IMPLEMENTATION OF HIGH-SPEED COUNTERS AND COMPRESSORS GENERATED BY SORTING NETWORK

1 K. SUMALATHA (Emai id:sumakunti123@gmail.com) 2 V. SHANKAR
1 PG Student, Department of Electronics and Communication Engineering G. Narayanamma Institute of Technology & Science Autonomous (for women) Shaikpet, Hyderabad, Telangana, India
2 Assistant Professor, Department of Electronics and Communication Engineering G. Narayanamma Institute of Technology & Science Autonomous (for women) Shaikpet, Hyderabad, Telangana, India

ABSTRACT: This project consists of an innovative way of rapid counters such as binary counters (7, 3), (15, 4) etc., and sorting network based approximate (4:2) compressors. To enhance the speed of process, a very high compressor counters and compressors are required. The counter inputs are split asymmetrically into 2 parts first and then given them as inputs to the sorting networks to form the rearranged sequences, that will alone represented by one-hot code sequences. By using this technique, we can build and further optimize the (7, 3) counter which do good in most cases than other designs for parameters such as latency, total area, and power consumption. A (15, 4) counter is built, and it gets a lesser latency even though having substantially low power consumption and taking up significantly less space. Besides, we also construct (4:2) approximate compressors based on sorting network. To analyze the performance of circuits designed, they have embedded in 8 x 8, 16 × 16 bit multiplier. A 24Bit multiplier is made and the effectiveness of the design is synthesized and simulated using Xilinx Vivado.

Keywords: - Binary counters, sorting network, approximate 4:2 compressor, 24 bit multiplier, one-hot code.

I. INTRODUCTION:

The Wallace Tree structure, which is the basic multiplier's bottleneck, sums up all the partial products in a basic multiplier circuit. The summing of several operands is used in various areas of the circuit. The summation of several operands is used as the primary processing. A Wallace tree structure is well-known way of summarizing numerous operands, and its upgraded version, Wallace tree, is even more well-known. To speed up the summing, these approaches employ complete adders as (3,2) counters. The architecture used is referred to as a "carry–save." Then onwards several research been published that look at ways to build a framework that speed up summing process. The fundamental concept is to use further bits at the same weight to build a counter or compressor having a greater compressing ratio comparing with (3,2) counter. By looking at the carry bits between neighboring columns, the compressors compress n rows into 2 rows. Compressors that compress four, five, or seven rows into two rows have been explored in certain articles (4,2), (5,2), and (7,2), respectively. They are, however, still part of the whole adder structure, which uses "XOR" gates as fundamental unit, and its logic are hard to reduce.

N rows are compressed into log2n rows by the counters. Counters (4,3), (5,3), and (6,3), as well as counters (7,3) and (15,4), have been explored in certain studies. The inputs are counted for the number of "1"s. For a saturated counter compressed results accurately show all of the "1"s in the inputs.

A symmetric stacking structure was presented. It is really quick in comparison to other designs, however it is unsaturated. Then, on the crucial route, they employ a MUX to build a (7,3) saturated counter which influences the swiftness. Further recommended reducing the intended (6:3) counter to a (5:3) counter and combining three (5:3) counters to form a (15:4) counter. This strategy, however, is ineffective. We begin by reviewing the design as the principal comparison object. They suggested a rapid counter having symmetric stacking structure, and based on this (6,3) counter, they built a (7,3) saturated counter. Although it is the quickest of the seven counter designs (7,3), it has the worst delay performance due to the fact that it just adds one Multiplexer to the route without optimizing. We offer this way of constructing a (7,3) counter directly to address the problem. In contrast to

the symmetric stacking structure, we begin by asymmetrically stacking two sorting networks. Approximate multipliers are commonly employed to speed up multiplication. Estimated booth encoding and partial product perforation are used to obtain an approximate multiplier.

A symmetric stacking structure is used in high-speed approximation (4:2) compressors. There are saturation counters with better efficiency in this design, namely (7,3) and (15,4). We begin by categorizing networks asymmetrically in this collection of designs. The new design is then optimized via logical simplification with the use of two extended bits. With the sorting network, we can also make exact/approximate (4:2) compressors. The sorting network is a high-performance parallel hardware network for sorting data. Any number can be sorted if a sorting network can sort a batch of data whose constituents are all 1-bit integers, according to the famous 0,1 principle. Only 1-bit data sorting is used in this article.

