IJIREEICE

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering
Vol. 8, Issue 8, August 2020

DOI 10.17148/IJIREEICE.2020.8810

High Performance and Error Tolerant Approximate Multiplier Based on Multi-bit Compressors

Dr.Senthil Kumar[1], Mohanasundari[2]

Professor, Department of ECE, Kalaignar Karunanidhi Institute of Technology, Coimbatore, Tamilnadu, India[1]

PG Scholar, Kalaignar Karunanidhi Institute of Technology, Coimbatore, Tamilnadu, India[2]

Abstract: Approximate computing is an emerging trend in digital design that trades off the requirement of exact computation for improved speed and power performance. The proposed method uses a novel approximate compressor and an algorithm to exploit them for the design of efficient approximate multipliers. The approximate compressors are a key element in the design of power-efficient approximate multipliers, the number of faulty rows in the compressor's truth table is significantly reduced by encoding its inputs using generate and propagate signals. Multi-bit approximate counters with 4:2, 6:3 and 7:3 counters are used to in each column to increase its performance. It uses bit stacking at initial stages and then converted to binary counts, producing counter output with no xor gates on the critical path. This avoidance of xor gates results in faster designs with efficient power and area utilization. Based on this improved counters, 8x8 multipliers are designed and then are used as building blocks for scaling up to 16×16 and 32×32 multipliers. Compared with existing compressor-based multiplier, the proposed multi-bit compressor-based multiplier results in low power consumption and high performance.

Keywords: Approximate computing, Compressors, Stacking, Multiplier.

I. INTRODUCTION

Approximate computing is an emerging trend in digital design [1], [2], relaxing the requisite of exact computation to gain substantial performance improvement in terms of power, speed and area. This approach is becoming more and more important for embedded and mobile systems, characterized by severe energy and speed constraints. Approximate computing can be fruitfully applied in several error-resilient applications. Examples are multimedia processing [3], data mining and recognition, machine learning. Multipliers are fundamental subsystems for microprocessors, digital signal processors, and embedded systems with applications ranging from filtering to convolutional neural networks. Unfortunately, multipliers are characterized by complex logic design [4] and constitute one of the most energy-hungry digital blocks. Therefore, approximate multiplier design has become an important research subject in recent years. A multiplier includes a few basic blocks: partial products generation, partial products reduction and carry-propagate addition. Approximations can be introduced in any of these blocks. For example, truncation of the partial products is a well-established approximation technique in which some of the partial products are not formed and the truncation error is reduced with the help of suitable correction functions.

For embedded applications, it has become essential to design more power-aware multipliers. Given their fairly complex structure and interconnections, multipliers can exhibit a large number of unbalanced paths, resulting in substantial glitch generation and propagation. This spurious switching activity can be mitigated by balancing internal paths through a combination of architectural and transistor-level optimization techniques. In addition to equalizing internal path delays, dynamic power reduction can also be achieved by monitoring the effective dynamic range of the input operands so as to disable unused sections of the multiplier and/or truncate the output product at the cost of reduced precision. This is possible because, in most sensor applications, the actual inputs do not always occupy the entire magnitude of its word-length.

For example, in artificial neural network applications, the weight precision used during the learning phase is approximately twice that of the retrieval phase. Besides, operations in lower precisions are the most frequently required. In contrast, most of today's full-custom DSPs and application-specific integrated circuits (ASICs) are designed for a fixed maximum word-length

so as to accommodate the worst-case scenario. Therefore, an 8-bit multiplication computed on a 32-bit Booth multiplier would result in unnecessary switching activity and power loss. In addition, the critical path may change as a result of the varying supply voltage or process or temperature variations. If this occurs, computations will completely fail regardless

This work is licensed under a Creative Commons Attribution 4.0 International License

IJIREEICE

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering
Vol. 8, Issue 8, August 2020

