Fast Binary Multiplier Based On Counter, Compressor And Parallel Prefix Adder

M.Karthik[1],S.Prabhasri[2],S.Sathya[3], ,C.Jothika[4].
[1]Assistant Professor,Department of Electronics And Communication Engineering,Cheran College Of Engineering,AnnaUniversity,Karur,Tamilnadu,India.
[2,3,4,5]U.G.Student,Department of Electronics And Communication Engineering,Cheran College Of Engineering,AnnaUniversity,Karur,Tamilnadu,India.
[1]mkarphd@gmail.com,[2]prabhasrikumar@gmail.com,[3]sathyasedhu2711@gmail.com,
[4]jokuttyjokutty44@gmail.com.

Abstract—The fundamental approach is adding up numerous operands, which is a common operation on DSP systems. The Wallace Tree structure, which functions as the circuit's bottleneck, adds all of the partial products of a basic multiplier circuit. Counters and compressors with high compression ratios are required to speed up the addition process. We apply a unique approach based on the sorting network and split logic in the proposed research, which employs fast saturated binary counters and compressors. When the counter's inputs are partitioned asymmetrically into two groups and fed into sorting networks, sequences that can only be represented by one-hot codes are reordered. Due to the smaller size of the sorting network needed, the (7,3) counter may be built using the aforementioned approach. In the LSB, compressors with a 4:2 reduction ratio are employed to compress just a piece of the product. Because of the space and energy savings, it is ideal for use in error-tolerant systems. For the final addition of multipliers, we suggest employing a kogge-stone adder-based parallel prefix adder to reduce critical path time even further. When compared to the present approximation multiplier, the suggested technique produces better area-delay and power-delay products. The fundamental approach is adding up numerous operands, which is a common operation on DSP systems. The Wallace Tree structure, which functions as the circuit's bottleneck, adds all of the partial products of a basic multiplier circuit. Counters and compressors with high compression ratios are required to speed up the addition process. We apply a unique approach based on the sorting network and split logic in the proposed research, which employs fast saturated binary counters and compressors. . When the counter's inputs are partitioned asymmetrically into two groups and fed into sorting networks, sequences that can only be represented by one-hot codes are reordered. Due to the smaller size of the sorting network needed, the (7,3) counter may be built using the aforementioned approach. In the LSB, compressors with a 4:2 reduction ratio are employed to compress just a piece of the product. Because of the space and energy savings, it is ideal for use in error-tolerant systems. For the final addition of multipliers, we suggest employing a kogge-stone adder-based parallel prefix adder to reduce critical path time even further. When compared to the present approximation multiplier, the suggested technique produces better area-delay and power-delay products.

Keywords—Binary counter, 4:2 compressor, counter, multiplier, one-hot code, sorting network.

I.INTRODUCTION

Any computer must be able to add several operands quickly and reliably. Multiplier circuits' speed and power efficiency are crucial to the overall performance of microprocessors. Multiplier circuits conduct filtering and convolution in a digital signal processor or arithmetic logic unit. When multiplying binary numbers or fixed-point values, some products must be assembled. The multiplier incurs a large amount of extra delay and requires more energy as a consequence of integrating these partial outputs. The fundamental approach is adding up numerous operands, which is a common operation on DSP systems. The Wallace Tree structure [1], whose performance is the bottleneck, is used to sum all the partial products of a basic multiplier circuit. In public-key cryptography, such as RSA and elliptic curve encryption, a large number multiplier based on the Toom-Cook [4] or Karatsuba approach [3] is used to generate modular multipliers (ECC). Several studies have looked at these two ways, with some even putting them into hardware. The summation of numerous operands is discussed in different areas of the circuit in the publications. In fully homomorphic encryption (FHE), a post-quantum cryptosystem that offers high security in cloud computing, a number theoretic transform (NTT) [6] is urgently needed to speed huge number and polynomial multiplication. The fundamental processing unit in certain implementations of the high radix [6] NTT consists of the aggregation of numerous operands..