The standard 3&4 way sorting networks. A) Sorting Network Working Principle: Every standing bar is a sorter with 2 data inputs and outputs, with all data being one-bit values. The larger input is always sorted first, followed by the smaller. The input of a 4 SN is in order [0, 1, 1, 1], whereas the input of a 3 way sorting network is sequence [0, 1, 1]. (3 SN). After three layers of sorters, the input sequences for both 4 SN and 3 SN are reordered with greater at high and the smaller at low.

B) 1-Bit Data Sorter: From previously stated, the sorter rearranges 2 inputs based on number. The logical circuit can easily sort two 1-bit data sets. A sorter uses 1 layer of 2-input basic logic gates, whereas 3 &4 way sorting networks use 3 layers of 2-input basic logic gates.

Exact or high-precision computing, on the other hand, isn't always required. Small mistakes, on the other hand, can compensate for each other or have no substantial impact on the computed results. As a result, approximation computing is evolved as a novel method with high accuracy.

The approximate computing is having a system design that is both high-performance and energy-efficient. Because addition mistakes are more sensitive than multiplication errors in such complicated computations, multipliers may tolerate a larger approximation than adders. Along with gain in factors such as performance and power consumption for these applications, the adoption of approximation arithmetic circuits will increase the image processing quality and deep learning. Rapid system with reduction in complex nature and consumed power emerge from arithmetic processes that are approximated. The compromise is to be a loss of accuracy, which would not necessarily impede machine learning and multimedia applications in their usual operation. The human eye's inability to discern subtle differences in photos and videos.

II. EARLIER WORK:

An efficient (7,3) counter was designed by making sequence of one-hot code. They have established 3 Boolean that reduces the Boolean equations having outputs.

The complete design is shown in Fig.1 Overall (7,3) counter circuit. Generally the way from sequence H and I to C2, C1, and S are totally not dependent. An efficient (7,3) saturated counter is designed.

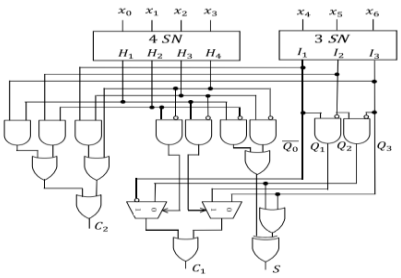

Fig. 1: Overall (7,3) counter circuit

1) 7 & 8 Way Sorting Networks: To produce the outcome, this sorting network employs 6 layers of fundamental logic gates. By deleting 1 bit from an 8 way sorting network, a 7 way sorting network with 6 layers of basic logic gates may be obtained. Sequence H (includes H1–H8 and is extended to H0–H9) and sequence I (includes I1–I7 and is extended to I0–I8), respectively, are the 8 way and 7 way output of sorting networks. The one-hot code sequences P (P0–P8) and Q (Q0–Q7) are obtained by employing A&B logic. These sequences like those in the counter (7,3).

2) (15,4) Counter: C3C2C1 S is the 4-bit output of the (15,4) counter. First, develop logic equations between

C3C2C1 S and sequences P and Q using addition of the subscripts, similar to how we did with the (7,3) counter. We adopt the Verilog syntax to express the original Boolean statements because they are excessively lengthy (specifically the ternary operator: "a?b:c").

3) Overall Structure: "HI BUS" is a module that mostly consists of AB logic gates used to calculate the relevant signals. Between sequences H and I, seven "AND" operations are required, and the results are denoted by R1–R8, i.e., R1 = H1&I7,..., R7 = H7&I1, and R8 = H8.

The logical purpose of a (4:2) compressor is similar. This also offer sorting networks to help build a rapid (4:2) compressor. The output expressions have been changed to rectify the divergence caused by inadequate sorting.

III. PROPOSED WORK:

Multiplication, division, addition, subtraction, cubing, squaring, and other arithmetic operations are performed by Arithmetic Logic Units (ALUs). Multiplication is the most basic and commonly utilised operation in ALUs of all the operations. It enables the scaling of one integer by another.

Two separate 24 bits numbers are multiplied by a 24 bit multiplier.

Because "1" is larger than "0," all "1"s are at the top of the series if there are ones & all "0"s are at the end when there are "0"s, as illustrated in Fig. Definition of a sequence. If both ones & zeros exist, the reordered sequence must have a point where the two "1"s and "0"s meet. If there are just ones & zeros in the sequence, manipulating it by placing one at top and zero at bottom to ensure that 0,1-junction always exits.

