International Journal of Engineering Innovations in Advanced Technology
ISSN: 2582-1431 (Online), Volume-6 Issue-4, December 2024

A Three-Operand Binary Adder Using An Advanced VLSI Architecture Fast and Area-Efficient

Daram Praveen[1] , Dr S Kishore Reddy[2] , Dr K. Sanjeeva Rao[3] , Vasantha Naga Raju[4], G.srinivas[5]

[1]M.Tech Student, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.
[2]Associate Professor, HOD, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., India.
[3]Assistant Professor, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.
[4]Assistant Professor, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.
[5]Assistant Professor, ECE, VLSI System Design, Avanthi Institute of Engg. & Tech., Hyderabad, India.

Abstract - A three-operand binary adder is used to execute modular arithmetic in a number of cryptography and pseudorandom bit generator (PRBG) methods. Most of the time, the carry save adder (CS3A) is utilised to execute three-operand addition. The ripple-carrying stage (n) in the CS3A adds a transmission delay of O. The critical route time can be lowered down by employing a parallel prefix two-operand adder, such as the Han-Carlson (HCA) method for three-operand addition. The three-operand binary addition operation is done with a revolutionary adder architecture that is both quick and takes up little space. The adder delay is reduced to O(log2 n) using this solution, which is a relatively modest amount given the complexity of the problem. This is accomplished by the utilisation of pre-compute bitwise addition, which is then followed by carry prefix calculation logic. The design that was suggested was developed using a 32nm CMOS technology source that is not difficult to locate. Additionally, it has been validated to ensure that it is compatible with an FPGA device. Following the completion of post-synthesis, the recommended adder performed 3.12, 5.31, and 9.28 times better than the CS3A for designs with 32 bits, 64 bits, and 128 bits respectively. When it comes to processing huge amounts of data in a short amount of time, the HC3B is superior to the adder due to the fact that it requires less space, consumes less power, and has a shorter delay. When it comes to peak delay power (PDP) and average delay power (ADP), the adder that was recommended performs far better than the three-operand adder approaches that are currently in use. This is something that should be mentioned.

Keywords: PDP, HC3B, FPGA, Three-Operand, CS3A.

1. INTRODUCTION

In the majority of cases, multi-operand adders are utilized in the calculation of partial products using hardware multipliers. It is essential to have multi-operand adders with you whenever you are performing multiplication operations that require adding a large number of partial products. Because mathematical building components like multipliers require a significant amount of electricity, it is necessary to have electronic devices that are not only quiet but also efficient in terms of energy usage.

As a result of the fact that the adder's operands, which could be one or more bits, are visible, it is able to take in and output a number of bits [3]. Because carry is uncommon and partial sums are compressed, it is more accurate to portray a multi-operand adder using compressor trees. Additionally, carry is uncommon. Several different types of multi-threaded algebra (MTA) are employed in the course of this research. Adders such as the Wallace, Overturned-stairs, Balanced delay tree, and Array tree are examples of adding operations that fall within this category. Although they need a significant amount of power and take a considerable amount of time to complete, adders are important for machines that do mathematical operations. In addition to adders, multipliers are another type of resource-intensive mathematical system that utilises adders.

Using two linked binary tree adders (BTAs) is a typical method that is utilised when adding multiple operands on top of one another [5]. The latency and energy economy of the binary tree adder structure are both impacted positively or negatively by the efficiency of the adders. There are several different adder topologies that can be used to construct the BTA structure. Some of these topologies include ripple carry, carry-look ahead, parallel prefix, carry-select, and CSKA. The three primary ways in which different types of adders are differentiated from one another are in terms of room, latency, and power consumption levels.

When compared to ripple carry adders (RCA), quick adders, which are also commonly referred to as "other adders," have a shorter latency. Two of the disadvantages are having a greater surface area and requirements for a higher amount of energy. On the other hand, the RCA requires more time to connect, but it is more space and power efficient than the other option. It is important to keep in mind, however, that systems that contain several operands typically do not have sufficient resources. When selecting a binary tree adder for such systems, the most important considerations to take into account are the amount of space available and the way power is consumed. It is possible that any improvements made to the capabilities of binary addition could result in improved performance of the computer system, which would ultimately lead to improved system functioning in the long run due to the significance of the operation.

