1 XOR A2 XOR A3 XOR A4)+A4 (~(A1 XOR A2 XOR A3 XOR A4))---- (2)

Cout = A3 (A1 XOR A2)+A1(~(A1 XOR A2)) ----------(3)

The approximate 4–2 compressor can be implemented with reducing the number of output bits to two. This approximate compressor is used in the implementation of multiplier, the reduction in the number of output bits effectively minimize the number of input bits of the succeeding compressors. Except the input combination ‘‘X4X3X2X1 = 1111”, in the remaining 15 cases the value of output bit Cout = 0. So in designing approximate compressors, Cout is not considered.

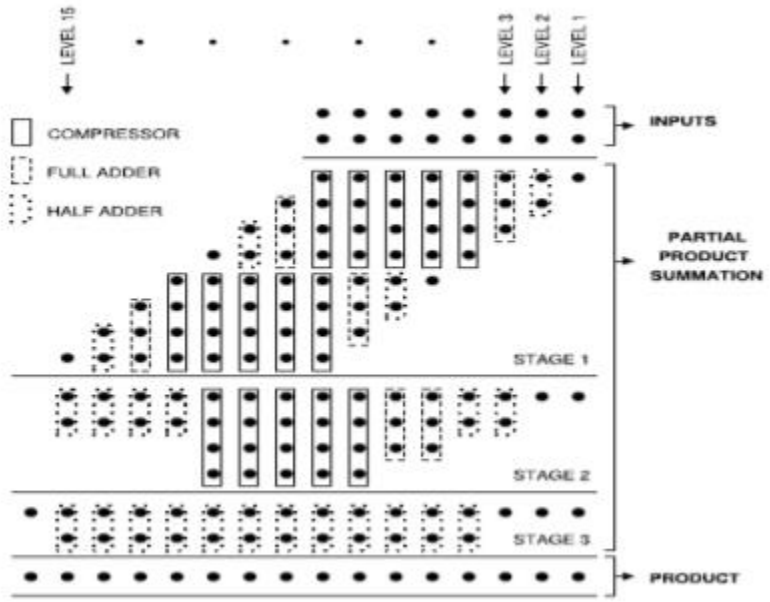

Fig: 5 8×8 approximate multiplier

Half adder, the carry bit Ch is defined as

Ch (X0,X1) = X0. X1

Full adder, the carry bit Cf is defined as

Cf (X0,X1,X2) = X0. X1 + X1. X2 + X0. X2

The Carry output of our approximate 5:2 compressor is

Cf (X0,X1,X2) +Ch (X3,X4) + Ch (X0+X1+X2,X3+X4).

The Carry output of our approximate 8:2 compressor is as

Cf (X0,X1,X2) + Cf (X3,X4,X5) + Ch(X6,X7) +Cf(X0+X1+ X2, X3+X4+X5, X6+X7).