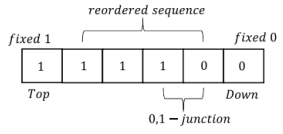

Fig. 2: Definition of a sequence.

Second, the total number of "1s" and "0s" in the reordered sequence is the same as in the initial sequence. Even the padded one will have an effect on the overall number of "1s" in the new sequence, it is established therefore it is ignored while counting.

One basic two-input logical gate layer is used by each layer of binary sorters, as shown in Fig. As a result, the 3&4 way sorting networks take similar amount of time to complete. We divide a (7,3) counter's seven inputs into two sections based on this. There are four bits in one portion and three bits in the other.

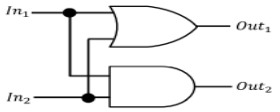

Fig. 3: Two-input binary sorter.

Find the code sequences 0,1-Junction and One-Hot: The junction only indicate a rearranged sequence under the extended fixed one and zero as illustrated in Fig. Definition of a sequence. From left to right, the 0,1-junction must be at 1,0. As a result, we'll continue to use the 4 SN as an example, and we'll end up with the structure shown in Figure 1. To produce a new sequence P0–P4, the design employs an expression (AB). There is one and only one "1" in sequence P0–P4 because there is one junction only in the rearranged and expanded sequence.

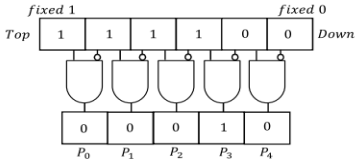

Fig. 4: One-hot code generation circuit.

This means that sequence P0–P4 is one-hot code that satisfies

P0|P1|P2|P3|P4 = 1. (1)

If the sequence elements (P0–P4) split into 2 groups in a random manner, such as P0, P2, and P4 as group 1 and P1 and P3 as group 2, then, because of one and only one “1” in the sequence, we have

P0|P2|P4 = P3|P4. (2)

This rule applies to every result of random separation. We use the same strategy to obtain the one-hot code sequence Q0–Q3 from the output sequence of a three-

way sorting network. The preceding rule is also satisfied by this sequence.

1) Generation of Basic Output: We have now 2 sequences, P and Q. P0 = 1 indicates that there is no ones in the 4N sorting network's input sequence, P1 = 1 indicates that there is one "1," Pi = 1 indicates that there are I "1"s in the input sequence, and Q indicates that there are I "1"s in the sequence. A few symbol conventions are listed below. C2, C1, and S are the outputs of the (7,3) counter, with C2 having the highest weight and S having the least. Num = 22C2 + 21C1 + 20 S.

Sequence H denotes the output of a four-way sorting network, which includes H1–H4 in order from left to right. A sequence is defined from left to right, sequence I denotes the output of a three-way sorting network, which includes I1–I3. When C2 = 1, we know the input sequence of the (7,3) counter contains at least four "1s." P4 = 1 denotes that there are four "1s" in sequence H, while Q0 = 1 denotes that there are no "1s" in sequence I. P4&Q0 = 1 indicates that the input 7 bits contain a total of 4 + 0 = 4 "1s." If total subscripts of P and Q is equal or more than 4, C2 equals 1 as a result of this form of representation. Thus, C2 can be expressed as

C2 = (P4&(Q0|Q1|Q2|Q3))|(P3&(Q1|Q2|Q3))|(P2&(Q2|Q3))|(P1&Q3). (3)

Note that the sequence Q, by same procedure in (2), satisfies

Q0|Q1|Q2|Q3 = 1 (4)

Q1|Q2|Q3 = Q0. (5)

Put (4) and (5) into (3), we get

$$C_2 = P_4 \left| \left(P_3 \& \overline{Q_0}\right) \right| \left(P_2 \& \left(Q_2 \left| Q_3\right.\right)\right) \left| \left(P_1 \& Q_3\right).\right. \quad (6)$$

C1 equals 1 when the addition of the subscripts of the sequences P and Q equals 2, 3, 6, and 7. As a result, we have:

$$C_1 = (Q_0 \& (P_2 | P_3)) | (Q_1 \& (P_1 | P_2)) | (Q_2 \& (P_0 | P_1 | P_4)) | (Q_3 \& (P_0 | P_3 | P_4)). \quad (7)$$

Note that

$$P_0 | P_1 | P_4 = \overline{P_2 | P_3} \quad (8)$$

$$P_0 | P_3 | P_4 = \overline{P_1 | P_2}. \quad (9)$$

Equation (7) is reduced to the following equation:

$$C_1 = (Q_0 \& (P_2 | P_3)) | (Q_1 \& (P_1 | P_2)) | (Q_2 \& (\overline{P_2 | P_3})) | (Q_3 \& (\overline{P_1 | P_2})). \quad (10)$$

In (10), $P_2|P_3$, $\overline{P_2|P_3}$, and $P_1|P_2$, $\overline{P_1|P_2}$, build two multichannel selection structures C1 may be effectively determined using the circuit depicted in Figure C1 generating circuit. The following equation can be used to determine S:

S = (P1|P3) ⊕ (Q1|Q3). (11)

,where ⊕ denotes “XOR.”