A. *Project Objectives and Justification*

Hold chains are extremely important to the process of binary addition. There exists a relationship that is inversely proportional between the depth of the carry chain and the width of the input operand. When the carry moves from the least significant bit (LSB) to the most significant bit (MSB), any negative outcome is realised. This occurs as soon as the carry moves from low to high. It is possible that the performance of carry-propagate adders could be improved by accelerating the carry chain without physically removing it totally. In order to improve computer architecture, digital designers commonly make use of speedier adders. These adders are used to determine which operations use the maximum amount of time. In a wide variety of digital circuit types, the binary adder is a component that is frequently found. A couple of examples include digital signal processors (DSPs) and microprocessor data routing units (MRUs). One of the most important goals is to improve the power delay performance of the adder. By demonstrating that three-operand adders perform better than two-operand adders in VLSI implementations, we made this point clear. Tanner EDA surpasses DSP and microprocessor-based solutions in terms of speed and power consumption, which is one of the factors that has contributed to the recent spike in popularity of reconfigurable logic. Because of the practical applications of the technology, this is especially noticeable in the mobile DSP industry and the telecommunications industry. When it comes to portable and mobile devices that make use of digital signal processing (DSP) more frequently, power efficiency is becoming an increasingly important factor.

Because of their customisable logic and routing capabilities, Field-Programmable Gate Arrays (FPGAs) offer a fresh approach to the development of three-operand adders. This is in contrast to their Very Large Scale Integration (VLSI) equivalents, which are comparable in terms of their capabilities. The basic Ripple Carry Adder (RCA) is useful because it makes use of a quick carry chain, which is a characteristic that is frequently found in modern Field-Programmable Gate Arrays (FPGAs).

2. LITERATURE REVIEW

Multiple operator summation, sometimes known as MOS, is a technique that is frequently used in block structures and fast functions. Researchers are investigating a variety of different alternatives in order to find a solution to the problem of having multi-operand addition operate well for medical Internet of Things devices. When it comes to classic adders, delays and problems with the area are common. In the context of filters and multipliers, they are employed for the purpose of gathering intermediate results. Compressor trees and multi-operand addition are two methods that can be utilised to expedite the process of performing decimal operations inside the given domain [1]. In order to do multiplication in an effective manner while utilising a minimal amount of resources, it is possible to make use of the Wallace tree (WT) multiplier in conjunction with the add and shift method [6].

A number of adder topologies have been presented by Carry Look-Ahead (CLA) and Parallel Prefix Adder (PPA) designs, both of which have the goal of reducing the amount of time required. These kinds of buildings, on the other hand, need for a significant amount of both area and energy. When space and power usage are taken into consideration, the Ripple Carry Adder (RCA) (15,16) emerges as the most advantageous option. The results that can be obtained with the carry selection scheme-based adder are equivalent to those that can be obtained with the RCA and the CLA [17]. Because of how quickly it performs its functions, the Carry Select Adder is an excellent choice for a wide variety of data processing tasks. For the purpose of lowering its impact on the environment and lowering its energy consumption, the CSLA has been extended (CSLA [18,19]).

By making a very slight modification to the gate level, the suggested configuration of the CSLA is able to significantly lower both its footprint and its power consumption. A low-bit-count variation of the traditional SQRT CSLA architecture has been taken into consideration. A logic architecture that makes use of binary to excess-1 converters is presented by the authors of reference [20] as a technique of reducing the amount of space and power that is consumed by CSLA. Christopher Wallace, an Australian computer scientist, is credited with developing a method that is known as the Wallace tree adder (WTA) [15]. This approach is an effective method for adding a large number of operands. In [15], the Wallace tree adder uses carry-save adders to divide the input into two words. This process is described in detail. Following that, it takes a separate two-operand adder and adds the two words together on its own. The full adders (FAs) are used in the construction of the carry-save adder, which also includes a single FA delay implementation. With its delay, the Binary Tree Adder (BTA) is frequently more efficient than the Wallace tree adder, despite the fact that it is less perfect than the Wallace tree adder.

The most important aspects of this thesis are the many implementations of three-operand adders as well as the computer systems that form the foundation of these adders. There is a possibility that the current problem will have an effect on a variety of computer designs, ranging from those that are specialist to those that are general-purpose. This initiative has the potential to be beneficial to a wide range of academic disciplines, including engineering and computer science, provided that sufficient effort is invested in it.

Researchers in this field are concentrating their efforts on tree-based adders on Tanner EDA[5] and the difficulties that its developers and users are encountering. In this work, the performance of the Han Carlson Adder (HCA) and the Carry Save Adder (CSA) is evaluated in comparison to that of Tanner EDA-based several-tree adder designs (HCA). It has been found that Tanner EDA[5] designs need to be improved in order to accomplish the goal of improving the performance of tree-based adders.

A Survey of the Literature

A number of individuals are of the opinion that computers routinely carry out actions that are illogical and do not have

any logical interpretations. This is related to the fact that computation generates heat and is irreversible. It is necessary for each irreversible action to generate a particular quantity of heat, which is roughly equivalent to kT, over the entirety of each machine cycle in order to demonstrate that the process is logically irreversible. By standardising indications and distinguishing them from one another in a manner that is both logical and contextual, dissipation serves to accomplish these goals. In this work, additional research into switching kinetics is carried out in order to investigate the relationship between switching speed and energy loss, as well as the potential errors that are produced by differences in temperature.