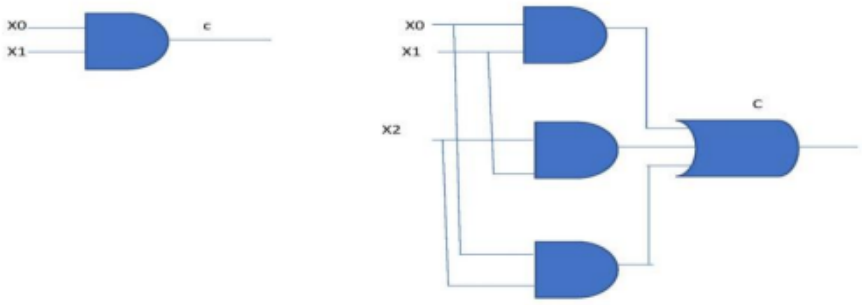

Fig. 6: Modified Half adder And Modified Full adder

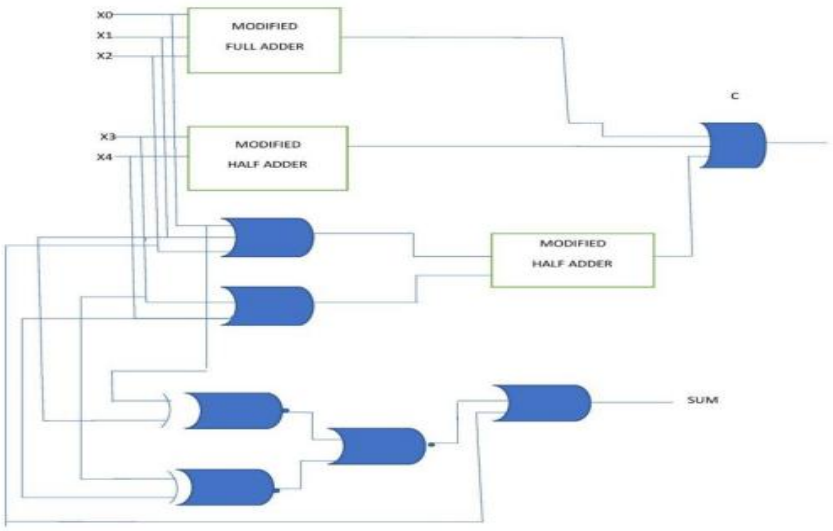

Fig. 7: Sum and Carry Output For Compressor

To reduce the power consumption with a small error, our PPM reduction circuitry applies the significance driven logic compression technique as below: the higher significance weights use accurate (i.e., exact) 4:2 compressors; the middle significance weights use approximate high-order compressors; the lower significance weights use inaccurate compressors (OR-tree based approximation). PPM reduction circuitry has two stages. The first stage is for all the weights. The second stage is only for the higher significance weights. After the second stage is completed, each weight has at most two product terms. For each lower significance weight, we use a simple OR tree based approximation for power saving. Suppose that the number of inputs is n. If $n \leq 2$, no action is performed. On the other hand, if $n > 2$, we use an OR tree for n-1 inputs to approximate the accumulation result of these n-1 inputs. Thus, after the first stage is done, each lower significance weight has at most two product terms. For each middle significance weight, we use our approximate n:2 compressor for power saving, where n is the number of product terms in this weight. The second stage is only for the higher significance weights. In order to achieve high accuracy, we use accurate (i.e., exact) 4:2 compressors to reduce the maximum height of the PPM. The carry bit Cin of the rightmost accurate 4:2 compressor is set to be 0. As shown in Fig. 5, after the second stage is completed, each higher significance weight has two product terms.

SIMULATION RESULTS:

All the synthesis and simulation results are performed using Verilog HDL. The synthesis and simulation are performed on Xilinx ISE 14.7. The simulation results are shown below figures. The corresponding simulation results of the Approximate 4:2 Multiplier are shown below.