2) Further Optimization: We are having 2 sequences H1–H4 and I1–I3. Here, we extend sequence H1–H4 by H0 and H5. Repeat for sequence I. I0 is fixed “1,” and I4 is fixed zero. The obtained equations are:

$$P_i = H_i \& \overline{H_{i+1}}, \quad i = 0, 1, 2, 3, 4$$

$$Q_i = I_i \& \overline{I_{i+1}}, \quad i = 0, 1, 2, 3. \quad (12)$$

Furthermore, the subsequences from the sequence Q or P are supplied, and their subscripts are consecutive, the result of adding them up may be simply described by sequence I or H.

$$\sum_{n=i}^{n=j} P_n = H_i \& \overline{H_{j+1}}, \quad (0 \le i \le j \le 4)$$

$$\sum_{n=i}^{n=j} Q_n = I_i \& \overline{I_{j+1}}, \quad (0 \le i \le j \le 3). \quad (13)$$

In the earlier works a counter and compressors have been implemented that reduces the count of multipliers. In the proposed design a similar optimized way as earlier work we can extend the design to higher bit multipliers without modifying the architecture of earlier proposed multipliers.

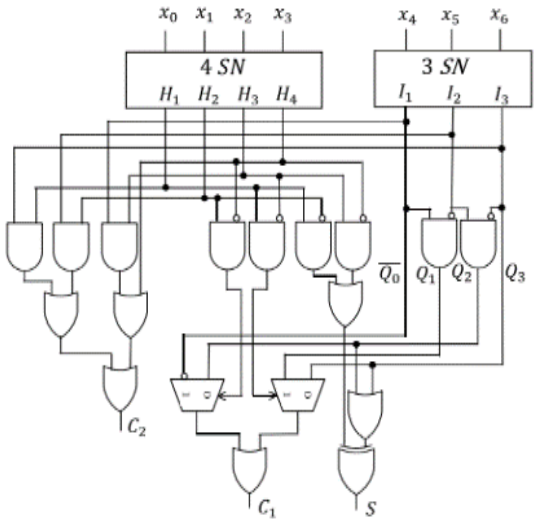

Fig.5 Overall (7,3) counter circuit

1) Seven- and Eight-Way Sorting Networks: Fig. Eight-way sorting network, shows an eight-way sorting network. This sorting network consumes six layers of basic logic gates to output the result. Removing one bit from the eight-way sorting network can obtain a seven-way sorting network that also consumes six layers of basic logic gates. The outputs of the eight-way and seven-way sorting networks are denoted as sequence H (includes H1–H8 and extended to H0 − H9) and sequence I (includes I1–I7 and extended to I0–I8), respectively. By utilizing A&B logic, we get the one-hot code sequences P (P0–P8) and Q (Q0 − Q7). These sequences are similar to the sequences in (7,3) counter.

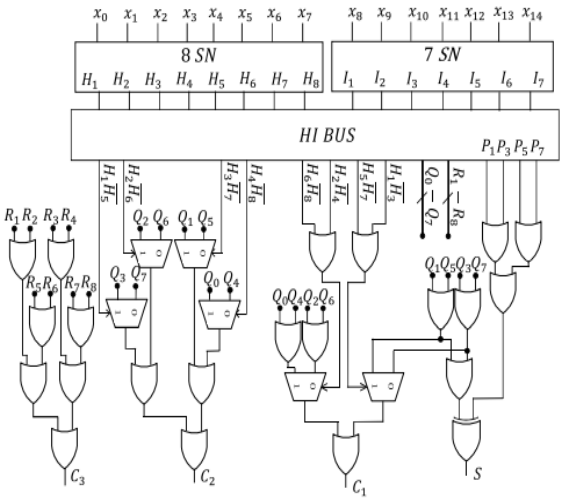

Fig.6 (15,4) counter circuit

Overall Structure: “HI BUS” is a module, which mainly contains AB logic gates for calculating the related signals. Seven “AND” operations are needed between sequence H and sequence I, and R1–R8 denote the results of “AND”, i.e., R1 = H1&I7,..., R7 = H7&I1, and R8 = H8.