The second property is that it is logically conceivable to carry out computation without going in the other direction. This is illustrated by the fact that general-purpose computer automata, such as the Turing machine, are theoretically irreversible. The operation of these machines is capable of carrying out complicated computations in a manner that can be reversed at any time. Theoretically, thermodynamically reversible computers have the potential to do realistic computations in a sufficient amount of time while minimising their energy usage to less than one kilowatt-second (KT). In the realm of physics, this particular aspect is of the utmost significance. This is because conceptually reversible automata keep all of their intermediate results, which makes it possible for them to circumvent the process of permanently erasing information. This is a distinguishing feature in comparison to their relatives that cannot be reversed. Following the completion of the assignment, the next step is to print it out. The third stage is able to proceed in the opposite direction if the order of the processes in the first stage is reversed. The elimination of any undesirable intermediate results is achieved through this action. The possibility of this reversibility is made possible by the fact that the initial step can be turned back on. When it comes to the setup of the machine's output, there is only one instance of the input. Through the utilisation of a Turing machine that is outfitted with three cassettes, the findings will be displayed. There is an example of reversible computation that occurs during messenger RNA (mRNA) synthesis. For example, this case demonstrates the point.

3. EXISTING WORK

Up to this point, there have been no significant efforts made to resolve the RCA-BTA delay that has been referenced in the literature. The main route delay analysis is utilised in order to ascertain whether or not the RCA-BTA configuration can be rendered as delay-free as is humanly practical [5]. Enhancing the design metrics of fast adders is the primary focus of the majority of the work that is recommended in the publications. This work is intended to improve the WTA architecture. Even though the WTA has a shorter delay than the RCA-BTA (as shown in [5]), it is required to redesign it for different N values because of its non-smooth structure. This is the case despite the fact that the WTA has a lesser delay. On the other hand, the BTA structure is extremely user-friendly and straightforward, making it suitable for use with any value of N. A fast adder can be added at the end of the WTA in order to improve its performance, and the RCA delay can be decreased in order to lessen the lag that occurs between the RCA and the BTA frequencies.

A. ADER CLASSES

A significant amount of reliance is placed on the addition function by digital signal processors, microprocessors, and computers. This most fundamental operation can serve as a foundation upon which all other operations in mathematics can be constructed. Therefore, binary adder structures are an essential component for the effective execution of mathematical operations. If you look through any book on computer mathematics, you will discover that it contains lengthy explanations of a variety of circuit designs, each of which has its own distinct set of performance characteristics and applications. When it comes to binary adder designs, there has been a significant amount of research conducted, but not nearly as much on their performance.A digital circuit is the subject of this question. An additional source of information about electronic circuits that handle analogue data can be found by exploring electronic mixer. The addition machine, often known as the summer, is a tool that electronic engineers frequently utilise to do fundamental arithmetic operations. The regions of a processor that are responsible for addressing and table indexing are often where adders are located the most frequently. In addition, they are utilised in the math logic unit or units of a great number of computers and other sorts of processors. The vast majority of adders are only capable of handling binary integers; however, it is possible to adapt them so that they can also handle other forms, such as binary-coded decimal or excess-3.

Simply displaying negative numbers by utilising the two's or ones' complement is all that is required to convert an adder into an adder-subtractor. Additionally, a more complicated adder is required for any other type of signed number. In this section, you can find a selection of calculators.

B. ADDER HALF

To do the addition of A and B together, the half adder is utilised. Upon completion of these procedures, we are able to obtain S and C, where C represents the appropriate value to be utilised in the future addition. The final sum is denoted by 2C plus S. The proper construction of a half-adder makes use of both an AND gate and an XOR gate in order to process the integer S. The carry outputs of two half adders can be connected to one another using an OR gate, which allows for the construction of a complete adder.

Figure 1:Half Adder

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

*Figure 2: **Half Adder***

C. COMPLETE ADDER

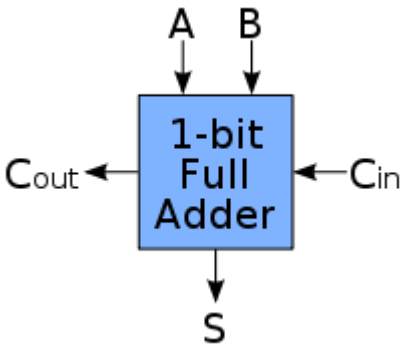


*Figure 3: **full adder***

In order to demonstrate the role that Cin and Cout play in a multi-bit adder, the sides of the block indicate their attributes. By transferring values between the input and output of the full adder, it is feasible to transfer values.