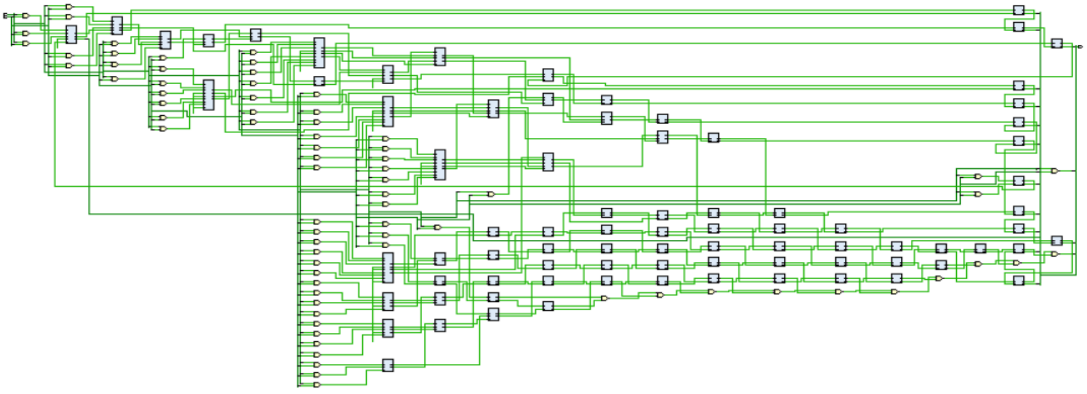

Fig 8: RTL Schematic

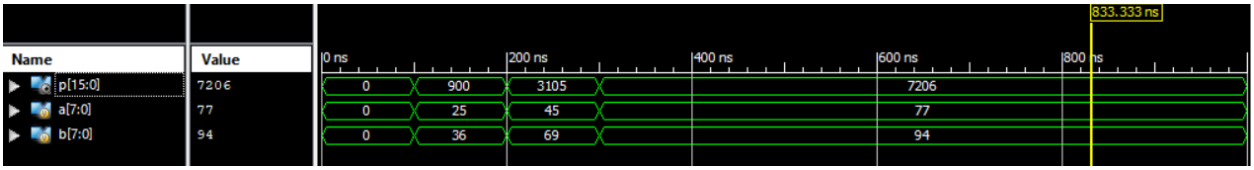

Fig 9 : Simulation Results

TABLE I

COMPARISON TABLE

PARAMETER	EXISTING SYSTEM	PROPOSED SYSTEM
DELAY(ns)	22.950	21.41
NO.OF LUT'S	190	156

CONCLUSION

A novel approximate 4 -2 compressor designs are presents in this paper. Firstly, a high speed area efficient compressor design is proposed, which attained a considerable reduction in area, delay and power when compared to other state-of-the-art approximate compressor designs. The proposed approximate multiplier design has accuracy with 25% error rate and equal positive and negative absolute error deviation of 1.The proposed approximate multiplier shows a significant improvement in terms of area, power consumption and delay as compared to the existing approximate multiplier. In conclusion, this work has shown that multiplier can be implemented for approximate computing by an approximate design of a compressor; this proposed multiplier offers advantages in terms of design parameters compared to existing approximate multipliers, and in terms of accuracy metrics, area, delay and power consumption.

REFERENCES

[1] M. Alioto, "Ultra-low power VLSI circuit design demystified and explained: A tutorial," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59, no. 1, pp. 3–29, Jan. 2012.

[2] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

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

[4] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: Modeling and analysis of circuits for approximate computing," in Proc. Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.

[5] F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in Proc. 17th Int. Symp. Comput. Archit. Digit. Syst. (CADS), Oct. 2013, pp. 25–30.

[6] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in Proc. 24th Int. Conf. VLSI Design, Jan. 2011, pp. 346–351.

[7] D. R. Kelly, B. J. Phillips, and S. AlSarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.

[8] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application," in Proc. IEEE Int. Conf. Electron Devices SolidState Circuits (EDSSC), Dec. 2010, pp. 1–4.

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

[10] K. Bhardwaj and P. S. Mane, "ACMA: Accuracy-configurable multiplier architecture for error-resilient system-on-chip," in Proc. 8th Int. Workshop Reconfigurable Commun.- Centric Syst.-Chip, 2013, pp. 1–6.

[11] 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. Quality Electron. Design (ISQED), 2014, pp. 263–269.

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

[13] V. Mahalingam and N. Ranganathan, "Improving accuracy in Mitchell's logarithmic multiplication using operand decomposition," IEEE Trans. Comput., vol. 55, no. 12, pp. 1523–1535, Dec. 2006.

[14] Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available: http://www.nangate.com/

[15] H. R. Myler and A. R. Weeks, The Pocket Handbook of Image Processing Algorithms in C. Englewood Cliffs, NJ, USA: Prentice-Hall, 2009.

[16] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.

[17] S. Hashemi, R. I. Bahar, and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD), Austin, TX, USA, 2015, pp. 418–425.