A (4:2) compressor has the same logical function. To construct a high-speed (4:2) compressor, we also introduce sorting networks. In order to correct the deviation introduced by incomplete sorting, the output expressions are modified.

By using 8*8 multiplier which was implemented in the previous work, a 24*24 multiplier is designed without changing the previous architecture. A (15,4) or (31,5) saturated counter will be advantageous in various applications.

IV. EXPERIMENTAL RESULTS

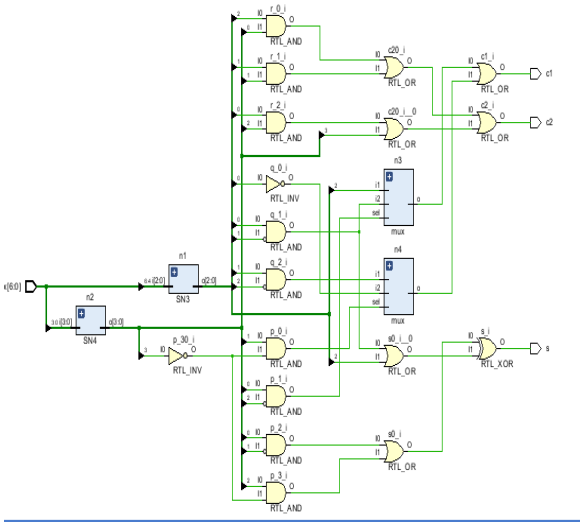

Fig. 7: RTL Schematic

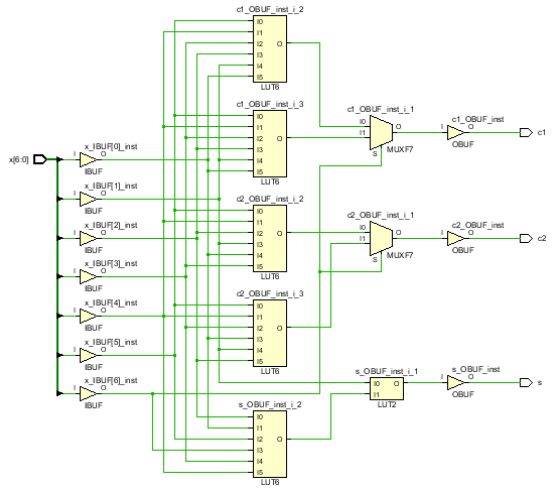

Fig. 8: Technology Schematic

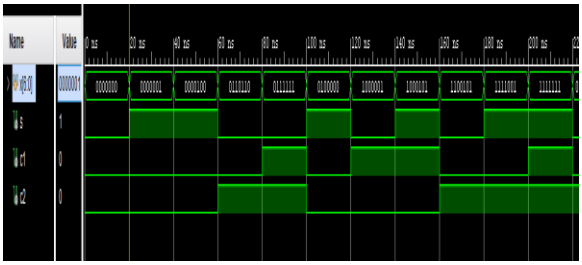

Fig. 9: Simulation Results

Evaluation table for Area, Delay:

	AREA(LUT's)	DELAY(ns)	POWER(Watts)
Bc_73	6	5.804	2.022
Bc_154	44	8.775	3.275
Bc_315	139	11.776	5.632
Mul8	122	12.041	14.187
Mul16	739	17.002	41.444

V. CONCLUSION

We can obtain optimized parameter values for higher bit multipliers by using the proposed multipliers which are 8 bit multiplier by extending it to 24 bit multiplier without changing its architecture. Hence, it can be used for higher bit multiplications.Thus area for the higher order multipliers would be same as that of 8 bit multiplier which results in area efficient circuit.

VI. REFERENCES

[1] C. S. Wallace, "A suggestion for a fast multiplier," IEEE Trans.Electron. Comput., vol. EC-13, no. 1, pp. 14–17, Feb. 1964, doi:10.1109/PGEC.1964.263830.

[2] R. S. Waters and E. E. Swartzlander, "A reduced complexity wallace multiplier reduction," IEEE Trans. Comput., vol. 59, no. 8,pp. 1134–1137, Aug. 2010, doi: 10.1109/TC.2010.103.

