Tuijin Jishu/Journal of Propulsion Technology
ISSN:1001-4055
Vol. 44 No.3 (2023)

High Speed Compressor Design Using Reversible Logic for Approximate Computing

[1]P. Pushpalatha, [2]K. Lakshmi Prasanthi
[1]Assistant professor, Department of ECE, UCEK, JNTUK, Kakinada, India,
[2]Department of ECE, UCEK, JNTUK, Kakinada, India,

Abstract*:* The Dual stage 4:2 compressor proposed in this study has a new architecture and an inexact Baugh-Wooley Wallace tree multiplier with reversible logic optimization for realization. The suggested Dual stage 4:2 compressor and Baugh-Wooley Wallace tree multiplier's effectiveness is evaluated using the scales of Gate Count (GC), Quantum Cost (QC), Garbage Output (GO), and Ancilla Input (AI). This paper implements an 8 8 Baugh-Wooley Wallace tree multiplier. The proposed multiplier is evaluated using the accuracy metrics MED and MRED, and it is discovered to be the least accurate among existing imprecise compressor-based multiplier designs. There are two uses for the suggested multiplier. Convolutional Neural Networks (CNN) and one-level decomposition of images using a rationalized db6 wavelet filter bank are two examples of image processing.

Keywords: Reversible logic, In-exact 4:2 compressor, Dual stage 4:2 compressor, Baugh-wooley method.

1. Introduction

Convolutional units are used in the majority of signal processing applications because they are computationally demanding and key operational units. Convolutional units mostly use adders and multipliers, with multipliers considerably contributing to the area, delay, and power. Convolutional neural networks, multimedia, and other real-world applications utilizing computational units are in significant need of high speed multipliers that are optimized for area and power. The three stages of a multiplication process can be roughly categorized. Partially producing a product, partially accumulating a product, and finally, partially adding a product .Research has been done to improve the partial product accumulation stage in order to produce the final two terms for stage three using parallel and high-speed accumulation algorithms. This stage contributes to the total delay. Dadda and Wallace's algorithms made a substantial contribution to the accumulation stage's achievement of delay-optimized architectures. By using compressors rather than full adders and half adders, accumulation phase delay is further decreased. The 4:2 compressor topology is the most popular compressor topology because, in contrast to other topologies like the 5:3, 7:2, and others, it can produce regularly structured systems. Since they are in such high demand for achieving space and power optimal designs for error tolerant applications like multimedia processing, neural networks, signal processing, etc., multipliers are currently being investigated in the context of approximation. Such improvements in multiplier realizations based on CMOS, FPGA, pass transistor, and FinFET are currently being studied. Four 4:2 compressors that may operate in either precise or approximate modes have been presented by Akbari. Even when approximation is introduced, the extra hardware required to transiti.3on between exact and approximate modes results in a space overhead. Esposito suggested an XOR-less design with a high ER. However, its high ER renders it ineffective in applications involving image processing. Reddy and Edavoor suggested multiplexer-based compressors. These designs are better suited for implementation at the standard gate level. Although the area/gate count is high, Gorantla and Deepa have proposed 4:2 compressors with optimal delay. Strollo has suggested a 4:2 compressor architecture that modifies the way circuits are stacked. The designs listed above have been tailored for CMOS-based applications. We provide 4:2 compressors designed for FPGA-based architecture.

Ton and Lee Chang have suggested an approximate 4:2 compressor that is optimized for pass transistor-based implementation, a small 4:2 compressor for FPGAs with minimal accuracy losses and greater electrical performance. Zakian and Ashli proposed a FinFET-based implementation for an approximate 4:2 compressor. The Moore's law-imposed physical restrictions apply to all of the aforementioned designs. For each bit lost in

irreversible computations, KTln2 joules of energy are lost. In more advanced technology, this dissipation was negligible.

