DESIGN AND IMPLEMENTATION OF A LOW-POWER HIGH-SPEED CONFIGURABLE ADDER FOR APPROXIMATE COMPUTING

Fahmeena Mahween[1], Tazeen[2]
[1]PG Scholar, Department of ECE, Shadan Women's College of Engineering and Technology, Hyderabad, fahmeenamahween0822@gmail.com
[2]Asst Professor, Department of ECE, Shadan Women's College of Engineering and Technology Tazeen@gmail.com

To Cite this Article
Fahmeena Mahween, Tazeen, "Design And Implementation Of A Low-Power High-Speed Configurable Adder For Approximate Computing", Journal of Science Engineering Technology and Management Science, Vol. 03, Issue 08, August 2026, pp: 736-744, DOI: http://doi.org/10.64771/jsetms.2026.v03.i08.pp736-744
Submitted: 01-07-2026 *Accepted: 07-08-2026* *Published: 14-08-2026*

ABSTRACT
Approximate computing is an efficient approach for error-tolerant applications because it can trade off accuracy. Addition is a key fundamental function for these applications. In this project, proposing accuracy-configurable adder that also maintains a small design area. The proposed adder is based on the approximation of least significant bits, and its configurability of accuracy is realized by with reversible logic gates. Compared with the conventional carry look-ahead adder using maskable half adder the proposed experimental result demonstrates the achievement of the original purpose of optimizing area without reducing the accuracy.

1. INTRODUCTION

Applications that have recently emerged (such as image recognition and synthesis, digital signal processing, which is computationally demanding, and wearable devices, which require battery power) have created challenges relative to power consumption. Addition is a fundamental arithmetic function for these applications. Most of these applications have an inherent tolerance for insignificant inaccuracies. By exploiting the inherent tolerance feature, approximate computing can be adopted for a tradeoff between accuracy and power. At present, this tradeoff plays a significant role in such application domains. As computation quality requirements of an application may vary significantly at runtime, it is preferable to design quality configurable systems that are able to tradeoff computation quality and computational effort according to application requirements. The previous proposals for configurability suffer the cost of the increase in power or in delay. In order to benefit such application, a low-power and high-speed adder for configurable approximation is strongly required. In this project, we propose a configurable approximate adder, which consumes lesser power than does with a comparable delay and area. In addition, the delay observed with the proposed adder is much smaller than that of with a comparable power consumption. Our primary contribution is that, to achieve accuracy configurability the proposed adder achieved the optimization of power and delay simultaneously and with no bias toward either. We implemented the proposed adder, the conventional carry look-ahead adder (CLA), and the ripple carry adder (RCA) in Verilog HDL.

ADDER

An adder is a digital circuit that performs addition of numbers. In many computers and other kinds of processors adders are used in the arithmetic logic units or ALU. They are also used in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators, and similar operations. Although adders can be constructed for many number representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers, it is trivial to modify an adder into an adder–subtractor. Other signed number representations require more logic around the basic adder. Another common and very useful combinational logic circuit which can be constructed using just a few basic logic gates allowing it to add together two or more binary numbers is the Binary Adder.

A basic Binary Adder circuit can be made from standard 'AND' and 'Ex-OR' gates allowing us to "add" together two single bit binary numbers, A and B. The addition of these two digits produces an output called the SUM of the addition and a second output called the CARRY or Carry-out, (COUT) bit according to the rules for binary addition. One of the main uses for the Binary Adder is in arithmetic and counting circuits.

BINARY ADDITION

Binary Addition follows these same basic rules as for the denary addition above except in binary there are only two digits with the largest digit being "1". So, when adding binary numbers, a carry out is generated when the "SUM" equals or is greater than two (1+1) and this becomes a "CARRY" bit for any subsequent

addition being passed over to the next column for addition and so on.

HALF ADDER CIRCUIT

A half adder is a logical circuit that performs an addition operation on two binary digits. The half adder produces a sum and a carry value which are both binary digits.

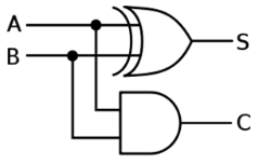


Fig: logic symbol of half adder

A	B	carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Fig: truth table

a, b are inputs and carry, sum are outputs.