The result of adding bits A and B is the binary value known as Cin.

[2] in the A full-adder is able to multiply a binary number by any number of bits, ranging from 8 to 32, when it is given the binary integer. At the output of the circuit, the two-bit sum is often represented as either Cout or S. This is the standard method. The following is the truth table for an adder that is complete and only uses one bit:

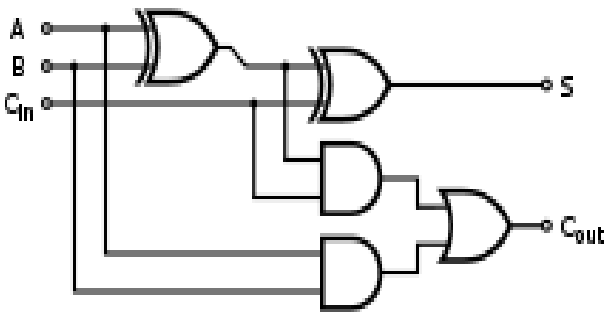


*Figure 4: **Implimentation Of full adder***

The construction of a complete adder can be accomplished in a number of different ways, ranging from a circuit at the transistor level to a set of additional gates. Through the utilisation of and, this option is illustrated. In this particular version, the logic will not be impacted in any way by substituting an XOR gate for the final OR gate that comes before the carry-out output. When working with integrated circuit chips that only have one form of gate, it is simple to use only two different kinds of gates. The prospective applications of cout are now open to include the following:

It is possible to create a complete adder by simply connecting A and B to the input of a half adder.

Following the connection of the Ci and the outputs of the other carriers, a sum is added to an input input. You can construct S and Cout as majority functions or as three-bit XORs of A, B, and Ci by using these three bits as a basis. Both of these options are possible.

4. PROPOSED METHOD

A. INTRODUCTION

Integrated circuits are becoming increasingly dependent on their field reliability as a result of increasing integration density and smaller processes. A, B, and C are the numbers. When it comes to mission-critical systems, such as embedded devices and server-class computers, error monitoring that functions in parallel is absolutely necessary. Due to the fact that addition is the first step in any computation, huge integrated circuits include a great deal of them. For the purpose of developing effective systems, having reliable adders is absolutely necessary. There has been discussion over the development of adders that are capable of simultaneously detecting faults in the numbers 4, 5, 6, and 7. [4] and [7] are two references that provide examples of error detection through the use of parallel prefix adders. In the event that these adders produce inaccurate results, we are able to identify the root problem by utilising parity prediction. A more accurate description of their contribution would be to say that it is an anticipated parity of the outcome. It is possible to compare this parity to the real parity of the result, which is also referred to as the XORed value of the sum bits. This comparison can be used to determine whether or not the output is erroneous. Carry-choice adders that are able to detect and fix errors were proven in two papers: [5] and [6]. Two internal numbers that are contained in the carry select adders can be compared to one another in order to reveal any inaccurate output. A proposal has been made in the past to make use of concurrent error detecting adders, which have the capability of identifying wrong results from a single defect. Nevertheless, determining the origin of a defective product that has several origins is not always an easy task. If the first problem occurs before the second problem is identified as a result of inaccurate output, then it may be difficult, if not impossible, to find two defects that occur simultaneously. In order to ensure that the machine addition process is error-free, it is necessary to identify the first difficulty before moving on to the second. Electromigration [9, 10] and stress migration

[11, 12] are two examples of the more challenging errors that can occur as a result of wear and tear [2, 8]. A concurrent error that indicates an adder failure must be readily apparent regardless of whether the device is operating, whether maintenance is being performed, or whether a reboot has been performed. People tend to believe that VLSI chips are error-free all the time, until they break. By the time it takes place, a failure has taken place. The passage of time is expected to bring to an increase in the visibility of faults. As a result of this, we focus our attention on a model that has a flaw known as stuck-at. The identical failure model is investigated by all three of these sources [4, 5]. In this short, we will demonstrate a proven concurrent adder that is able to identify errors. None of the proposed adders are able to identify concurrent faults [13–19], despite the fact that there are the most adders ever proposed. According to the findings of these research, the most important characteristic is C-testability, which refers to the capability of testing a circuit using a test set of a given size independent of the bit width of the circuit. In light of this, the C-testability is the most important feature to consider. C is the language that you can use to test the adder that has been proposed. This function is a godsend when it comes to testing the system after it has been put into production. It was the multi-block carry option adder that served as the inspiration for the suggested adder. The multi-block carry select adder will choose one of two sum outputs for each and every possible value of the carry input. Other options are also available. For additional information, please refer to [20]. This decision is going to be determined by the carry input value. The design of the adder that has been proposed makes it possible to use parity prediction in order to discover errors concurrently because of its architecture. In the event that only one stuck-at error were to occur, the expected parity output and the repeated carry outputs of the adder might be different from the parity of the sum result. With the help of this information, we will be able to determine whether or not the issue is causing any outputs that are erroneous. There are just 10 different designs that can be utilised to test the adder in a single scenario where it is stuck at fault. In this demonstration, both the simplicity of testing anything and the capacity to find problems simultaneously are illustrated. By creating a 32-bit adder, we were able to demonstrate a 70 percent increase in the amount of hardware overhead. By employing random pattern generators, we were able to generate errors in order to evaluate the capability of the system to detect simultaneous errors. The results of the fault simulation show that each of the ten input patterns has been totally successful each and every time. This is the case regardless of the circumstances presented.