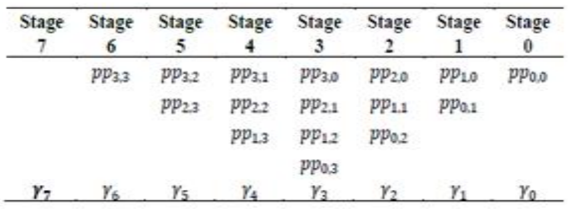
DOI 10.17148/IJIREEICE.2020.8810

of the safety margins. The aforementioned limitations of conventional DVS techniques motivated recent research efforts into error-tolerant DVS approaches, which can run-time operate the circuit even at a voltage level at which timing errors occur. A recovery mechanism is then applied to detect error occurrences and restore the correct data. Because it completely removes worst case safety margins, error-tolerant DVS techniques can further aggressively reduce power consumption. The rest of the paper is organized as follows: Section II briefs about partial product generation and exact compressors. Section III describes the Multi-bit approximate compressors. The proposed multiplier based on compressors is explained in section IV. Results are evaluated in section V and section VI concludes the paper.

II. PARTIAL PRODUCT GENERATION AND EXACT COMPRESSION

The multiplication process consists of 3 steps:
• Partial product generation,
• Partial product reduction and
• Final carry propagating addition.

Various recoding schemes are used to reduce the number of partial products. Compressors have been widely used for reduction process which usually contributes the most to the delay, power and area of the multiplier. To achieve a better performance, the use of higher order compressors instead of conventional compressors, e.g. 3:2 compressors, have been considered. Fig.1 shows the partial product generation of 4X4 Multiplier.



Stage 7	Stage 6	Stage 5	Stage 4	Stage 3	Stage 2	Stage 1	Stage 0
	$pp_{3,3}$	$pp_{3,2}$	$pp_{3,1}$	$pp_{3,0}$	$pp_{2,0}$	$pp_{1,0}$	$pp_{0,0}$
		$pp_{2,3}$	$pp_{2,2}$	$pp_{2,1}$	$pp_{1,1}$	$pp_{0,1}$	
			$pp_{1,3}$	$pp_{1,2}$	$pp_{0,2}$		
				$pp_{0,3}$			
γ_7	γ_6	γ_5	γ_4	γ_3	γ_2	γ_1	γ_0

Fig.1 Partial Product of 4X4 Multiplier

The function of the exact 4:2 compressor is implemented by using two appropriately connected full adders is shown in Fig.2.

sum = $x1 \oplus x2 \oplus x3 \oplus x4 \oplus cin$,
cout = $(x1 \oplus x2).\, x3 + \overline{(x1 \oplus x2).\, x1}$,
carry = $(x1 \oplus x2 \oplus x3 \oplus x4).\, cin + \overline{(x1 \oplus x2 \oplus x3 \oplus x4)}.\, x4$. -------(1)

The sum output has the same weight as the four input signals while the cout is used as the carry in for the next higher-order compressor and the output carry is weighted like a pp bit in a one-bit-higher position. Note that cout and carry have the same weight.

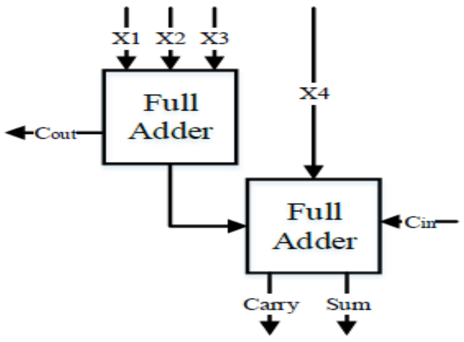


Fig.2. Exact 4:2 Compressors

III. MULTI-BIT APPROXIMATE COMPRESSORS

A. Approximate 4:2 Compressor

The proposed 4×4 approximate multipliers considers the carry from the previous stage (c4) and uses an exact full-adder to add pp terms pp3,2, pp2,3, and c4.Since c4 is generated from the four LSBs, it does not introduce a large error in an 8×8 multiplier. Note that ignoring c4 breaks the longest path (that is, the carry propagation) and it is a common technique to reduce the circuit's latency. The sixth sum output of the full adder in design are both denoted by $\gamma5$ and the