[3] P. L. Montgomery, "Five, six, and seven-term karatsuba-like formulae,"IEEE Trans. Comput., vol. 54, no. 3, pp. 362–369, Mar. 2005, doi:10.1109/TC.2005.49.

[4] J. Ding, S. Li, and Z. Gu, "High-speed ECC processor over NIST primefields applied with Toom–Cook multiplication," in IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 3, pp. 1003–1016, Mar. 2019, doi:10.1109/TCSI.2018.2878598.

[5] R. Liu and S. Li, "A design and implementation of montgomery modular multiplier," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Sapporo, Japan, May 2019, pp. 1–4, doi: 10.1109/ISCAS.2019.8702684.

[6] W. Wang, X. Huang, N. Emmart, and C. Weems, "VLSI designof a large-number multiplier for fully homomorphic encryption," in IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 22, no. 9, pp. 1879–1887, Sep. 2014, doi: 10.1109/TVLSI.2013.2281786.

[7] S. Asif and Y. Kong, "Analysis of different architectures of counter based wallace multipliers," in Proc. 10th Int. Conf. Comput. Eng. Syst. (ICCES), Cairo, Egypt, Dec. 2015, pp. 139–144, doi: 10.1109/ICCES.2015.7393034.

[8] A. Najafi, B. Mazloom-nezhad, and A. Najafi, "Low-power and highspeed 4-2 compressor," in Proc. 36th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO), Opatija, Croatia, May 2013,pp. 66–69.

[9] A. Najafi, S. Timarchi, and A. Najafi, "High-speed energy-efficient 5:2 compressor," in Proc. 37th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO), Opatija, Croatia, May 2014, pp. 80–84, doi: 10.1109/MIPRO.2014.6859537.

[10] S. Asif and Y. Kong, "Design of an algorithmic wallace multiplier using high speed counters," in Proc. 10th Int. Conf. Comput. Eng. Syst. (ICCES), Cairo, Egypt, Dec. 2015, pp. 133–138, doi:10.1109/ICCES.2015.7393033.

[11] C. Fritz and A. T. Fam, "Fast binary counters based on symmetric stacking," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 10, pp. 2971–2975, Oct. 2017, doi: 10.1109/TVLSI.2017.2723475.

[12] Q. Jiang and S. Li, "A design of manually optimized (15, 4) parallel counter," in Proc. Int. Conf. Electron Devices SolidState Circuits (EDSSC), Hsinchu, Taiwan, Oct. 2017, pp. 1–2, doi:

10.1109/EDSSC.2017.8126527.

[13] M. H. Najafi, D. J. Lilja, M. D. Riedel, and K. Bazargan, "Low-cost sorting network circuits using unary processing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 8, pp. 1471–1480, Aug. 2018, doi: 10.1109/TVLSI.2018.2822300.

[14] D. E. Knuth, The Art of Computer Programming: Sorting and Searching, vol. 3. Reading, MA, USA: Addison-Wesley, 1973.

[15] M. Mehta, V. Parmar, and E. Swartzlander, "High-speed multiplier design using multi-input counter and compressor circuits," in Proc.10th IEEE Symp. Comput. Arithmetic, Grenoble, France, Jun. 1991,pp. 43–50, doi: 10.1109/ARITH.1991.145532.

[16] A. Fathi, B. Mashoufi, and S. Azizian, "Very fast, high-performance 5-2 and 7-2 compressors in CMOS process for rapid parallel accumulations," IEEE Trans. Very Large Scale Integr. (VLSI) Syst.,vol. 28, no. 6, pp. 1403–1412, Jun. 2020, doi: 10.1109/TVLSI.2020.2983458

[17] T. Satish and K. S. Pande, "Multiplier using NAND based compressors," in Proc. 3rd Int. Conf. Electron., Mater. Eng. Nano Technol. (IEMENTech), Kolkata, India, Aug. 2019, pp. 1–6, doi:10.1109/IEMENTech48150.2019.8981067.

[18] A. G. M. Strollo, E. Napoli, D. De Caro, N. Petra, and G. D. Meo, "Comparison and extension of approximate 4-2 compressors for low-power approximate multipliers," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 67, no. 9, pp. 3021–3034, Sep. 2020, doi: 10.1109/TCSI.2020.2988353.

[19] Z. Yang, J. Han, and F. Lombardi, "Approximate compressors for error resilient multiplier design," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFTS), Amherst, MA, USA, Oct. 2015, pp. 183–186, doi: 10.1109/DFT.2015.7315159.