B. ***The size of a multiple-block selection agent***

Figure 1 provides a simple illustration of an n-bit multi-block carry select adder. This adder is used to select several blocks. It takes in the addend Y, which is written as [yn1.. y0], and the agehend X, which is written as [xn1.. x0]. Specifically, it is responsible for the output of both the carry output (cn) and the total (S): [sn1.. s0]. It is made up of a great number of little things. Block 0 is the name of the first installment included in this series. As an illustration, let us assume that the bit width of the kth block (henceforth referred to as "blockk") is about nk. It is possible for each and every NK to be one of a kind. The carry information is contained in the block's input cink as well as the exit coutk. Block 0 is comprised of a row of full adders for the purpose of performing n-bit addition through the utilisation of a ripple carry mechanism. The construction of these complete adders involves the utilisation of an incrementer (INC) and a half adder (HA). A two-bit binary number, in addition to a carry bit, is what an INC takes as its input. In the subsequent step, it sends the sum as a binary number consisting of two bits. The remaining blocks are made up of a row of HAs, a row of a 2:1 multiplexer (MUX), two rows of interconnect circuits (INCs) (INC0 and INC1), and yet another row of INCs. Figure 1(b) displays the plans for the HA, INC, and MUX gates while they are at the gate level. Both INC0 and INC1 carry out two distinct sums in each and every block, with the exception of block 0. For one of these responses, it is assumed that the carry input cink is zero, whereas for the other, it is assumed to be one. It is the MUX row that chooses the appropriate one. Because of this truncated form, the position of a signal within the block as well as the block number are given by the signal's name, which contains two subscripts. It reads (0 j nk) for the values of xk,j and yk,j, which are associated with the bits in the input array that are located at position j in blockk. The equation L = j + Pk1 m = 0 nm applies here, and xk,j = xl. Using the number that comes after their names, signals are assigned a label that corresponds to the INC column in which they are discovered. There is also the possibility of using the notation s 0 k,j (0 j nk). The output of running INC0 at blockk's point k is denoted by the letter j in this instance. Both the carry indications for bit j 1 are represented by the expressions c 0 k,j and c 1 k,j, depending on the context. What these mean is that the INC0 k and INC1 k indicators are being used. By sending out the range [sk,nk1.. sk,0], we are able to feed the ranges [xk,nk1.. xk,0] and [yk,nk1.. yk,0] into blockk.

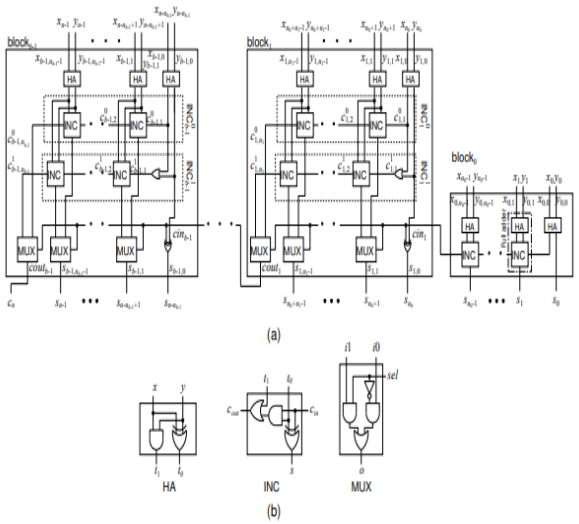


Figure 5: Multi-block carry select adder (a), and the gate-level designs of HA, INC and MUX (b).

C. With concurrently error-detectable adversity and easy testability

The solution that we have proposed is an adder that is simple to test and has the ability to detect problems concurrently. In Part 4.1, we will discuss the design of the adder as well as the linkages that exist between its blocks. With the help of Sections 4.2 and 4.3, we demonstrate that the adder is simple to test and has the ability to identify concurrent problems. Within Section 4.4, the hardware cost of the adder is broken down and explained.