Sum = A xor B

Carry = A and B

From the truth table of the half adder, we can see that the SUM (S) output is the result of the Exclusive-OR gate and the Carry-out (Cout) is the result of the AND gate. Then the Boolean expression for a half adder is as follows One major disadvantage of the Half Adder circuit when used as a binary adder, is that there is no provision for a "Carry-in" from the previous circuit when adding together multiple data bits.

FULL ADDER CIRCUIT

The main difference between the Full Adder and the previous Half Adder is that a full adder has three inputs. The same two single bit data inputs A and B as before plus an additional Carry-in (C-in) input to receive the carry from a previous stage as shown below.

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

Fig: truth table

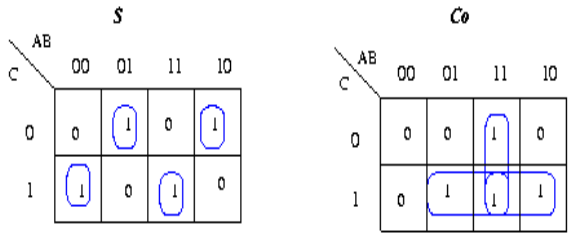


Fig: k-maps for sum and carry

Sum value is calculated using Sum= A xor B xor Cin

carry value is calculated using Carry = (A and B) or (B and Cin) or (A and Cin).

Then the full adder is a logical circuit that performs an addition operation on three binary digits and just like the half adder, it also generates a carry out to the next addition column. Then a Carry-in is a possible carry from a less significant digit, while a Carry-out represents a carry to a more significant digit.

In many ways, the full adder can be thought of as two half adders connected together, with the first half adder passing its carry to the second half adder as shown

FULL ADDERS USING HALF ADDERS

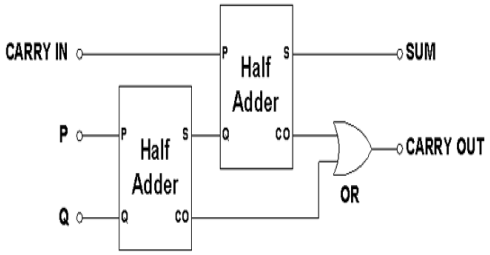


Fig: full adder using two half adders

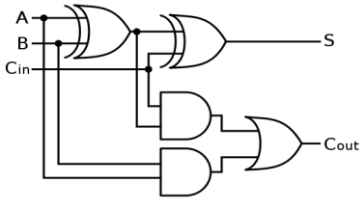


Fig: logic circuit

As the full adder circuit above is basically two half adders connected together, the truth table for the full adder includes an additional column to take into account Carry-in, CIN input as well as the summed output, S and the Carry-out, COUT bit.

N-BIT BINARY ADDER

We have seen above that single 1-bit binary adders can be constructed from basic logic gates. But what if we wanted to add together two n-bit numbers, then n number of 1-bit full adders need to be connected or "cascaded" together to produce what is known as a Ripple Carry Adder.

A "ripple carry adder" is simply "n", 1-bit full adders cascaded together with each full adder representing a single weighted column in a long binary addition. It is

called a ripple carry adder because the carry signals produce a "ripple" effect through the binary adder from right to left, (LSB to MSB).

A 4-BIT RIPPLE CARRY ADDER

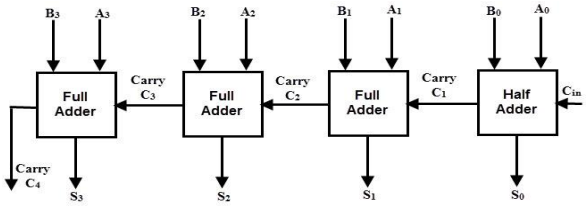
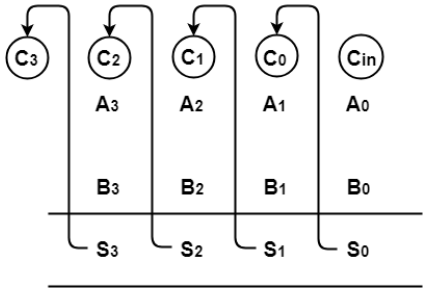


Fig: 4-bit binary parallel adder

• 4-bit ripple carry adder is used for the purpose of adding two 4-bit binary numbers.

