International Journal of Gender, Science and Technology
ISSN: 2040-0748
UGC Care Group I Journal.
Vol-13 Issue-02 Dec 2024
https://ijgst.com.2024.v13.i2.pp957-965

Implementation of a Novel Approximate (8; 2) Compressor for Applications in Image Processing on FPGA

Dr.P.Prasanna Murali Krishna[(1)], Mrs.N.Swarupa Rani[(2)], Devandla Subbu[(3),] Modi Vishnu[(4),] Balu Venkateswara Reddy[(5),] Cherukupalli Sai Charan Dy[(6)]

[1,2] *Krishna Chaithanya Institute Of Technology & Sciences, Ece Department, Markapur, Andhra Pradesh.*

[3,4,5,6,] *Krishna Chaithanya Institute Of Technology & Sciences, UG Student-ECE, Markapur, Andhra Pradesh.*

Abstract In this project, one technique to enhance performance in a variety of error-tolerant applications, such processing images, and videos, is approximate computing. A component of their computing unit that frequently uses a lot of resources is the multiplier. This study compares the quality, power consumption, delay, and circuit area of an approximate (8; 2) compressor against other approximate compressors already on the market. In 8 × 8 and 16 × 16 multipliers, the approximate compressor that is suggested is utilized. To demonstrate the quality of the suggested compressor, two images were multiplied by the suggested 8 × 8 approximation multiplier using Xilinx 2018.1 tools. The qualitative parameters of SSIM and PSNR were then measured, and satisfactory results were obtained. Furthermore, the suggested 8 × 8 multiplier circuit's allowed error rate is indicated by the MED and NED accuracy parameters. Finally, using a Synopsys Design Compiler, we synthesized the approximate compressor and multiplier designs that were suggested. In comparison to other comparable current approximate multipliers, the suggested 16 × 16 multiplier can enhance latency, area, and Power latency Products by 5%, 17%, and 8%, respectively.

Keywords: Copressors 8x2,4x2,5G , FPGA, Verilog HDL, XOR and MUXes.

I.INTRODUCTION

There are specific applications for error tolerance, such as image, video and audio processing, which are based on the sensitivity of the human [1–9]. Most of these applications need to be used in real-time and must provide goodquality output in the shortest possible time. On the other hand, the design of digital circuits is no longer based on Moore's Law, and the shrinking of digital circuits cannot be used as a general solution to increase the efficiency or reduce the power consumption of circuits [10]. As a result, in fault-tolerant applications, approximate computing can be used to reduce computational complexity, followed by increasing speed and reducing power consumption [7–9,11,12]. In fault-tolerant calculations, a good performance and energy gain can be generated till an acceptable output result is obtained. Approximate computing can be applied to a variety of operations. Of course, it is economical to use it when the computational volume is high, such as consecutive addition or multiplication operations in digital signal processing. The multiplication operation is more complex than the addition operation and, therefore, requires more time, power consumption and area. For this reason, the design of a high-speed multiplier has become a necessity [13]. In some designs, they have been made to provide a complete multiplier circuit with 100% accuracy and low power consumption or high speed, and in some other designs, they have been made to approach this goal by applying

approximate computing techniques [12,14–20].

multiplication operation consists of three stages: Partial Product Generation (PPG), PP reduction (PPR), and the final sum. The partial product reduction circuit affects the impactor performance in terms of speed and power consumption more than other parts. The depth reduction process of partial products is done in either of two ways: DADDA and Wallace tree. Carry Save Adders (CSAs) or compressors are commonly used to reduce the partial product depth [21].