Configuration

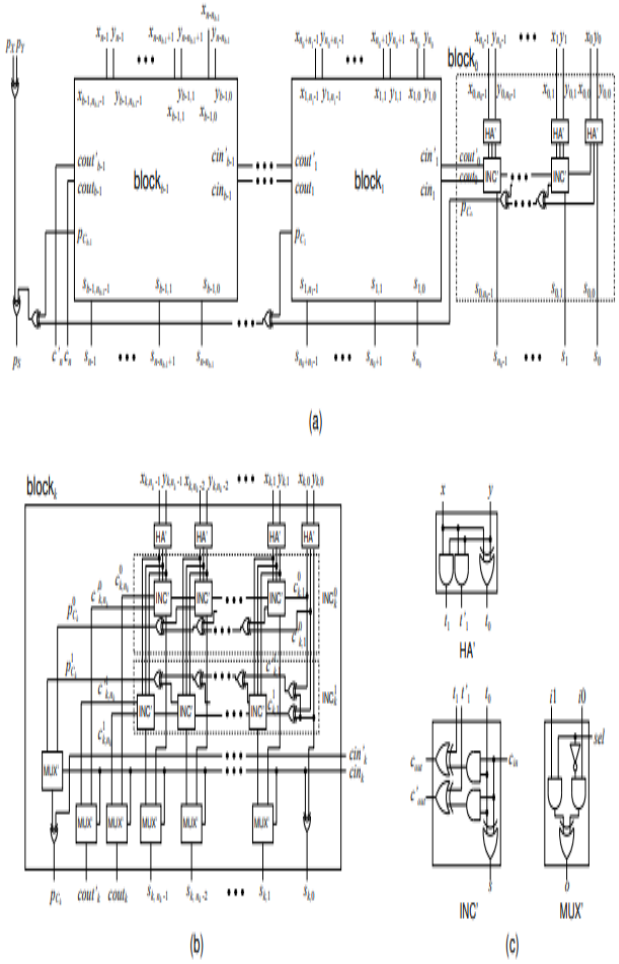


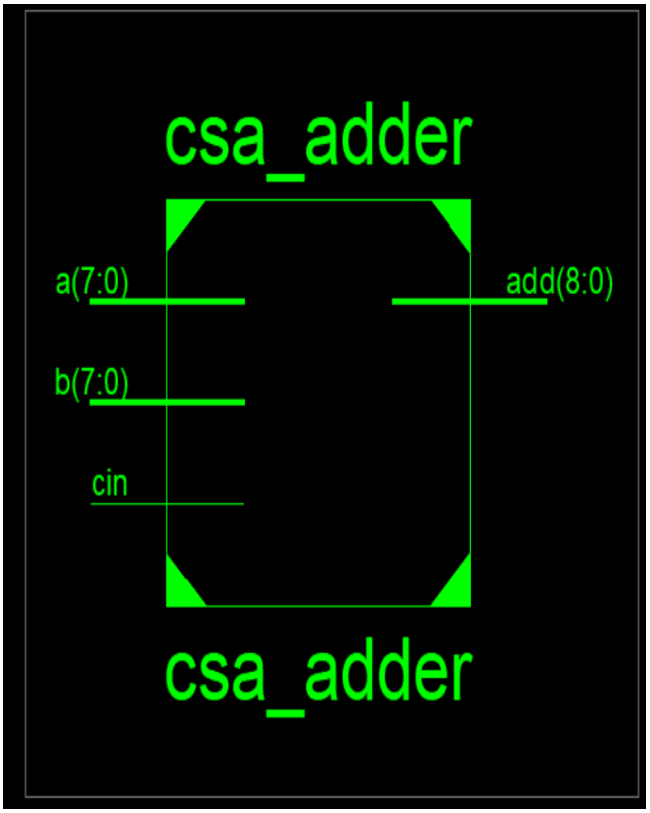
Figure 6: depicts a concurrent error detectable adder (a), the adder's blockk (k = 1) design (b), and the HA', INC', and MUX' gate-level designs (c).