• In mathematics, any two 4-bit binary numbers A3A2A1A0 and B3B2B1B0 will be added as-

As shown, Ripple Carry Adder works in different stages where the carry out produced by each full adder as output serves as the carry in input for its adjacent most significant full adder. When the carry in becomes available to the full adder, it activates that full adder and it comes into operation.



Why Ripple Carry Adder is called so?

In Ripple Carry Adder, the carry out produced by each full adder as output serves as the carry in input for its next most significant full adder.

• Since in ripple carry adder, each carry bit ripples or waves into the next stage, that's why it is called by the name "Ripple Carry Adder".

LIMITATION OF RIPPLE CARRY ORDER•

Ripple Carry Adder does not allow all full adders to be used simultaneously and each full adder has to necessarily wait till the carry bit becomes available from its adjacent less significant full adder.

• This increases the propagation time and due to this reason, ripple carry adder becomes extremely slow which is considered to be the biggest disadvantage of using ripple carry adder.

2. LITERATURE REVIEW

A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and Analysis of Approximate Compressors for Multiplication", Inexact (or approximate) computing is an attractive paradigm for digital processing at nanometric scales. Inexact computing is particularly interesting for computer arithmetic designs. This paper deals with the analysis and design of two new approximate 4-2 compressors for utilization in a multiplier. These designs rely on different features of compression, such that imprecision in computation (as measured by the error rate and the so-called normalized error distance) can meet with respect to circuit-based figures of merit of a design (number of transistors, delay and power consumption). Four different schemes for utilizing the proposed approximate compressors are proposed and analyzed for a Dadda multiplier. Extensive simulation results are provided and an application of the approximate multipliers to image processing is presented. The results show that the proposed designs accomplish significant reductions in power dissipation, delay and transistor count compared to an exact design; moreover, two of the proposed multiplier designs provide excellent capabilities for image multiplication with respect to average normalized error distance and peak signal-to noise ratio (more than 50 dB for the considered image examples).

C. Liu, J. Han, and F. Lombardi, "A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery", Proc. of IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE), [Approximate circuits have been considered for error-tolerant applications that can tolerate some loss of accuracy with improved performance and energy efficiency. Multipliers are key arithmetic circuits in many such applications such as digital signal processing (DSP). In this paper, a novel approximate multiplier with a lower power consumption and a shorter critical path than traditional multipliers are proposed for high-performance DSP applications. This multiplier leverages a newly-designed approximate adder that limits its carry propagation to the nearest neighbors for fast partial product accumulation. Different levels of accuracy can be achieved through a configurable error recovery by using different numbers of most significant bits (MSBs) for error reduction. The approximate multiplier has a low mean error distance, i.e., most of the errors are not significant in magnitude. Compared to the Wallace multiplier, a 16-bit approximate multiplier implemented in a 28nm CMOS process shows a reduction in delay and power of 20% and up to 69%, respectively. It is shown that by utilizing an appropriate error recovery, the proposed approximate multiplier achieves similar processing accuracy as traditional exact multipliers but with significant improvements in power and performance.

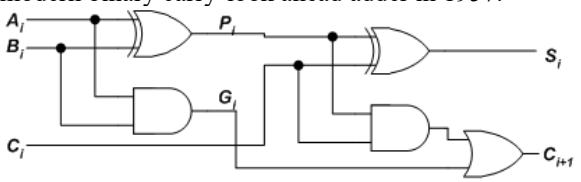
G. Zervakis, et al., "Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation" Approximate computing has received significant attention as a promising strategy to decrease power consumption of inherently error

tolerant applications. In this paper, we focus on hardware-level approximation by introducing the partial product perforation technique for designing approximate multiplication circuits. We prove in a mathematically rigorous manner that in partial product perforation, the imposed errors are bounded and predictable, depending only on the input distribution. Through extensive experimental evaluation, we apply the partial product perforation method on different multiplier architectures and expose the optimal architecture-perforation configuration pairs for different error constraints. We show that, compared with the respective exact design, the partial product perforation delivers reductions of up to 50% in power consumption, 45% in area, and 35% in critical delay. In addition, the product perforation method is compared with the state-of-the-art approximation techniques, i.e., truncation, voltage over scaling, and logic approximation, showing that it outperforms them in terms of power dissipation and error.