A compressor with a general view (m; 2) is a combinational circuit in which m-input with the same value adds and produces two outputs [21]. For example, the (5; 2) compressor has 5 inputs and 2 outputs. Figure 1 shows the general structure of the compressor. In the compressor, the weights of inputs and Sum output are the same, and the weights of Carries and Carry Out outputs are one order higher. In circuits where compressors are used horizontally and side by side to reduce the depth, in addition to the main inputs and outputs, the compressor has additional transfer outputs that enter the next compressor (left) from the output of the previous compressor (right). Currently, various compressors are used in multiplication operations [20]. The calculation width has an effect on the size and the way of using compressors. For this reason, different compressors have always been compared and attempts have been made to optimize them [14,15,17,18,22,23]. The logic shown in Figure 2(a) for the (3; 2) compressor is the same as the full adder. The (3; 2) compressor is used as the main part in the building blocks of the DADDA tree in reducing the depth of partial products [24]. However, CSAs cause problems such as glitches and uneven signal transition; and it will take longer to reduce the partial product depth. To prevent these problems, higher-order compressors are used in multiplication operations [15]. The (4; 2) compressor in Figure 2(b) is composed of the consecutive combination of two (3; 2) compressors. The general structure of (5; 2) compressors and (7; 2) (Figure 2(c) and 2(d)) can also be seen in the same way. In Figure 2(d), CGEN (Carry Generator) is used for the (7; 2) compressor circuit in addition to the previous circuits [25]. As the multiplier width increases, the demand for high-end compressors (compressors with more than 5 inputs) increases. On the other hand, researchers have used approximation in compressors as a way to reduce power consumption and increase speed. In this paper, a high-order approximate compressor (i.e. (8; 2)) is presented that can be used in multiplier circuits of 8 × 8 or larger.

FIG1. The general structure of the compressor [21]

2.LITERATURE

Basic compressors can be combined to handle a larger number of inputs simultaneously. (8; 2) Compressor accepts 8 values simultaneously and produces five carryout bits (i.e. cout0-cout4) for the next compressor in addition to the two final outputs (i.e. Carry and Sum). Equation (1) shows the logic function related to Sum output in the (8; 2) compressor in which A to H are the primary inputs and Cin0 to Cin4 are previous compressor Carries. Different combinations of basic compressors can be used to implement it. Recent works, using a combination of smaller compressors, have

https://ijgst.com.2024.v13.i2.pp957-965

used four different structures to form a hierarchical (8; 2) compressor [25]. Figure 3 shows different types of (8; 2) compressor designs.These compressors are hierarchical because a hierarchy of smaller adders builds them. The number of Carry inputs and outputs depends on the basic compressors used to design the (8; 2) compressor. For example, the (8; 2) compressor in Figure 3(d) consists of (7; 2) and (3; 2) compressors and only has three Carry inputs (Cin0 − 2) and three Carry outputs (Cout0-2). Because the internal structure of the (7; 2) compressor performs part of the

FIG2. Different types of compressors presented in Ref. [4]: a. (3; 2), b. (4;2), c. (7; 2)

calculation of the Carry bits. This makes the compressor depend on previous compressors. That is, the i-th compressor depends on the (i-1)-th compressor, which, in turn, depends on the previous compressors. Therefore, in this style of design, the delay of the circuit will depend on the delay of the previous circuit(s).