Therefore, it is necessary to investigate innovative ways to decrease power dissipation. An emerging field of study to deal with this problem is the reversible approach in circuit and system design. According to Bennett's comparison of typical irreversible and reversible systems, the power dissipation of a circuit or system is lowered to zero or to insignificant levels when a reversible model is built for it. By projecting GO, GC, and QC, the authors have shown how to create adder circuits and have measured and compared the effectiveness of the suggested adder. Both the first and second constructions lowered GO and GC, respectively. The comparative analysis demonstrates that, when compared to that which has the least reversible logic realization parameters in the literature, the suggested 8-bit barrel shifter is capable of reducing GC, GO, and QC. A Min-Max algebra based synthesis technique was developed by Khan and Rice to produce reversible circuits for ternary logic functions. The ternary multiple-controlled unary gates are used to implement this circuit mapping. In terms of AI and QC, the suggested method performs better than the Ternary Galois Field Sum of Products (TGFSOP)-based method currently in use.

This work suggests a non-exact Baugh-Wooley Wallace tree multiplier to answer the need to actualize low power computational units for error tolerant applications. This study introduces a unique architecture for an approximate 4:2 compressor that is compatible with implementation using reversible logic by optimizing reversible logic realization parameters GC, QC, GO, and AI. This architecture achieves approximation in the multiplier architecture. The effectiveness of the suggested inexact compressor-based Baugh-Wooley Wallace tree multiplier in CNN-based applications is confirmed. For the experimental analysis, a one-level decomposition employing a rationalized db6 wavelet filter bank and picture smoothing applications is carried out, and the effectiveness is calculated in terms of SSIM. In CNN-based applications, the model's accuracy is tested to assess its effectiveness.

2. Objectives

A reversible logic gate is a specialized n-input n-output logic device characterized by its one-to-one mapping between inputs and outputs. This unique property enables the precise determination of outputs from given inputs while also allowing for the complete recovery of inputs from the outputs, ensuring full reversibility of information. In the synthesis of reversible circuits, it is important to note that direct fan-out, or the one-to-many concept common in traditional logic circuits, is not permissible as it contradicts the reversibility principle. However, fan-out functionality in reversible circuits is achieved through the incorporation of additional gates, which serve to duplicate output signals while maintaining the one-to-one mapping. The optimal design of a reversible circuit revolves around the fundamental objective of using the minimum number of reversible logic gates, thus minimizing circuit complexity and enhancing performance. The field of reversible circuit design encompasses a multitude of parameters that play pivotal roles in determining both the complexity and performance of these circuits.

The suggested Dual stage 4:2 compressor and Baugh-Wooley Wallace tree multiplier's effectiveness is evaluated using the scales of Gate Count (GC), Quantum Cost (QC), Garbage Output (GO), and Ancilla Input (AI). This paper implements an 8 8 Baugh-Wooley Wallace tree multiplier. The proposed multiplier is evaluated using the accuracy metrics MED and MRED, and it is discovered to be the least accurate among existing imprecise compressor-based multiplier designs.

3. Methods

In AI and DSP applications, multiplication is undoubtedly a performance-determining process. For these applications to require high speed parallel operations with acceptable levels of precision, high speed multiplier designs are required. With the introduction of approximation in multipliers, computations can be completed more quickly, with less hardware complexity, delay, and power consumption, and with precision at desired levels. Due to the propagation delay in adder networks, partial product summation is the speed-limiting process in multiplication. Compressors are used to shorten the propagation delay. At every level, compressors

simultaneously calculate the total and carry. In the following step, the resultant carry is added with a more significant sum bit.

This study suggests alternative multiplier architecture with more than three stages of cascaded compressors to maximize the hardware utilization of the suggested design. One XOR, one AND, and two OR gates are needed in the high speed area-efficient compressor architecture), in addition to the MUX. In order to implement OR and AND gates, CMOS logic requires 8 transistors per gate. This research suggests a design with NAND and NOR gates to lower the transistor count. Even if the SUM and CARRY produced by the updated architecture are not exactly the same as those produced by the suggested 4:2 compressor architecture, the mistake is eliminated thanks to the compressor's cascading in multiples of 2.

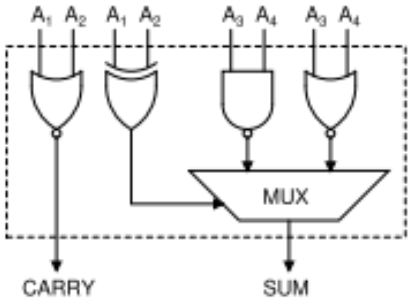

Figure 1: Basic building b lock for proposed modified Dual-stage 4:2 compressor.