T. Yang, T. Ukezono, and T. Sato "A Low-Power High-Speed Accuracy-Controllable Approximate Multiplier Design",
Multiplication is a key fundamental function for many error-tolerant applications. Approximate multiplication is considered to be an efficient technique for trading off energy against performance and accuracy. This paper proposes an accuracy-controllable multiplier whose final product is generated by a carry-maskable adder. The proposed scheme can dynamically select the length of the carry propagation to satisfy the accuracy requirements flexibly. The partial product tree of the multiplier is approximated by the proposed tree compressor. An 8 × 8multiplier design is implemented by employing the carry maskable adder and the compressor. Compared with a conventional Wallace tree multiplier, the proposed multiplier reduced power consumption by between 47.3% and 56.2% and critical path delay by between 29.9% and 60.5%, depending on the required accuracy. Its silicon area was also 44.6% smaller. In addition, results from an image processing application demonstrate that the quality of the processed images can be controlled by the proposed multiplier design.
A. Cilardo, et al., "High-Speed Speculative Multipliers Based on Speculative Carry-Save Tree", Sacrificing exact calculations to improve digital circuit performance is at the foundation of approximate computing. In this paper, an approximate multiply-and-accumulate (MAC) unit is introduced. The MAC partial product terms are compressed by using simple OR gates as approximate counters; moreover, to further save energy, selected columns of the partial product terms are not formed. A compensation term is introduced in the proposed MAC, to reduce the overall approximation error. A MAC unit, specialized to perform 2D convolution, is designed following the proposed approach and implemented in TSMC 40nm technology in four different configurations. The proposed circuits achieve power savings more than 60%, compared to standard, exact MAC, with tolerable image quality degradation.
J. Liang, et al., "New Metrics for The Reliability of Approximate and Probabilistic Adders", Approximate/inexact computing has become an attractive approach for designing high performance and low power arithmetic circuits. Floating-point (FLP) arithmetic is required in many applications, such as digital signal processing, image processing and machine learning. Approximate FLP multipliers with variable accuracy are proposed in this paper; the accuracy and the circuit requirements of these designs are analyzed and assessed according to different metrics. It is shown that the proposed approximate FLP multiplier designs further reduce delay, area, power consumption and power-delay product (PDP) while incurring about half of the normalized mean error distance (NMED) compared with the previous designs. The proposed IFLPM24–15 is the most efficient design when considering both PDP and NMED. Case studies with three error-tolerant applications show the validity of the proposed approximate designs.

3. CARRY LOOK AHEAD ADDER

A carry-look ahead adder (CLA) or fast adder is a type of adder used in digital logic. A carry-look ahead adder improves speed by reducing the amount of time required to determine carry bits. It can be contrasted with the simpler, but usually slower, ripple-carry adder (RCA), for which the carry bit is calculated alongside the sum bit, and each stage must wait until the previous carry bit has been calculated to begin calculating its own sum bit and carry bit. The carry-look ahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger-value bits of the adder. The Kogge–Stone adder (KSA) and Brent–Kung adder (BKA) are examples of this type of adder.
Charles Babbage recognized the performance penalty imposed by ripple-carry and developed mechanisms for anticipating carriage in his computing engines.[1] Gerald B. Rosenberger of IBM filed for a patent on a modern binary carry-look ahead adder in 1957.



The following are the methods to get the high speed in the parallel adder to produce the binary addition.
1. By employing faster gates with reduced delays, we can reduce the propagation delay. But there will be a capability limit for every physical logic gate.

2. Another way is to increase the circuit complexity in order to reduce the carry delay time. There are several methods available to speeding up the parallel adder, one commonly used method employs the principle of look ahead-carry addition by eliminating inter stage carry logic.