II.EXISTING SYSTEM AND PROPOSED SYSTEM

1.EXISTING SYSTEM

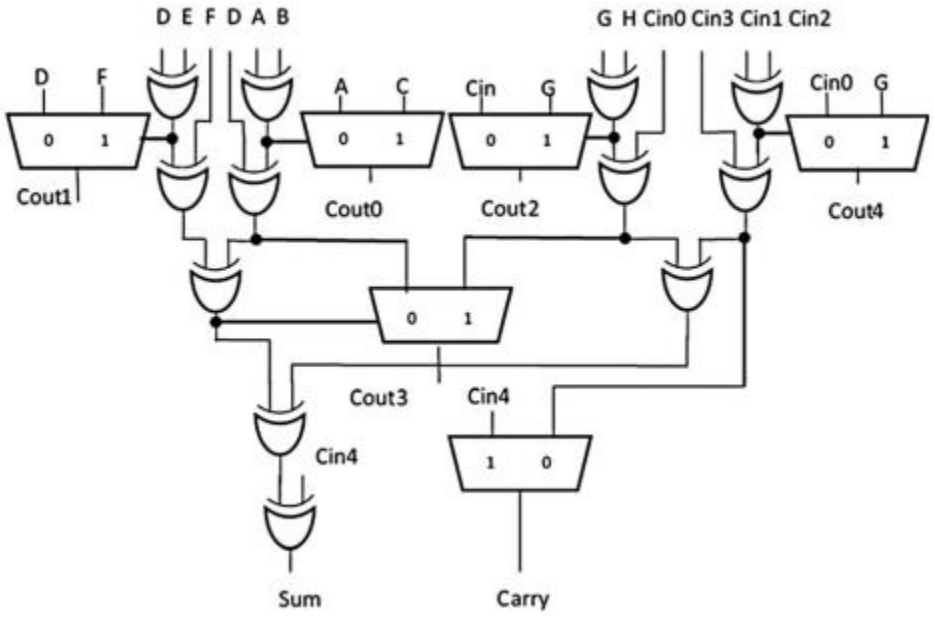

FIG3. The (8; 2) non-hierarchical compressor

hierarchical combination of smaller compressors, the compressor is implemented using only XOR and MUX gates and has a shorter delay than hierarchical versions, as shown in Figure 3. This method of compressor design is used to present the approximate compressor. A "Novel Approximate (8; 2) Compressor for Applications in Image Processing" likely refers to a specific design or algorithm for an approximate 8:2 compressor that finds applications in image processing. Let's break down the key components and discuss

https://ijgst.com.2024.v13.i2.pp957-965

potential features, advantages, and considerations:

1. (8; 2) Compressor:

Definition:In the context of data compression, (8; 2) indicates an 8-to-2 compressor. A compressor reduces the number of bits needed to represent a piece of data, and in this case, it's specifically designed for handling 8 bits and outputting 2 compressed bits.

Purpose:

Compressors are often used to reduce data size for efficient storage, transmission, or processing. The (8; 2) configuration suggests that eight input bits are compressed into two output bits.

2. Approximate Compressor:

Approximate Computing:

Approximate computing involves allowing for some degree of imprecision or error in computational processes. This can be leveraged to achieve computational efficiency, especially in applications where exact precision is not critical.

2.PROPOSED SYSTEM

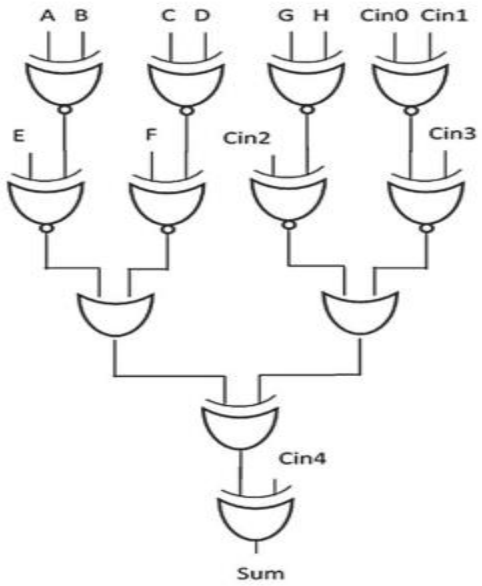

FIG4. The proposed design for the sum output of the approximate (8; 2) compressor.

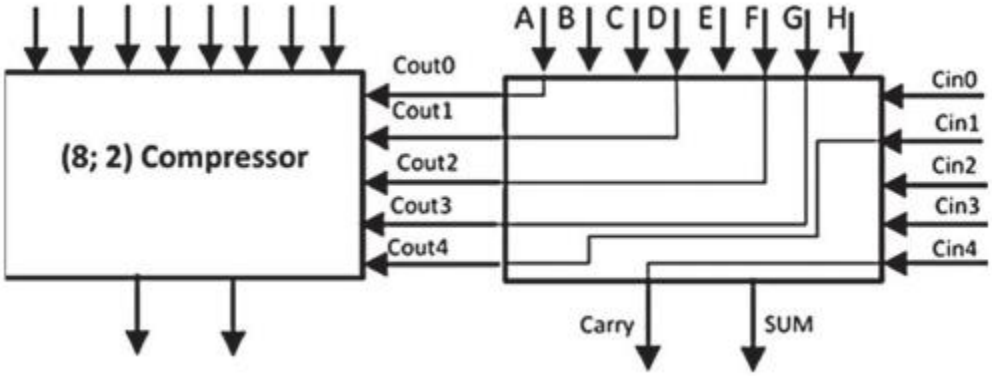

Fig5. Architecture of parallel approximated (8; 2) compressors in the sum of two 8-input vectors.