The Wallace tree structure [1] and its version, the shortened Wallace tree [2], are the most often employed to sum multiple operands. To speed up the summing, these approaches employ complete adders as (3,2) counters, resulting

in logarithmic time complexity. Another term for this structure is a carry–save structure. Several articles since then, notably [7]–[12], have focused on improving the framework in order to speed up the summing process.

Data mining and database management [20, 21], automated teller machines and communication switching [1, 14], scientific computing [13], artificial intelligence and robotics [7], video [11], and signal processing [12] all need classification. Hardware implementations of sorting are prevalent in high-performance applications, frequently in the form of application-specific integrated circuits or field-programmable gate arrays [12]. The hardware sorting devices may be configured in a variety of ways to meet a variety of demands. Depending on the application, the number of inputs might vary from nine to tens of thousands when working with photos. Any combination of binary values, integers, and floating-point numbers with a precision of 4 bits or more up to 256 bits may be used as data.

Efficiency and low power consumption are major goals when building devices. The quantity of space on a chip is often restricted. Because leakage current grows exponentially with temperature, it becomes more vital to maintain low chip temperatures as manufacturing processes improve. It is critical to consume as little energy as possible. A primary priority is the development of low-cost, low-energy sorting systems. A network of compare and swap (CAS) nodes, often known as a Batcher (or bitonic) network, is the most common technique. Pipelining is a breeze in this kind of network. Hardware-based solutions outperform sequential software-based solutions due to their parallel nature. The total number of CAS blocks and the price of each individual CAS block have an impact on hardware costs and power usage.

II.SORTING NETWORK

The sorting network [14], a high-performance parallel hardware network, is used to sort data. If a sorting network can sort a collection of data whose components are all 1-bit integers, then it can sort any set of numbers, according to the well-known 0,1 principle [14]. It is utilised only for sorting data that consists of a single bit in this investigation [13]. Figure 3.1 [14] depicts a typical three- and four-way sorting network. Each vertical line represents a distinct sorter that accepts and outputs one-bit information as input and output. The sorter processes the bigger inputs first, followed by the smaller ones. Figure 1 depicts the input of a four-way sorting network (4 SN) as [0, 1, 1, 1], while the input of a three-way sorting network (3 SN) is shown as [0, 1, 1]. (3) SN After being processed by a three-tiered sorter, the input sequences for both 4 SN and 3 SN are rearranged with the greater number at the top and the smaller number at the bottom.

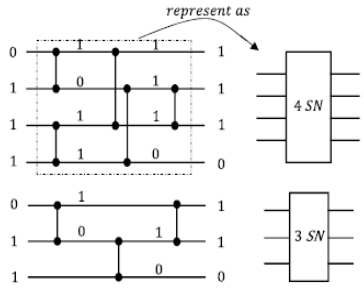

Fig. 1. Three- and four-way sorting networks.

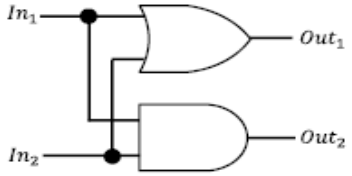

Fig. 2. Two-input binary sorter.

As previously stated, the sorter rearranges two inputs based on numerical values. Figure 2 shows a logical circuit that might be used to sort two sets of 1-bit data effectively. In a single layer of a sorter, two-input basic logic gates are employed, while three- and four-input basic logic gates are utilised in a network to sort data.

III.(7,3) COUNTER

We'll begin by examining the primary point of comparison, the design in [11]. Fritz and Fam [11] presented fast (6,3) counters with symmetric stacking structures, and a saturated (7,3) counter was then built from them. The quickest of the seven counter designs (7,3) is achieved by adding a MUX to the essential route without optimising it, although it has poor delay performance. We suggest an immediate creation of a (7,3) counter using this technique to answer the issue in [11]. Instead of one symmetrically stacked set of sorting networks, as shown in Fig. 3.1, we start with two sets of networks stacked in opposing orientations. We generate three special Boolean equations [see (2),

(13) and (15)] by constructing one-hot code sequences, considerably reducing the complexity of the Boolean expressions associated with the outputs.