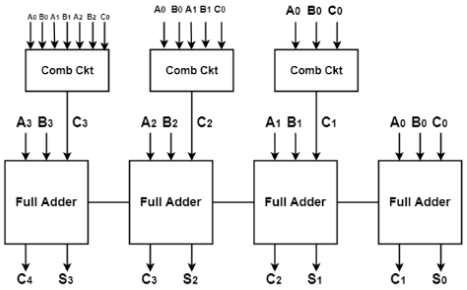


Fig :4 bit carry look ahead adder

3. A carry-Lookahead adder is a fast parallel adder as it reduces the propagation delay by more complex hardware, hence it is costlier. In this design, the carry logic over fixed groups of bits of the adder is reduced to two-level logic, which is nothing but a transformation of the ripple carry design.
This method makes use of logic gates so as to look at the lower order bits of the augend and addend to see whether a higher order carry is to be generated or not.

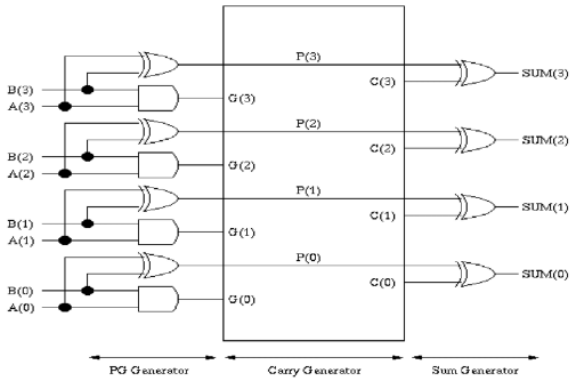


Fig :4 bit carry look ahead adder

4. PROPOSED ACCURACY-CONFIGURABLE ADDER

Operation Of the Proposed Adder

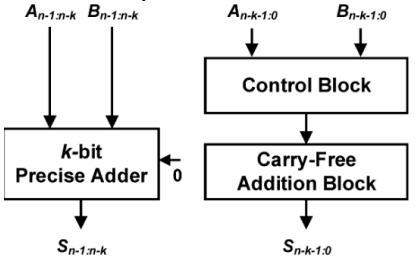


Fig1 : block diagram of proposed approximate adder

Figure 1 shows the operation of the proposed adder with 16-bit inputs. An n-bit addition requires two different kinds of additions. The precise part is obtained using a k-bit accurate adder (e.g., RCA or CLA) for k MSBs, and the approximate part is obtained via OR and XOR operations of the remaining (n−k) LSBs, where k < n. The example in Figure 3 splits a 16-bit input into an 8-bit precise part and an 8-bit approximate part. It is worth mentioning that the bit-widths of two parts do not need to be equal. The precise part performs an accurate addition using k MSB inputs, and the carry-in signal (Cin) is predicted via an AND operation of two (n−k−1) th inputs (i.e., Cin = An−k−1 AND Bn−k−1). Similar to the LOA, the approximate part leverages the OR function to add the lower order input bits of two operands; however, the main difference with the LOA is that, in the proposed adder, the bit operation of (n−k−1)th inputs is replaced with the XOR operation. This results in the formation of a half adder for (n−k−1)th LSB inputs and effectively extends the length of accurate addition by one bit, which consequently leads to an overall improvement in accuracy compared to that of the LOA. In other words, the proposed n-bit adder always generates correct outputs for bit positions from (n−1) to (n−k−1) by replacing the OR gate with the XOR gate at the (n−k−1)th LSB position.

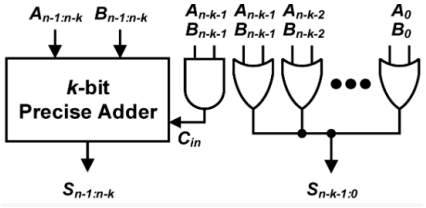


Fig 3: Proposed Hybrid Error Reduction Scheme

The proposed approximate adder performs an error reduction to further decrease the output errors when both (n−k−2)th input bits are “1” (i.e., An−k−2 = 1 and Bn−k−2 = 1). Otherwise, it does not perform any error reduction, as shown in the example in Figure 3. In fact, our adder implements the error reduction logic, but this does not affect the final outputs of the adder in this case. The proposed hybrid error reduction is performed differently depending on the (n−k−1)th inputs. In other words, the proposed adder checks the (n−k−1)th output bit to determine which of the two error reduction schemes is applicable.