https://ijgst.com.2024.v13.i2.pp957-965

In the proposed design, the delay of the production path of the sum and the carry figures has been reduced. Because, compared with other logic gates, the XOR gate often has larger design overheads. By replacing the logic OR gate instead of XOR in the Sum circuit, an approximation was applied to the circuit so that the outputs can be obtained faster, while in 75% of the cases, the outputs of these two gates are similar. to calculate Carries, it is tried to use the value of inputs instead of calculating them. The proposed design for the sum output of the approximate (8; 2) compressor is shown in Figure 5 and Equation (2) shows its logic. Its logic (i.e. truth Table) in Python has been implemented to check the accuracy and precision of the approximate compressor. By applying all possible inputs to the proposed compressor, compared to the exact (8; 2), 5120 correct results out of 8192 possible outputs from Sum have been obtained, which shows that the output of Sum is approximately 63% of the time.

$$\begin{aligned} \text{Sum} = {} & (((A \oplus B)' \oplus E) + ((C \oplus D)' \oplus F))' \\ & \oplus (((G \oplus H)' \oplus Cin2)' + ((Cin0 \oplus Cin1)' \\ & \oplus Cin3)')' \oplus cin4 \\ Cout0 = A, Cout1 = D, Cout2 = F, Cout3 = H, & \\ Cout4 = Cin1, Carry = Cin4 & \quad (2) \end{aligned}$$

As mentioned above, the Carry and Cout outputs have been put equal to some compressor inputs to simplify and reduce the hardware of the circuit. Similarly to the sum function, in the software test of the circuit in Python, it was found that in 75% of the cases (that is, 6144 cases out of 8192 possible cases), the Cout outputs have an equal value. Therefore, a good approximation for Cout and Carry outputs can be considered,. For example, input A is connected to output Cout0. However, increasing the degree of approximation in the calculaion of the Carry outputs increases the error rate (i.e. Normalized Error Distance, NED [26]). Figure 6 shows the general architecture of parallel approximated (8; 2) compressors in the sum of two 8-input vectors.

IV. RESULTS AND ANALYSIS DISCUSSION

International Journal of Gender, Science and Technology
ISSN: 2040-0748
UGC Care Group I Journal.
Vol-13 Issue-02 Dec 2024
https://ijgst.com.2024.v13.i2.pp957-965