A compressor can be approximated by reducing the output count to 2. COUT is eliminated as an approximation [25]. Only when the input combination is "1111" does this result in an error. The CARRY and SUM are set to '11' and an error of 1 is introduced when the input bits are '1111'.In the majority of MAC modules, multiplier components take up 46% of the chip space. In low-power VLSI system design, an energy-efficient multiplier design can therefore be crucial. The higher order multiplication operations are precisely carried out while just the lowest significant portion of the result is approximated in the proposed strategy to control error significance. Thus, we propose using approximate array multipliers with 9-bit of the result estimated out of 16-bit in the approximate MAC units as shown in Figure based on quality evaluation of several approximate multipliers with varying numbers of approximated bits. Greater gains in space, power, and delay reduction come with approximating beyond 9 bits. The higher order multiplication operations are precisely carried out while just the lowest significant portion of the result is approximated in the proposed strategy to control error significance. Thus, we propose using approximate array multipliers with 9-bit of the result estimated out of 16-bit in the approximate MAC units as shown in Figure, based on quality evaluation of several approximation multipliers with a variable number of approximated bits. Greater gains in space, power, and delay reduction come with approximating beyond 9 bits. However, there has been a noticeable decline in quality. In order to utilize multiplier designs and obtain greater area, power, and speed, a mac unit is created employing four approximate Adder designs. Four parts make up the proposed MAC unit: a multiplier, an adder, an accumulator, and a controller. Therefore, it is possible to approximate any of these components. In order to attain energy efficiency, we focus on multiplier blocks in this study. A crucial arithmetic module in the digital signal processing (DSP) system is the multiplier. It mostly affects speed and power consumption; hence effective multipliers are urgently needed. By lowering area, power, and delay, approximate computing has introduced a special dimension to the field of digital design. The high speed, fault tolerance, and power economy of effective approximate multipliers are driving up demand for them. This study uses proposed approximation compressors to create an 8-bit multiplier.

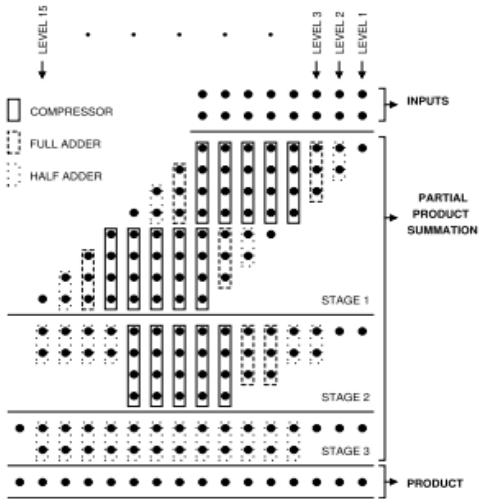

Figure 2: 8 × 8 approximates multipliers.

Parallel Prefix adder: The Ladner-Fischer adder is adaptable to speed up binary addition, and the high performance of arithmetic operations is supported by a structure that resembles a tree. Device development is aided by research on the motivations behind binary operations. Because they increase the speed of microprocessor-based applications like mobile DSP and telecommunication, field programmable gate arrays, or FPGAs, have become increasingly popular in recent years. Three steps make up the Ladner-Fischer adder's construction. Pre-processing, carry generation, and post-processing are the three stages.

Pre-Processing Stage: Generate and propagate are taken from each pair of inputs during the pre-processing phase. The propagate performs a "XOR" operation on the input bits whereas the create does a "AND" operation. In the equations 1 and 2 below, the terms propagate (Pi) and generate (Gi) are displayed.

$$Pi=Ai \text{ XOR } Bi \text{ ------ } (1)$$

$$Gi=Ai \text{ AND } Bi \text{ ------ } (2)$$

Carry Generation Stage: This stage is known as carry generate (Cg), when carry is created for each bit. The final cell present in each bit operation gives carry, however carry propagates and generates for subsequent operations. The last bit carry will assist in producing the total of the next bits while keeping track of the last bit. The following equations 3 and 4 provide the carry generates and carry propagate.

$$Cp=P1 \text{ AND } P0 \text{ ---------- } (3)$$

$$Cg=G1 \text{ OR } (P1 \text{ AND } G0) \text{ ------ } (4)$$

The black cell in equations 3 and 4 represents the carry propagates Cp and Cg mentioned previously, whereas the gray cell in equation 5 represents the carry generation that is described below. The final cell included in every bit operation gives carry, but the carry propagate is generated for the subsequent operation. The last bit carry will assist in producing the total of the next bits while keeping track of the last bit. The carry generated in the following equations 5 is used for the subsequent bit sum operation.