This work is licensed under a Creative Commons Attribution 4.0 International License

IJIREEICE
ISSN (Online) 2321-2004
ISSN (Print) 2321-5526
International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering
Vol. 8, Issue 8, August 2020
DOI 10.17148/IJIREEICE.2020.8810

corresponding carry signal, $c5$, goes to the next stage to be added to pp3,3 using an exact half adder. The sum and carry outputs of this final half adder produce $\gamma 6$ and $\gamma 7$, respectively. Fig. 3 shows different blocks used for reducing the partial products. These blocks include: (1) half adders, (2) full adders, and (3) 4:2 compressors.

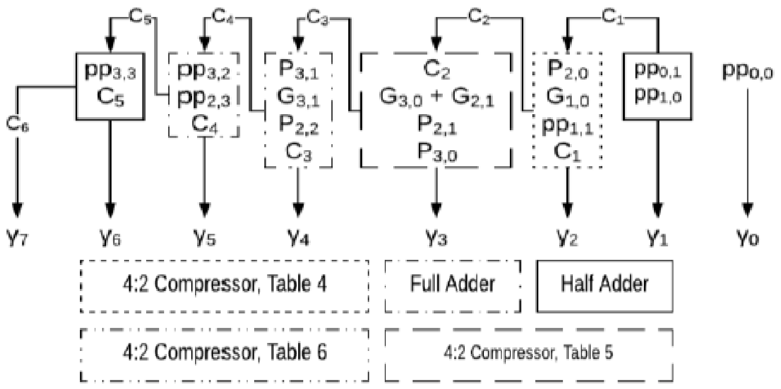


Fig.3. Partial product reduction in multiplier

The function of an exact 4:2 compressor can be approximated to reduce the hardware cost. It has been shown that cout does not have a significant impact on the compressor's accuracy, so cout is ignored in our design. Moreover, our SPICE simulations confirm that an XOR gate consumes more power and is slower than the AND and OR gates, Ignoring cout and not using XOR gates as well as our goal to use as few gates as possible led to the approximate compressor.

The sum and carry signals in the compressor for Stage 2 can be simplified as
sum =x1+x3, carry =x2+x4.------------(2)
The sum and carry signals in the compressor for Stage 3 can be simplified as
sum =x1+x3+x4, carry =x1.x2+x3.x4.----(3)
The sum and carry signals in the compressor for Stage 4 can be simplified as
sum =x1+x2+x3, carry =x2+x3.x4.-------(4)

B. Approximate 6:3 and 7:3 Compressors based on symmetric stacking

The proposed 6:3 counter is realized by first stacking all of the input bits such that all of the "1" bits are grouped together. After stacking the input bits, this stack can be converted into a binary count to output the 6- bit count. Small 3-bit stacking circuits are first used to form 3-bit stacks. These 3-bit stacks are then combined to make a 6-bit stack using a symmetric technique that adds one extra layer of logic.

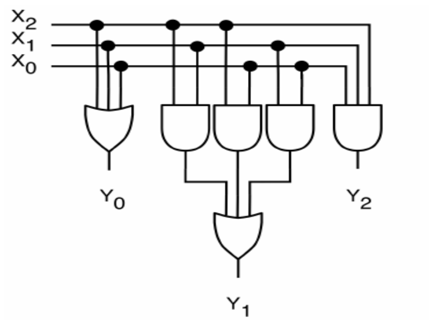


Fig.4. Three bit stacking circuit

The 3-bit stacking circuit is shown in Fig 4.
Given inputs X0, X1, and X2, a 3-bit stacker circuit will have three outputs Y0, Y1, and Y2 such that the number of "1" bits in the outputs is the same as the number of "1" bits in the inputs, but the "1" bits are grouped together to the left followed by the "0" bits. It is clear that the outputs are then formed by

Y0 = X0 + X1 + X2
Y1 = X0X1 + X0X2 + X1X2
Y2 = X0X1X2.-----------------------------(5)

Namely, the first output will be "1" if any of the inputs is one, the second output will be "1" if any two of the inputs are one, and the last output will be one if all three of the inputs are "1." The Y1 output is a majority function and can be implemented using one complex CMOS gate.