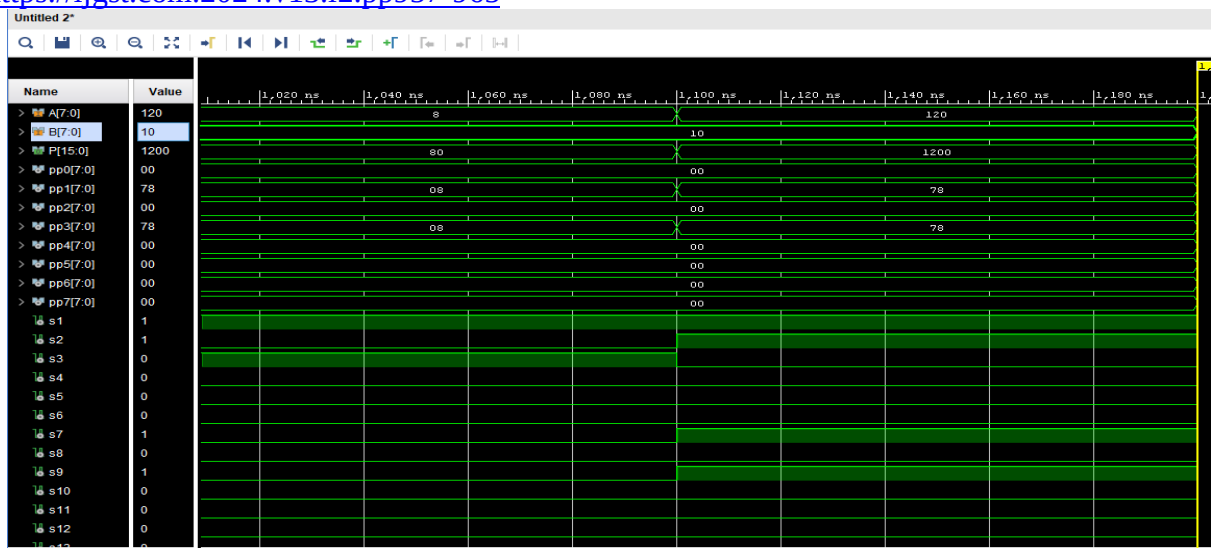

FIG1.SIMULATION OUTPUT

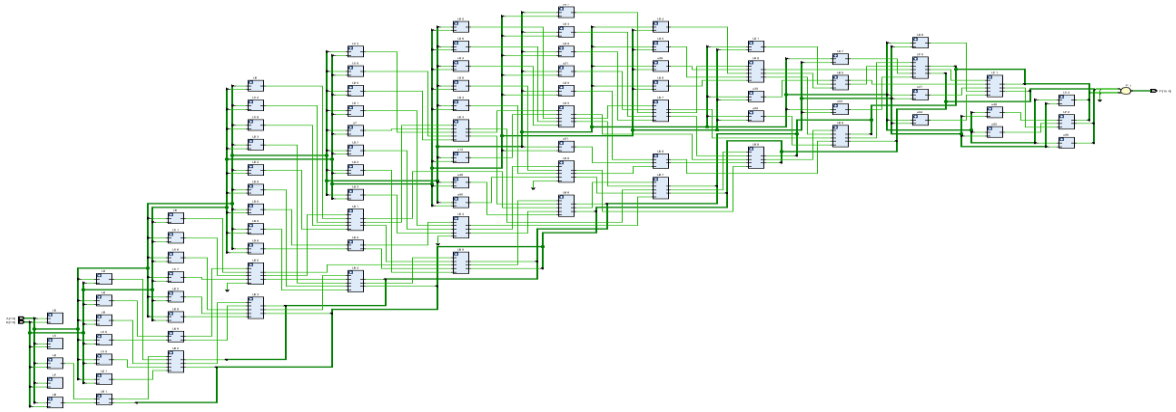

FIG2.RTL

International Journal of Gender, Science and Technology
ISSN: 2040-0748
UGC Care Group I Journal.
Vol-13 Issue-02 Dec 2024
https://ijgst.com.2024.v13.i2.pp957-965