$$Cg=G1 \text{ OR } (P1 \text{ AND } G0) \text{ ------ } (5)$$

Post-Processing Stage: The carry of a first bit is XORed with the next bit of propagates in the Ladner-Fischer adder's final step, which is depicted in equation 6. The result is then presented as a total.

$$Si=Pi \text{ XOR } Ci\text{-}1 \text{ ------- } (6)$$

It is utilized for two sixteen-bit addition operations, and the final sum is produced by post-processing each bit carried through a propagate stage. Pre-processing is applied to the first input bits before they are produced,

propagated, and generated. These generates and propagates proceed through the carry generation step, which results in carry generates and carry propagates. These then go through the post-processing stage, which results in the final sum. The Ladner-Fischer adder's step-by-step procedure is depicted in Fig. The fastest adder that focuses on gate level logic is the Ladner-Fischer adder, which has a topology that resembles a tree for high speed arithmetic operations. It uses fewer gates in its designs. Therefore, it reduces the amount of time and memory consumed in this design.

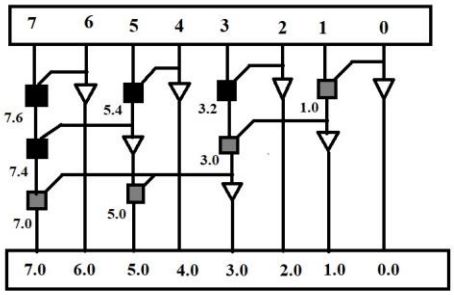

Figure 3: Ladner-Fischer Adder

4. Results

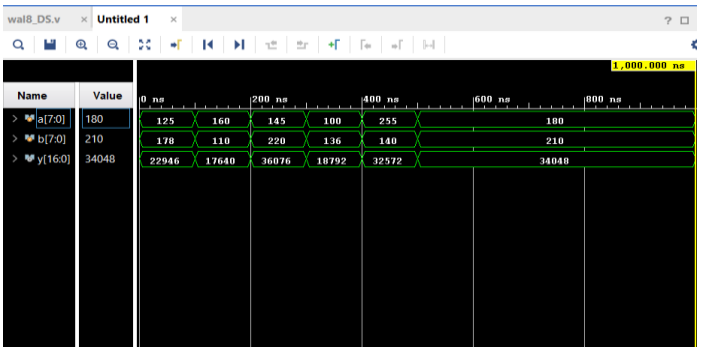

Figure 1: Simulation of the Dual stage 4:2 compressor.

Comparison of the Dual stage 4:2 compressor and the Baugh Wooley Wallace tree multiplier.

Types	Area (LUT'S)	Delay(ns)
A	97	8.599
B	88	7.949

A= Baugh Woolley Wallace tree multiplier.

B= Dual stage 4:2 compressor.

5. Discussion

This study proposes a 4:2 Dual stage 4:2 compressors modified with complementary gate that, when compared to reversible logic-based realization of architectures, has the lowest metrics for reversible logic implementation. We can decrease hardware complexity and time by using complementary gate-based compressor and parallel

prefix adders. The proposed design is able to achieve the best optimization in scales of reversible logic realization parameters, according to the experimental analysis, and it is able to obtain accuracy metrics that are comparable to those of the precise Baugh-Wooley Wallace tree multiplier.

Refrences

[1] L. Dada, ‘‘some schemes for parallel multipliers,’’ Alta Frequenza, vol. 34, no. 5, pp. 349–356, Mar. 1965.

[2] C. S. Wallace, ‘‘A suggestion for a fast multiplier,’’ IEEE Trans. Electron. Compute. Vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[3] Z. Wang, G. A. Jillian, and W. C. Miller, ‘‘A new design technique for column compression multipliers,’’ IEEE Trans. Compute., vol. 44, no. 8, pp. 962–970, Aug. 1995.

[4] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, ‘‘Dual-quality 4:2 compressors for utilizing in dynamic accuracy configurable multipliers,’’ IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 4, pp. 1352–1361, Apr. 2017.