This work is licensed under a Creative Commons Attribution 4.0 International License

IJIREEICE

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering
Vol. 8, Issue 8, August 2020

DOI 10.17148/IJIREEICE.2020.8810

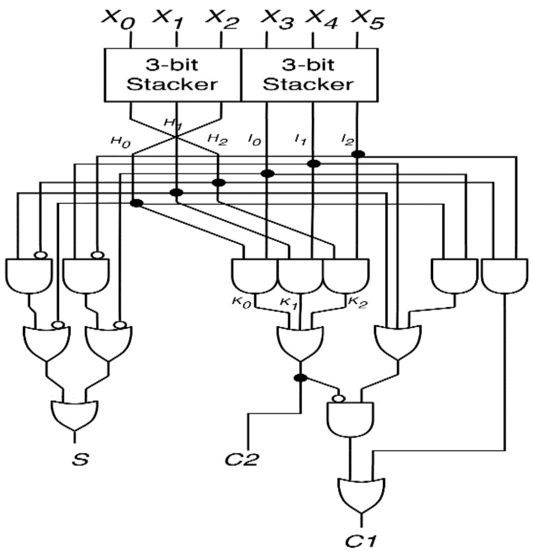


Fig.5. Approximate 6:3 Compressors

In order to implement a 6:3 counter circuit, the 6-bit stack described must be converted to a binary number as shown in Fig.5. For a faster, more efficient count, we can use intermediate values H, I, and K to quickly compute each output bit without needing the bottom layer of stackers. Call the output bits C2, C1, and S in which C2, C1, S is the binary representation of the number of "1" input bits. To compute S, we note that we can easily determine the parity of the outputs from the first layer of 3-bit stackers.

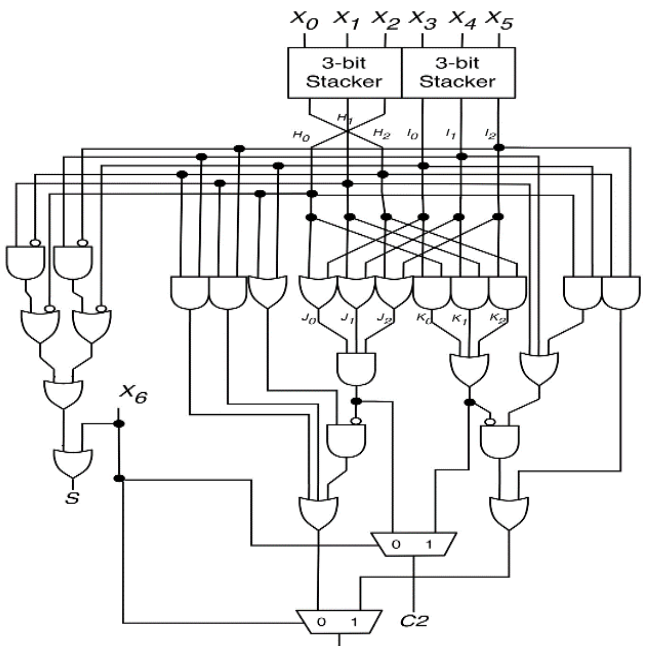


Fig.6. Approximate 7:3 Compressors

Even parity occurs in the H if zero or two "1" bits appear in X0, X1, and X2. Thus, He and Ie, which indicate even parity in the H and I bits, are given by

He = H0b + H1.H2b
Ie = I0b + I1.I2b. ---------------------------(6)

As S indicates odd parity over all of the input bits, and because of the approximation, we can compute it as

S = He + Ie. --------------------------------(7)

The approximate 7:3 compressor is shown in Fig.6. The same procedure is used to construct 7:3 approximate compressors, but it uses additional input X6 which acts as a selection line for selecting between output of 6 and 7 bits.