To begin, all "1"s will be at the top of the series if there are any, while all "0"s will be at the bottom if there are any, as illustrated in Fig. 3. If the two "1"s and "0"s exist at all, they must meet somewhere in the newly ordered sequence. If the sequence contains just ones or zeroes, we may deal with it by adding a fixed one-bit "1" at the start and a fixed one-bit "0" at the conclusion of the reordered sequence to ensure that the 0,1-junction always resolves to the exit state. Second, the total number of ones and zeros in the original and reordered sequences remains unchanged (the inputs of two sorting networks). We disregard the padded "1" since it cannot be erased and hence has no influence on the overall number of "1"s in the padded sequence.

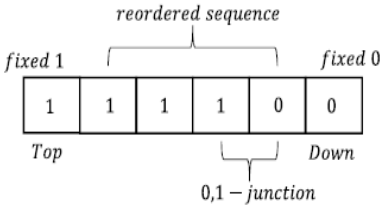

Fig. 3. Definition of a sequence.

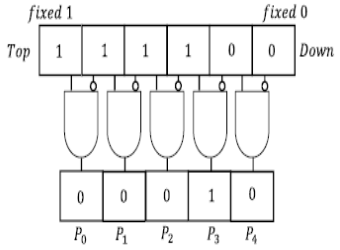

Fig. 4. One-hot code generation circuit.

Both three-way and four-way sorting networks need three layers of binary sorters, as shown in Fig. 1. (the two binary sorters on the same layer in fourway sorting network can be calculated in parallel). Each layer of binary sorters is made up of a single two-input logical gate, as illustrated in Figure 2. This shows that the three-way and four-way sorting networks take about the same amount of time to complete. The seven inputs of a (7,3) counter were split in half as a result of this. One portion has four bits, whereas the other has just three.

As seen in Fig. 3, the rearranged sequence can only be adequately represented by the 0,1-junction, which promises an extended fixed "0" and "1." The 1,0 at the 0,1-junction must be read from left to right. As a result, the four-way sorting network will be used as an example once again, with a final shape similar to that shown in Fig. 4. To create the new sequence P0P4, this framework employs a Boolean expression (AB). Because the altered and extended sequence includes just one 0,1-junction, sequence P0–P4 only contains one "1." If and only if the values P0|P1|P2|P3|P4 = 1, the range P0P4 is a one-hot code. If the sequence's components (P0–P4) are divided into two groups at random, with P0, P2, and P4 in group 1 and P1, and P3 in group 2, there is precisely one "1" in the sequence.

$$P_0|P_2|P_4 = \overline{P_3|P_4}. \qquad (2)$$

This rule applies to all results of random separation. From the output sequence of a three-way sorting network, we utilise the same method to construct the one-hot code sequence Q0–Q3. The preceding rule applies to this sequence as well. The two sequences we have currently are P and Q. P0 = 1 indicates that the four-way sorting network's input sequence contains no "1," P1 = 1 indicates one "1," Pi = 1 indicates I "1"s, and Q indicates the sequence. The following are some symbol conventions. The outputs of the (7,3) counter are C2, C1, and S, with C2 having the most significant weight and S having the least.

The output of a four-way sorting network, which comprises H1–H4 from left to right in Fig. 3.3, is denoted by the sequence H. From left to right, sequence I represents the output of a three-way sorting network, which contains I1–I3. When C2 = 1, the input sequence of the (7,3) counter comprises at least four "1s," as shown in Table I. P4 = 1 indicates that sequence H contains four "1s" (also in the input sequence of 4 SN, as the sorting network has no effect on the total number of "1s"), but Q0 = 1 shows that sequence I has no "1s." P4&Q0 = 1 shows that there are 4 + 0 = 4 "1s" in the input 7 bits. As a result of this method of representation, C2 = 1 when the sum of the subscripts of P and Q is no less than 4.