5. MOTIVATION BEHIND REVERSIBLE LOGIC

High-performance chips releasing large amounts of heat impose practical limitation on how far can we improve the performance of the system. Reversible circuits that conserve information, by un-computing bits instead of throwing them away, will soon offer the only physically possible way to keep improving performance. Reversible computing will also lead to improvement in energy efficiency. Energy efficiency will fundamentally affect the speed of circuits such as nanocircuits and therefore the speed of most computing applications. To increase the portability of

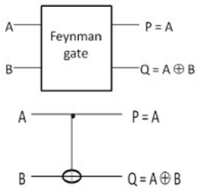
devices again reversible computing is required. It will let circuit element sizes to reduce to atomic size limits and hence devices will become more portable. Although the hardware design costs incurred in near future may be high but the power cost and performance being more dominant than logic hardware cost in today's computing era, the need of reversible

REVERSIBLE GATES

Reversible circuits provide a one-to-one relation between inputs and outputs; therefore, inputs can be recovered from outputs. This interesting feature results in significant power saving in digital circuits. Classical digital gates are not reversible, reversible gates should be designed as basic components to design logical reversible circuits. Well known reversible gates are Feynman, Peres and HNG. The Feynman or controlled not (CNOT) gate is frequently used in reversible circuits, since it can provide exclusive OR (XOR) as well as copy and complement of the input. Since reversible circuits do not take advantage of fan-out, this gate can be used to achieve two copies of the same input by setting the other input of the gate to the zero-logic level. Similarly, by setting the second input of the CNOT to one-logic level, we can achieve the complement of the other input.

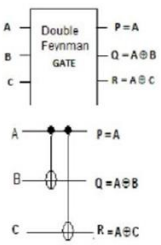
Feynman Gate

• Feynman gate is a 2*2 one through reversible gate as shown in figure 1.
• The input vector is I(A, B) and the output vector is O(P, Q).
• The outputs are defined by P=A, Q=A xor B. Quantum cost of a Feynman gate is 1.
• Feynman Gate (FG) can be used as a copying gate. Since a fan-out is not allowed in reversible logic, this gate is useful for duplication of the required outputs.



A	B	P	Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

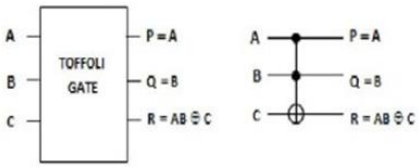
Double Feynman Gate (F2G)



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	0	0

Fig.2 shows a 3*3 Double Feynman gate.
• The input vector is I (A, B, C) and the output vector is O (P, Q, R). The outputs are defined by P = A, Q=A xor B, R=A xor C.
• Quantum cost of double Feynman gate is 2.

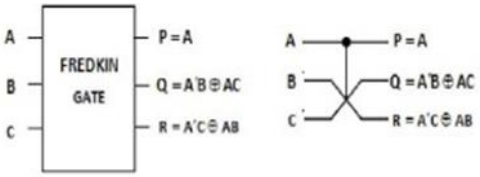
Toffoli Gate:



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Fig 3 shows a 3*3 Toffoli gate.
• The input vector is I (A, B, C) and the output vector is O(P,Q,R).
• The outputs are defined by P=A, Q=B, R=AB xor C.
• Quantum cost of a Toffoli gate is 5.

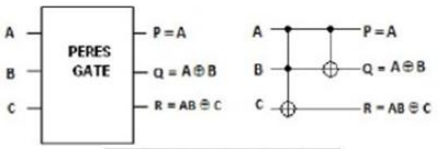
Fredkin Gate



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

Fig 4 shows a 3*3 Fredkin gate.
• The input vector is I (A, B, C) and the output vector is O (P, Q, R).
• The output is defined by P=A, Q=A′B xor AC and R=A′C xor AB.
• Quantum cost of a Fredkin gate is 5.

Peres Gate

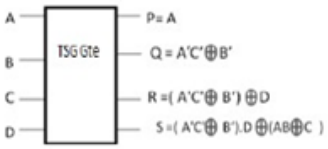


A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Fig 5 shows a 3*3 Peres gate.
• The input vector is I(A, B, C) and the output vector is O (P, Q, R).

• The output is defined by P = A, Q = A xor B and R=AB xor C.
• Quantum cost of a Peres gate is 4.
• In the proposed design Peres gate is used because of its lowest quantum cost.