This work is licensed under a Creative Commons Attribution 4.0 International License

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

Vol. 8, Issue 8, August 2020

DOI 10.17148/IJIREEICE.2020.8810

IV. PROPOSED MULTIPLIER BASED ON MULTI-BIT COMPRESSORS

In order to construct larger, e.g. 16×16 and 32×32, approximate multipliers, the two proposed 4×4 multipliers are combined in an array structure. For instance, to construct an 8×8 multiplier using a 4× 4 design, the two 8-bit operands A and B are partitioned into two 4-bit nibbles, namely αH and αL for A and βH and βL for B. Note that αH and βH are the 4 MSBs and αL and βL indicate the 4 LSBs of A and B, respectively. Each two of these four nibbles (in total 4 possible combinations) are multiplied using 4×4 multipliers and the partial products are then shifted (based on the nibble's importance) and added together (using a Wallace tree architecture) to produce the final multiplication result. Building 2n×2n multipliers using n×n multipliers is specified in Fig. 7. The proposed approximate compressor can also be utilized in signed Booth multipliers. In a Booth multiplier, the partial products are generated using a Booth encoder, and the major difference between the unsigned and signed Booth multiplication is in the generation of the partial products. Therefore, the partial products in Booth multipliers can be accumulated using approximate compressors.

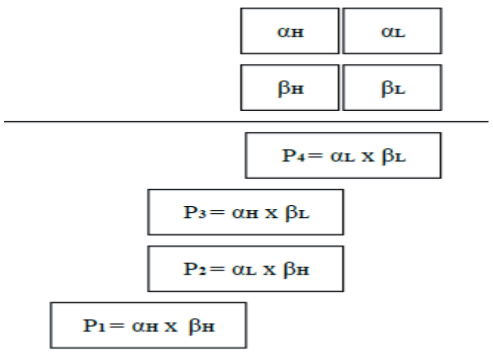

α_H
α_L
β_H
β_L
$P_4 = \alpha_L \times \beta_L$
$P_3 = \alpha_H \times \beta_L$
$P_2 = \alpha_L \times \beta_H$
$P_1 = \alpha_H \times \beta_H$

Fig.7. Building 2n×2n multipliers using n×n multipliers.

V. EVALUATION RESULTS

Table 1 show that various parameters like area, power and delay of 32 bit Approximate multiplier. From the fundamental 4 bit approximate multiplier 8, 16 and 32 bit multiplier are computed.

Table.1. Comparison Results of Multiplier

Parameters	Area (Gate Count)	Power (mW)	Delay(ns)
Conventional Approximate Multiplier	16615	453.87	45.626
Approximate Multi-bit Compressor Multiplier	4937	355.34	38.189

The proposed multi-bit compressor based multiplier results in low power consumption and higher performance compared to conventional approximate multiplier. Fig 8 shows the simulation result of Multiprecision multiplier. Based on the operand selection 8 bit, 16 bit or 32 bit operation is performed.