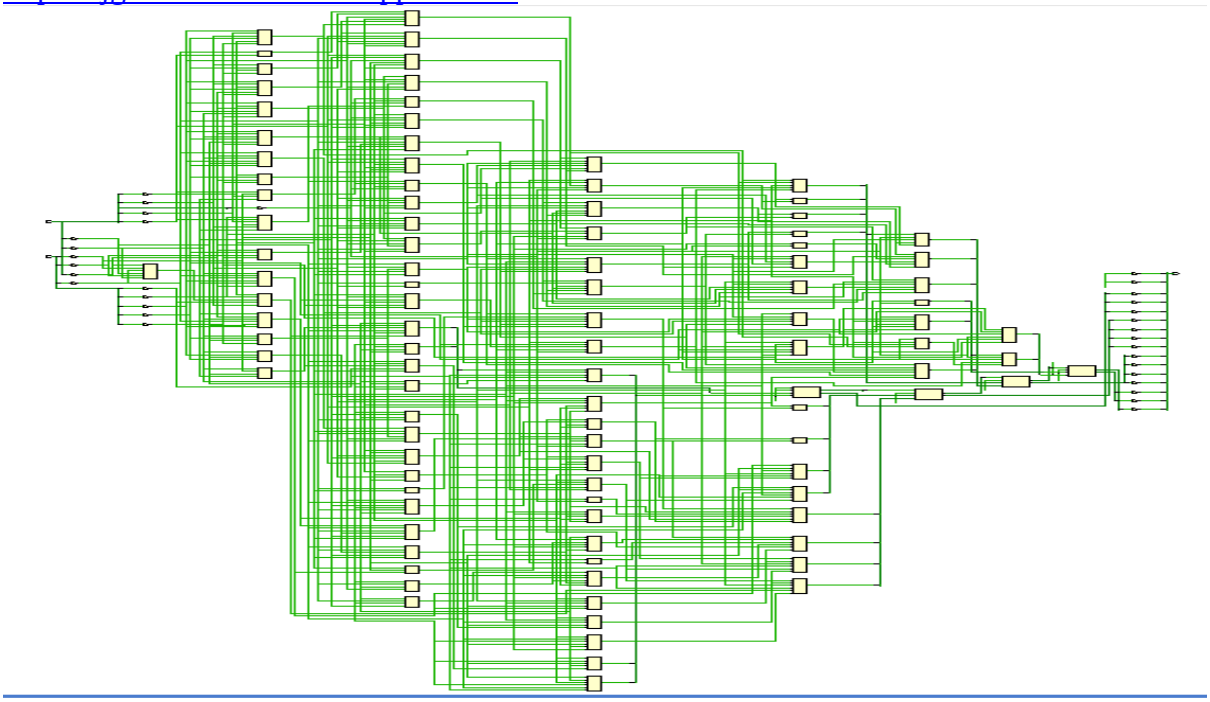

FIG3.SYNTHESIS DESIGN

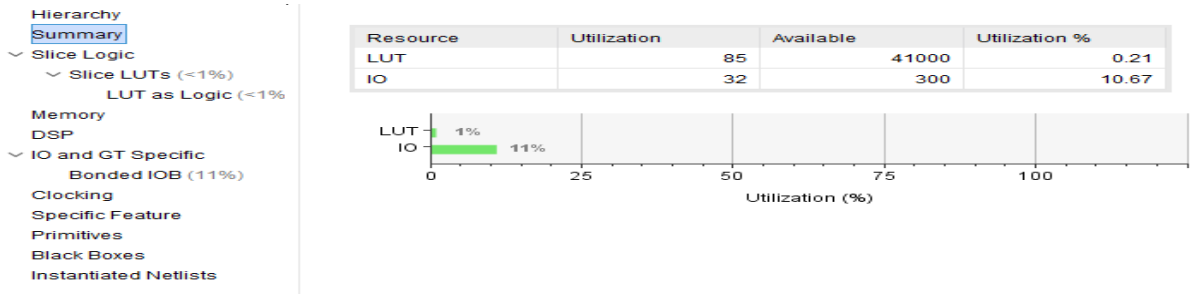

FIG4.UTILIZATION REPORT

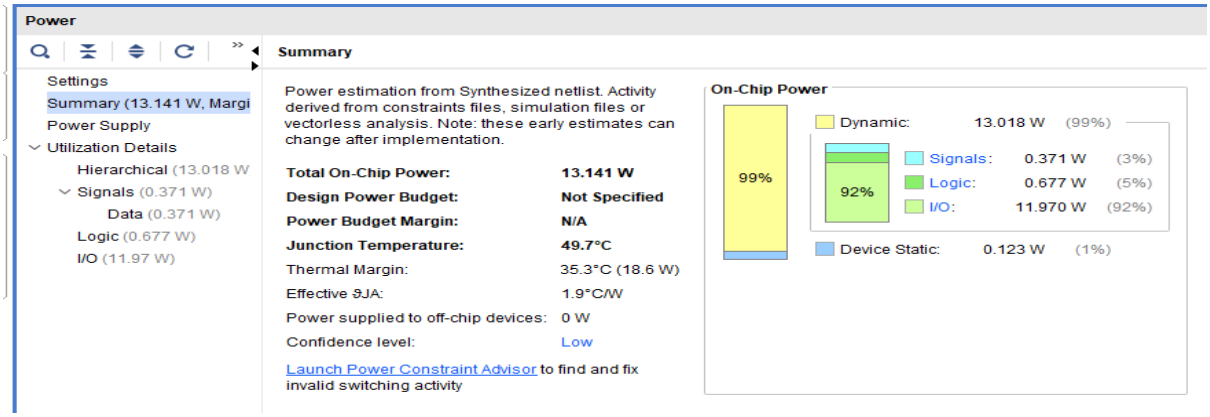

FIG5.POWER REPORT

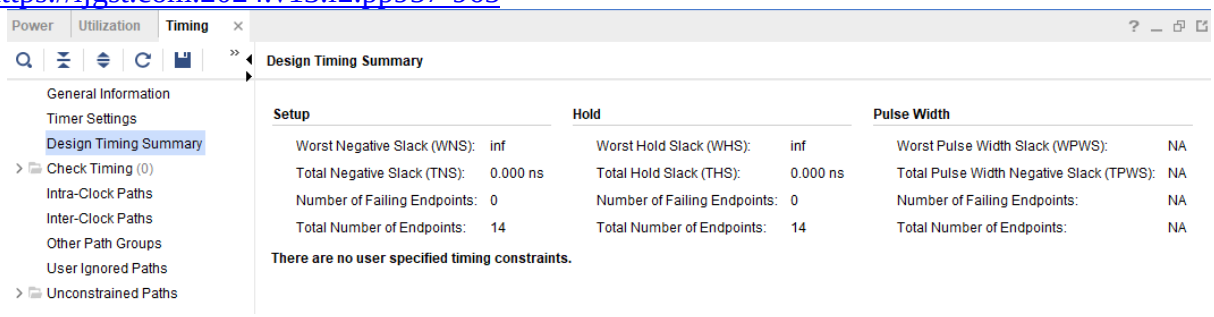

FIG6.TIMING REPORT

CONCLUSION

The higher speed of mathematical computing along with less power, delay and area is a challenge that has always occupied researchers in digital circuit design. Approximate computing is one of the methods that can provide a suitable answer to this challenge. Of course, considering the accuracy and correctness of the results in the desired application be acceptable. In this paper, the second stage of the multiplier circuit (i.e.reduction of partial products) was investigated. By designing the approximate (8; 2) compressor, the circuit design overhead has been significantly reduced. A Python program was written to check the number of correct states of the approximate compressor outputs. In 63% and 75% of cases, the results of the Sum and Cout outputs were correct, respectively. To evaluate the proposed circuit, we first used it in the implementation of 8 × 8 and 16 × 16 multipliers.

3.REFERENCES

1. J. Xun, V. Akhlaghi, Y. Jiang, and R. K. Gupta, “Energyefficient neural networks using approximate computation reuse,” in Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2018, pp. 1223–8. doi:10.23919/DATE.2018.8342202.