C2=P4&(Q0|Q1|Q2|Q3))|(P3&(Q1|Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q2|Q3))|(P2&(Q (P1&Q3). (3) Notice that the sequence Q satisfies using the same approach as in (2).

$$Q_0|Q_1|Q_2|Q_3 = 1 \quad (4)$$

$$Q_1|Q_2|Q_3 = \overline{Q_0}. \quad (5)$$

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

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

C1 = 1 because the total of the subscripts of the sequences P and Q equals 2, 3, 6, and 7. (see Table I). As a result, we get the following equation:

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

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:

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

P2|P3, P2|P3, and P1|P2, P1|P2, build two multichannel selection structures in (10). C1 may be determined quickly using the circuit depicted in Fig. 5. S may be readily calculated using the following equation: (P1|P3) (Q1|Q3) (11) S = (P1|P3) (Q1|Q3) (11) S = (P1|P3) (Q1|Q3) (11) S = (P1|P3) (

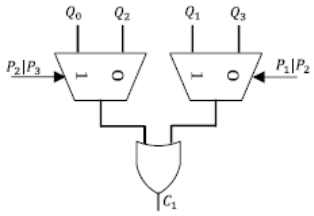

Fig. 5. C1 generating circuit.

We have got two sequences H1–H4 and I1–I3. Here, we extend sequence H1–H4 by H0 (denotes the fixed "1" in Fig. 3) and H5 (denotes the fixed "0"). Do the same for sequence I. I0 denotes the fixed "1," and I4 denotes the fixed "0." Thus, we have the following equation:

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

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

In addition, we notice that, when subsequences selected from the sequence Q or P are given, if their subscripts are successive (P1, P2, and P3 for example), the result of "OR" them up can be easily expressed by sequence I or H (P1|P2|P3 =H1&H4 for example). Thus, the Boolean equation (12) is generalized as (13).

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

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

Based on this, (10) is simplified to (14), which can also be calculated from the circuit in Fig. 6

$$C_1 = (Q_0\&(P_2|P_3))|(Q_1\&(P_1|P_2))|(Q_2\&(\overline{P_2|P_3})) |(Q_3\&(\overline{P_1|P_2})) = (Q_0\&(H_2|\overline{H_4}))|(Q_1\&(H_1|\overline{H_5}))|\left(Q_2\&\left(\overline{H_2|H_4}\right)\right) \left|\left(Q_3\&\left(\overline{H_1|H_5}\right)\right)\right. \quad (14)$$

There is another trick for sequences H and I. Because H0–H5 are all in order, this means that, if Hi = 1 (i= 0, 1, 2, 3, 4, 5), then, for every j <i,Hj= 1 always holds and so is the sequence Q. Then, we get the following equation:

$$I_i = I_i|I_{i+j}, \quad (i = 0, 1, 2, 3; j \ge 0; i + j \le 4)$$

$$H_i = H_i|H_{i+j}, \quad (i = 0, 1, 2, 3, 4; j \ge 0; i + j \le 5). \quad (15)$$

Note that H0 = I0 = 1 and H5 = I4 = 0 always hold; we can simplify (3) as (using a trick:A|(A&B) = A|B)
C2=(P4&(Q0|Q1|Q2|Q3))|(P3&(Q1|Q2|Q3))|(P2&(Q2|Q3))|(P1&Q3)
= H4|(H3&I1)|(H1&I3)|(H2&I2). (16)

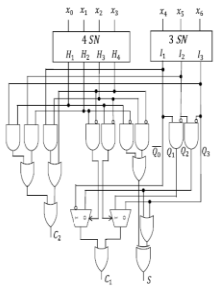

Fig. 6. Overall (7,3) counter circuit.

Figure 6 depicts the whole structure. The pathways from H and I sequences to C2, C1, and S are virtually indistinguishable. This property, which is derived from (3) and (4), adds to the parallelism of the circuit (7). The area of the suggested design, on the other hand, will not expand as a result of the parallelism, as demonstrated in the following sections. In reality, it is decreasing. Boolean statement simplifications at stages (2), (13) and (14) are among the reasons for the lower footprint (15).

IV.(4:2) COMPRESSORS

A (4:2) compressor, as shown in Fig. 9, serves the same logical goal. We also use sorting networks to build a high-speed (4:2) compressor. We noticed that the final step of the four-way sorting network (Fig. 1) only sorts the two data in the centre, meaning that the data at the top and bottom are the highest and lowest of the four data, respectively.