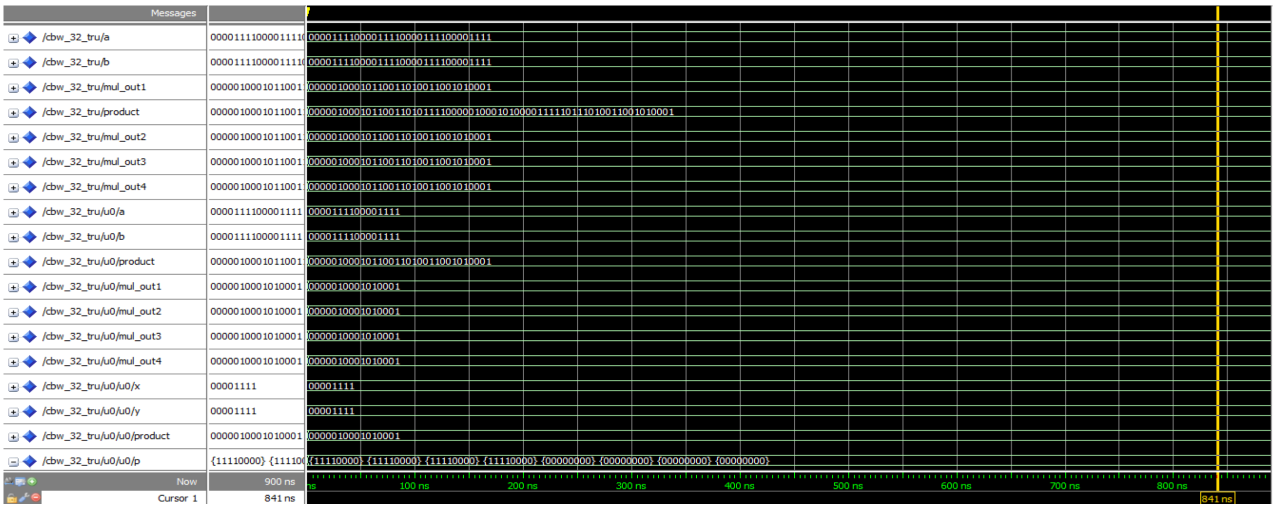

```
Messages
/cbw_32_tru/a                0000111100001111( 00001111000011110000111100001111
/cbw_32_tru/b                0000111100001111( 00001111000011110000111100001111
/cbw_32_tru/mul_out1         000001000101100 1 000001000101100110100110010 10001
/cbw_32_tru/product          000001000101100 1 0000010001011001101011110000010001010000111110111010011001010001
/cbw_32_tru/mul_out2         000001000101100 1 00000100010110011010011001010001
/cbw_32_tru/mul_out3         000001000101100 1 00000100010110011010011001010001
/cbw_32_tru/mul_out4         000001000101100 1 00000100010110011010011001010001
/cbw_32_tru/u0/a             0000111100001111  0000111100001111
/cbw_32_tru/u0/b             0000111100001111  0000111100001111
/cbw_32_tru/u0/product       000001000101100 1 00000100010110011010011001010001
/cbw_32_tru/u0/mul_out1      0000010001010001  0000010001010001
/cbw_32_tru/u0/mul_out2      0000010001010001  0000010001010001
/cbw_32_tru/u0/mul_out3      0000010001010001  0000010001010001
/cbw_32_tru/u0/mul_out4      0000010001010001  0000010001010001
/cbw_32_tru/u0/u0/x          00001111          00001111
/cbw_32_tru/u0/u0/y          00001111          00001111
/cbw_32_tru/u0/u0/product    0000010001010001  0000010001010001
/cbw_32_tru/u0/u0/p          {11110000} {11110( {11110000} {11110000} {11110000} {11110000} {00000000} {00000000} {00000000} {00000000}
                      Now    900 ns            ns  100 ns  200 ns  300 ns  400 ns  500 ns  600 ns  700 ns  800 ns
                 Cursor 1    841 ns                                                                          841 ns
```

Fig. 8 Simulation Result of Multiprecision Multiplier

This work is licensed under a Creative Commons Attribution 4.0 International License

IJIREEICE

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

International Journal of Innovative Research in
Electrical, Electronics, Instrumentation and Control Engineering
Vol. 8, Issue 8, August 2020

DOI 10.17148/IJIREEICE.2020.8810

VI. CONCLUSION

A low power multi-bit compression-based multiplier based on approximate 4:2, 6:3 and 7:3 compressor is proposed. The 8x8 multipliers with different accuracies are constructed. The 8×8 designs are then scaled up to 16×16 and 32×32 multipliers that provide a wide range of accuracy-performance trade-offs. The 6:3 and 7:3 compressors counters implemented with this bit stacking technique achieve higher speed than other higher order counter designs while reducing power consumption. This is due to the lack of XOR gates and multiplexers on the critical path. The 32-bit wallace tree multipliers built using the proposed compressors outperform the standard wallace tree implementation. The proposed method results in 16.3% performance improvement along with 21.7% power reduction compared with existing design.