Figure 2a illustrates a possible application of the adder device in a hypothetical scenario. The result that is produced by XORing Y with X is accompanied by the operands X and Y, as well as their respective parities, which are denoted by the marks pX and pY. The main, the anticipated parity (pS) of the sum, and two carry computations (cn and c′ n) are the three outputs that are derived from S. The length of a block, denoted by nk, must always be between zero and two. One stuck-at mistake is the symptom that this adder problem presents itself as. In order to determine it, one method involves comparing the expected parity pS with the actual parity of the sum result S, in addition to the values of cn and c′ n. This is one approach to determining it. There is also the possibility that input data sets, such as X and pX or Y and pY, include a single discrepancy. Carry inputs are present in each and every one of them, with the sole exception of the first addition block. The parity pCk of carry signals, which is equal to ck,nk1, ck,1 cink, is a representation of the carry bits that are generated when Xk, Yk, and cink are added together. Within each addition block, there are two carry outputs that are available. In Figure 2(b), the adder's addition block that was suggested is represented by the block with the size blockk (k - 1). Sk is the total that is produced by the addition block after it has been given Xk and Yk as inputs. Along with the inputs and outputs that are considered to be conventional, there are also carry inputs and outputs that are symbolised by the symbols cink and cin′ k. The indication that it sends out is referred to as pCk, which is an abbreviation for "parity of carries." There is a connection between coutk and cout'k, as well as between cink+1 and cin'k+1. FIGURE 2(c) illustrates the gate-level architectures for the MUX', INC', and HA' respectively.

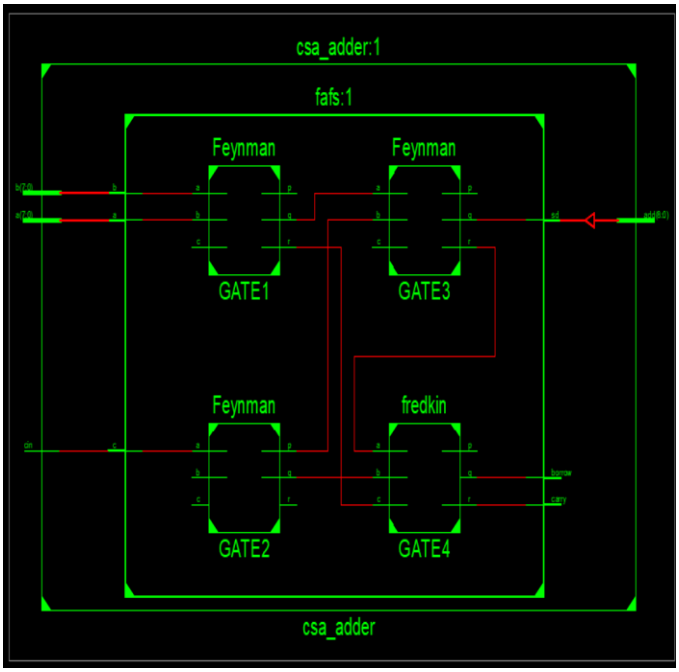
HA' and INC' are able to identify flaws concurrently thanks to the presence of two carry outputs. One of the carry bits can be used to add, while the other carry bit can be used to predict the parity. It is strictly forbidden for any INC or rising HA to ever have a single stuck-at fault that manifests itself concurrently in both the carry signals and the sum output. The primary goal of both HA' and INC' is to prevent this from happening. What exactly is going on? Due to the fact that an INC or an ascending HA being stuck at fault is the cause of an inaccurate total result and an improperly predicted parity, the rationale for this is straightforward. XOR gates are utilised in order to make the testing of the MUX and INC methods more straightforward. Due to the fact that they eliminate the potential of concealing the repercussions, XOR gates make it easy to observe the result of a problem. Except for the most and least significant bits of INC0 k and INC1 k, respectively, every addition block has an XOR gate appended to it. The only exceptions to this rule are the latter two bits. Different calculations are made by the two rows of INCs in order to arrive at the parity output pCk. p 0 Ck is the symbol that is used to represent it in INC0 k, and the formula that can be used to compute it is c′0 k,nk1 c′0 k,1. An application of the formula c ′1 k,nk1 c ′1 k,1 is required in order to acquire the second number, which is indicated as p 1 Ck. After selecting an XOR gate as its output, the leftmost MUX connects an XOR gate to one of the carry inputs, which is denoted by the letter cink. This results in the creation of the parity of the carry bits, which is notated as cin′ k. XOR gates are utilised by the adder in order to obtain pCk and, ultimately, the projected parity pS of the sum, which is computed as (pX pY) (pCb1 pC0). Due to the fact that, in a perfect scenario, si is equal to xi, yi, and ci, this declaration is accurate.

4. SIMULATION RESULT

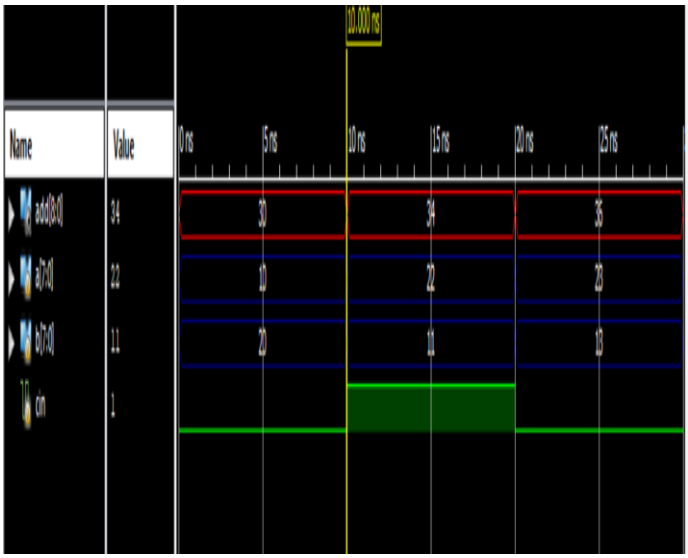
RTL:



Internship Blood Glucose Test:



Simulation Outcomes:



CONCLUSION

A space-saving adder algorithm and the related VLSI design for performing modular mathematics are proposed in this study. This technique has the potential to be beneficial in PRBG and cryptography environments. Through the utilisation of a four-stage prefix adder, this method effectively adds three input operands concurrently. In PG logic and bit-addition logic, the elimination of unnecessary prefix calculation steps often results in a smaller proposed design. This is because the designs are more compact. The hybrid Han-Carlson two-operand adder (HHC2A) served as the foundation for the development of the hybrid Han-Carlson three-operand adder (HHC3A), which was a further development of the HHC2A. This three-operand Kogge Stone hybrid Han-Carlson adder, which was written in Verilog HDL, is quite similar to the adder architecture that was suggested. Included in the Spartan 3 technology bundle are the central processor units and delay timing components of each and every one of these devices. Our adder was incorporated into the adder component of the FIR filter in order to enhance the performance of both the area and the latency performance. Regardless of the delay, we were certain that our adder was superior to the competition.

REFERENCES

M. M. Islam, M. S. Hossain, M. K. Hasan, M. Shahjalal, and Y. M. Jang, "FPGA implementation of high-speed area-efficient processor for elliptic curve point multiplication over prime field," *IEEE Access*, vol. 7, pp. 178811–178826, 2019.

[2] Z. Liu, J. GroBschadl, Z. Hu, K. Jarvinen, H.Wang, and I. Verbauwhede, "Elliptic curve cryptography with efficiently computable endomorphisms and its hardware

implementations for the Internet of Things," *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 773–785, May 2017.

[3] Z. Liu, D. Liu, and X. Zou, "An efficient and flexible hardware implementation of the dual-field elliptic curve cryptographic processor,"*IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2353–2362, Mar. 2017.

[4] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Design*.New York, NY, USA: Oxford Univ. Press, 2000.

[5] P. L. Montgomery, "Modular multiplication without trial division," *Math.Comput.*, vol. 44, no. 170, pp. 519–521, Apr. 1985.

[6] S.-R. Kuang, K.-Y. Wu, and R.-Y. Lu, "Low-cost high-performance VLSI architecture or montgomery modular multiplication," *IEEE Trans.Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 2, pp. 434–443,Feb. 2016.

[7] S.-R. Kuang, J.-P. Wang, K.-C. Chang, and H.-W. Hsu, "Energy-efficient high-hroughput montgomery modular multipliers for RSA cryptosystems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 11,pp. 1999–2009, Nov. 2013.

[8] S. S. Erdem, T. Yanik, and A. Celebi, "A general digit-serial architecture for montgomery modular multiplication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 5, pp. 1658–1668, May 2017.

[9] R. S. Katti and S. K. Srinivasan, "Efficient hardware implementation of a new pseudo-random bit sequence generator," in *Proc. IEEE Int. Symp.Circuits Syst.*, Taipei, Taiwan, May 2009, pp. 1393–1396.

[10] A. K. Panda and K. C. Ray, "Modified dual-CLCG method and its VLSI architecture for pseudorandom bit generation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 3, pp. 989–1002, Mar. 2019.

[11] A. Kumar Panda and K. Chandra Ray, "A coupled variable input LCG method and its VLSI architecture for pseudorandom bit generation," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 4, pp. 1011–1019, Apr. 2020.

[12] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design—A Systems Perspective*. Reading, MA, USA: Addison-Wesley, 1985.

[13] T. Kim, W. Jao, and S. Tjiang, "Circuit optimization using carry-saveadder cells," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*,vol. 17, no. 10, pp. 974–984, Oct. 1998.

[14] A. Rezai and P. Keshavarzi, "High-throughput modular multiplication and exponentiation algorithms using multibit-scan–multibit-shift technique,"*IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 9,pp. 1710–1719, Sep. 2015.

[15] A. K. Panda and K. C. Ray, "Design and FPGA prototype of 1024-bit Blum-Blum-Shub PRBG architecture," in *Proc. IEEE Int. Conf. Inf.Commun. Signal Process. (ICICSP)*, Singapore, Sep. 2018, pp. 38–43.