2. V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, “Analysis and characterization of inherent application resilience for approximate computing,” in Proceedings of the 50th Annual Design Automation Conference, Austin, 2013. pp. 1–9. doi:10.1145/2463209.2488873.

3. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. Image Process., Vol. 13, no. 4, pp. 600–12, April 2004.

4. B. Silveira, G. Paim, B. Abreu, et al., “Power-efficient sum of absolute differences hardware architecture using adder compressors for integer motion estimation design,” IEEE Trans. Circuits Syst. Regul. Pap., Vol. 64, no. 12, pp. 3126–37, August 2017. doi:10.1109/TCSI.2017. 2728802.

5. V. Gupta, D. Mohapatra, S. Phill Park, A. Raghunathan, and K. Roy, “IMPACT: IMPrecise adders for lowpower approximate computing,” in IEEE/ACM International Symposium on Low Power Electronics and Design, Fukuoka, 2011, pp. 409–14, doi:10.1109/ISLPED.2011. 5993675.

6. J. Han, and M. Orshansky, “Approximate computing: An emerging paradigm For energy-efficient design,” in the 18th IEEE European Test Symposium (ETS), Avignon, France, 2013, pp. 1–6, doi:10.1109/ETS.2013.6569370.

https://ijgst.com.2024.v13.i2.pp957-965

7. R. Amjada, R. Hafiz, M. U. Ilyas, M. S. Younis, and M. Shafique, “m-SAAC: multi-stage adaptive approximation control to select approximate computing modes for vision applications,” Microelectron. J., Vol. 91, pp. 84–91, August 2019. Doi: 10.1016/j.mejo.2019.07.010.
8. B. Liu, Z. Wang, S. Guo, et al., “An energy-efficient voice activity detector using deep neural networks and approximate computing,” Microelectron. J., Vol. 87, p