TSG gate



• The input vector is I (A,B, C, D) and the output vector is O (P, Q, R, S).
• The output is defined by P = A, Q = A'C' xor B', R = (A'C' xor B')xor D and
• S =(A'C' xor B'). D xor (AB xor C) .
• Quantum cost of a Peres gate is 4.
• In the proposed design Peres gate is used because of its lowest quantum cost.
• It can be verified that the input pattern corresponding to a particular output pattern can be uniquely determined.
• The proposed TSG gate is capable of implementing all Boolean functions and can also work singly as a reversible Full adder.

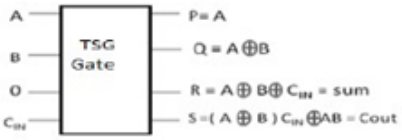
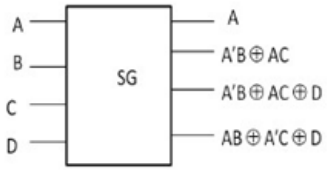
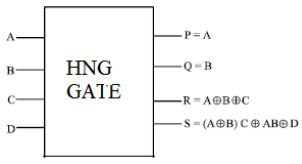


Fig 7 shows a 4*4 TSG gate.

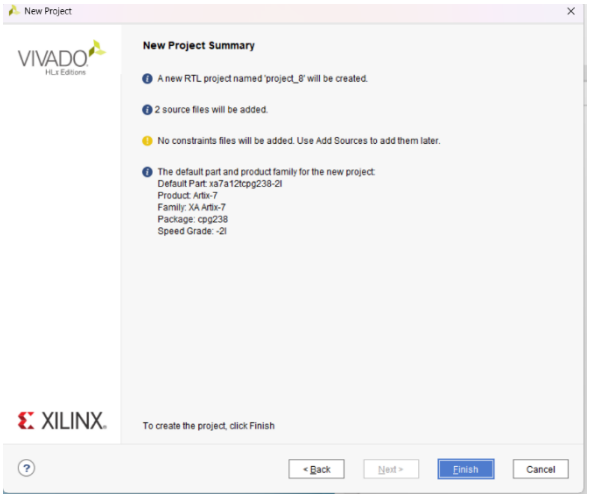
Sayem gate
• SG is a 1 trough 4x4 reversible gate.
• The input and output vector of this gate are, Iv = (A, B, C, D) and
• Ov = (A, A'B xor AC, A'B xor AC xor D, AB xor A'C xor D).
• The block diagram of this gate is shown in Fig .

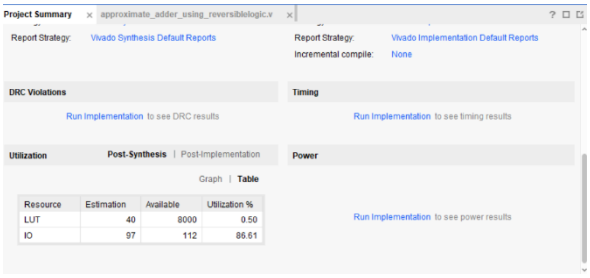


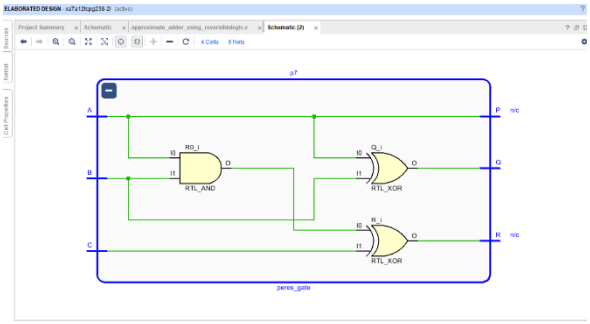
HNG Gate
A reversible logic gate has the same number of inputs and outputs with one-to-one correspondence between the input and the output. Hybrid new gate (HNG) is a universal reversible gate and realizes all Boolean functions.