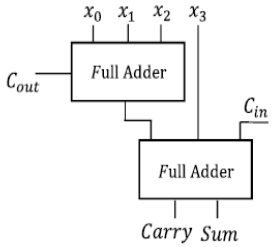

Fig. 9. (4:2) compressor combined by full adders.

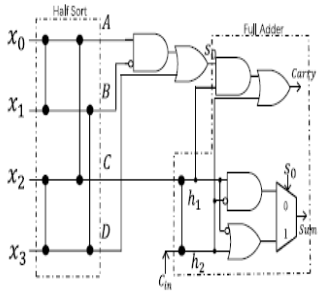

Fig. 10. Proposed exact (4:2) compressor.

The "Half Sort" results are labelled A, B, C, and D in Figure 10, which is titled "Half Sort." Because A and D are the largest and least data, the sequence [A, B, D] is already sorted completely. A modified "Full Adder" is used to calculate the sum of s0, Cin, and C. (as illustrated in Fig. 11). Equation describes the "Full Adder" (30).

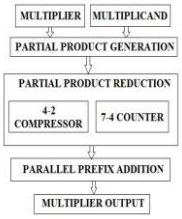

Fig. 11. Block diagram of proposed multiplier.

A novel counter and compressor architecture based on a sorting network is presented, as well as the construction of an 8x8 multiplier. Using a parallel prefix adder also reduces the latency of the final addition. In comparison to existing options, the suggested approach uses less resources and has a lower latency. Because it employs a sorting network, the (7,3) counter is more adaptable than previous designs. In addition, exact/approximate (4:2) compressors, which are built using a sorting network, are presented. When utilised in approximation applications, they perform better in ADP and PDP.

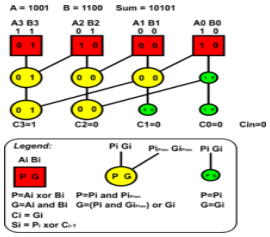

Fig.12 Kogge stone adder

V. EXPERIMENTAL RESULTS

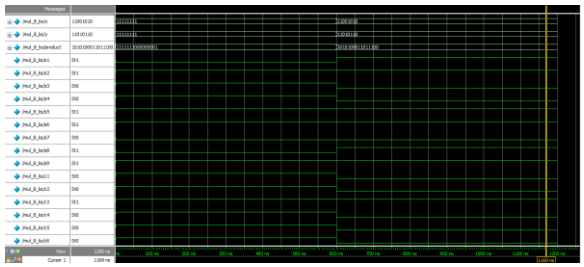

Fig.13 Simulation result of Propose Multiplier

The suggested multiplier's simulation result is shown in Figure 13. The counter and compressor are used to implement multipliers. It employs a kogge stone parallel prefix adder for the final addition to increase latency and power efficiency even further. Table I lists some of the parameters for the proposed multi-level compressor and counter-based multiplier, including area, power, and latency. Several kinds of compressors and counters are used in the suggested design to optimise space, latency, and power consumption.

Table.I. Comparison Results of Multiplier

Parameters	Area (Gate Count)	Power (mW)	Delay(ns)
Existing Multiplier	1260	181.97	32.340
Proposed Multiplier	1272	177.87	29.780

VI. CONCLUSION

A revolutionary counter and compressor design is presented with the use of a sorting network and the building of an 8x8 multiplier. In addition, a parallel prefix adder is employed to reduce the final addition's total latency. The proposed technique has a lower latency and uses less resources than current methods. The (7,3) counter is more versatile than rival systems because it uses a sorting network. In addition, to generate exact/approximate (4:2) compressors, a sorting network-based technique is applied. They perform well in ADP and PDP when used for approximation purposes.

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 prime fields 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] ParameswariSubbian, Chitra Chinnasamy and KannadhasanSuriyan, Textile UWB Antenna Performance for Healthcare Monitoring System, Frequenz, De Gruyter, 15 March 2022, https://doi.org/10.1515/freq-2021-0227
[6] S.Kannadhasan, R.Nagarajan and R.Banupriya, Performance Improvement of an ultra wide band antenna using textile material with a PIN diode, Textile Research Journal, DOI: 10.1177/00405175221089690
journals.sagepub.com/home/trj
[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 high speed 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 Solid-
State 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.