REFERENCES

[1]. A. J. Sanchez-Clemente, L. Entrena, R. Hrbacek, and L. Sekanina, “Error mitigation using approximate logic circuits: a comparison of probabilistic and evolutionary approaches,” IEEE Transactions on Reliability, vol. 65, no. 4, pp. 1871-1883, 2016.
[2]. J. Schlachter, V. Camus, K. V. Palem, and C. Enz, “Design and applications of approximate circuits by gate-level pruning,” IEEE Transactions on Very Large-Scale Integration Systems, vol. 25, no. 5, pp.1694-1702, 2017.
[3]. B. Moons and M. Verhelst, “Energy-efficiency and accuracy of stochastic computing circuits in emerging technologies,” IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 4, no. 4, pp. 475–486, 2014.
[4]. J.Han & M.Orshansky, “Approx computing: an emerging paradigm for energy-efficient design,” IEEE European Test Symposium, pp.1-6, 2013.
[5]. C. Liu, “Design and analysis of approximate adders and multipliers,” Master's Thesis, University of Alberta, Canada, 2014.
[6]. A. Wang and A. Chandrakasan, “Energy-aware architectures for a realvalued FFT implementation,” in Proc. IEEE Int. Symp. Low Power Electron. Design, Aug. 2003, pp. 360–365.
[7]. T. Kuroda, “Low power CMOS digital design for multimedia processors,” in Proc. Int. Conf. VLSI CAD, Oct. 1999, pp. 359–367.
[8]. H. Lee, “A power-aware scalable pipelined booth multiplier,” in Proc. IEEE Int. SOC Conf., Sep. 2004, pp. 123–126.
[9]. S.-R. Kuang and J.-P. Wang, “Design of power-efficient configurable booth multiplier,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 3, pp. 568–580, Mar. 2010.
[10]. O. A. Pfander, R. Hacker, and H.-J. Pfleiderer, “A multiplexer-based concept for reconfigurable multiplier arrays,” in Proc. Int. Conf. Field Program. Logic Appl., vol. 3203. Sep. 2004, pp. 938–942.
[11]. N. Maheshwari, Z. Yang, J. Han, and F. Lombardi, “A design approach for compressor based approximate multipliers,” IEEE International Conference on VLSI Design, pp. 209-214, 2015.
[12]. S.Venkatachalam, S.B.Ko, “Design of power & area efficient approximate multipliers,” IEEE Transactions on VLSI, vol.25, no.5, pp.1–5, 2017.
[13]. W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, and F. Lombardi, “Design of approximate Radix-4 Booth multipliers for error-tolerant computing,” IEEE Transactions on Computers, vol. 66, no. 8, pp. 1435-1441, 2017.
[14]. L. Qian, C. Wang, W. Liu, F. Lombardi, and J. Han, “Design and evaluation of an approximate Wallace-Booth multiplier,” IEEE International Symposium on Circuits and Systems, pp. 1974-1977, 2016.
[15]. J. Liang, J. Han, and F. Lombardi, “New metrics for the reliability of approximate and probabilistic adders,” IEEE Transactions on Computers, vol. 62, no. 9, pp. 1760–1771, 2013.
[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 Transactions on Very Large Scale Integrgration Systems, vol. 23, no. 6, pp. 1180–1184, 2014.

This work is licensed under a Creative Commons Attribution 4.0 International License