[5] D. Esposito, A. G. M. Strollo, E. Napoli, D. De Caro, and N. Petra, ‘‘Approximate multipliers based on new approximate compressors,’’ IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 65, no. 12, pp. 4169–4182, Dec. 2018.

[6] K. Manikantta Reddy, M. H. Vasantha, Y. B. Nithin Kumar, and D. Dwivedi, ‘‘Design and analysis of multiplier using approximate 4- 2 compressor,’’ AEU Int. J. Electron. Commun., vol. 107, pp. 89–97, Jul. 2019.

[7] P. J. Edavoor, S. Raveendran, and A. D. Rahulkar, ‘‘Approximate multiplier design using novel dual-stage 4:2 compressors,’’ IEEE Access, vol. 8, pp. 48337–48351, 2020.

[8] Gorantla and P. Deepa, ‘‘Design of approximate compressors for multiplication,’’ ACM J. Emerg. Technol. Comput. Syst., vol. 13, no. 3, p. 44, May 2017.

[9] S. Nowrin, L. Jamal, and H. M. H. Babu, ‘‘Design of an optimized reversible bidirectional barrel shifter,’’ in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2016, pp. 730–733.

[10] Hashmi and H. M. H. Babu, ‘‘An efficient design of a reversible barrel shifter,’’ in Proc. 23rd Int. Conf. VLSI Design, Jan. 2010, pp. 93–98.

[11] M. Khan and J. E. Rice, ‘‘Synthesis of reversible logic functions using ternary max-min algebra,’’ in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2016, pp. 1674–1677.

[12] M. H. A. Khan, ‘‘Design of reversible synchronous sequential circuits using pseudo reed-muller expressions,’’ IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 22, no. 11, pp. 2278–2286, Nov. 2014.

[13] H. Thapliyal and N. Ranganathan, ‘‘Design of reversible sequential circuits optimizing quantum cost, delay and garbage outputs,’’ ACM J. Emerg. Technol. Comput. Syst., vol. 6, no. 4, pp. 14:1–14:35, 2008.

[14] S. Molahosseini, A. Asadpoor, A. A. E. Zarandi, and L. Sousa, ‘‘towards efficient modular adders based on reversible circuits,’’ in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2018, pp. 1–5.

[15] H. M. Gaur, A. K. Singh, and U. Ghanekar, ‘‘Testable design of reversible circuits using parity preserving gates,’’ IEEE Des. Test. Comput., vol. 35, no. 4, pp. 56–64, Aug. 2018.

[16] H. M. Gaur, A. K. Singh, and U. Ghanekar, ‘‘Design of reversible arithmetic logic unit with built-in testability,’’ IEEE Des. Test. Comput., vol. 36, no. 5, pp. 54–61, Oct. 2019.

[17] K. Datta, I. Sengupta, and H. Rahaman, ‘‘A post-synthesis optimization technique for reversible circuits exploiting negative control lines,’’ IEEE Trans. Comput., vol. 64, no. 4, pp. 1208–1214, Apr. 2015.

[18] S. Raveendran, P. J. Edavoor, N. K. Y. Balachandra, and V. M. Harishchandra, ‘‘Design and implementation of image kernels using reversible logic gates,’’ IET Image Process., vol. 14, no. 16, pp. 4110–4121, Dec. 2020.

[19] S. Raveendran, P. J. Edavoor, N. Y. B. Kumar, and M. H. Vasantha, ‘‘an approximate low-power lifting scheme using reversible logic,’’ IEEE Access, vol. 8, pp. 183367–183377, 2020.

[20] R. Wille, D. Grosse, L. Teuber, G. W. Dueck, and R. Drechsler, ‘‘RevLib: An online resource for reversible functions and reversible circuits,’’ in Proc. Int. Symp. Multi-Valued Log., May 2008, pp. 220–225.

[21] R. Baugh and B. A. Wooley, ‘‘A two’s complement parallel array multiplication algorithm,’’ IEEE Trans. Comput., vol. C-22, no. 12, pp. 1045–1047, Dec. 1973.

[22] K. Hwang, Computer Arithmetic: Principles, Architecture, and Design. Hoboken, NJ, USA: Wiley, 1979.

[23] F. Cavanagh, Digital Computer Arithmetic: Design and Implementation. New York, NY, USA: McGraw-Hill, 1984.