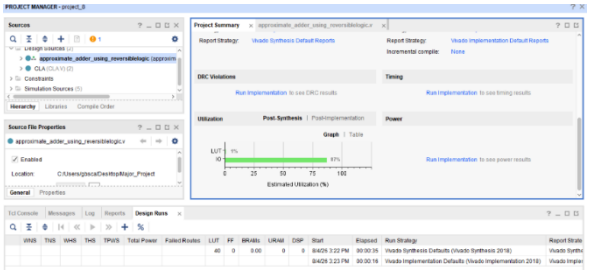


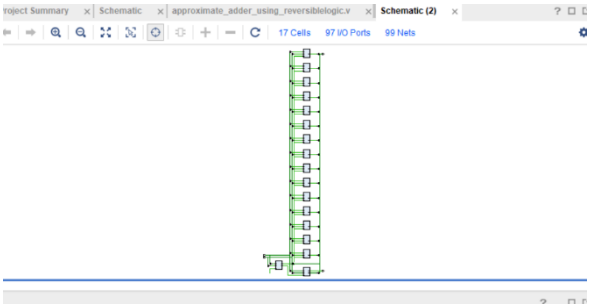
6. RESULTS

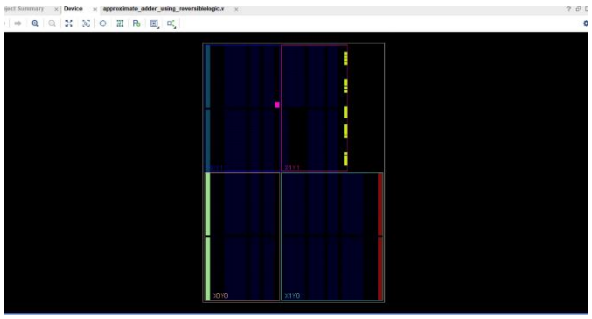


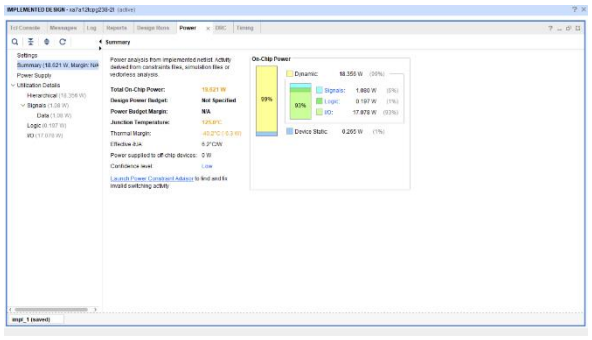


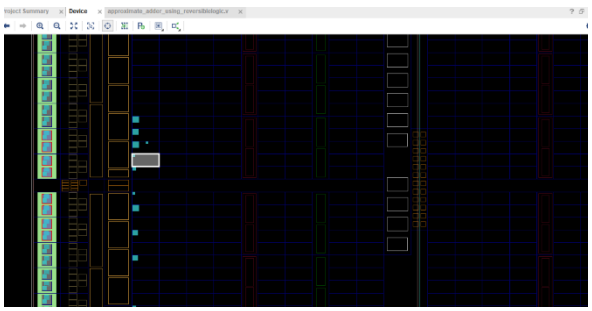


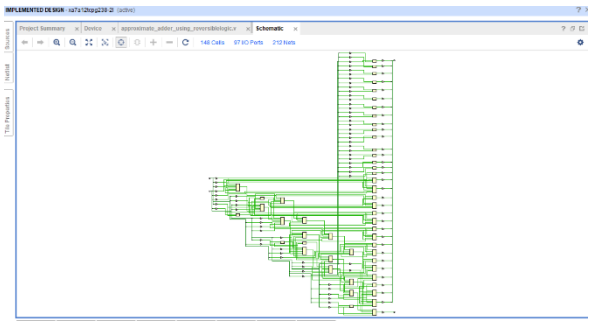


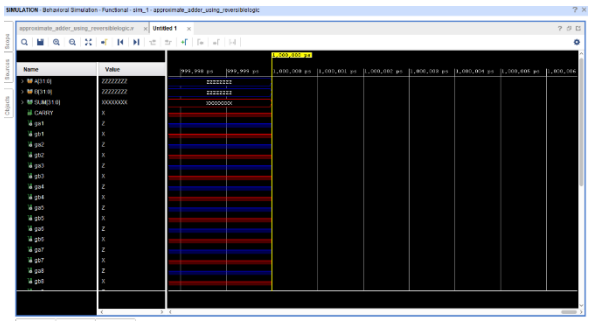












7. CONCLUSION

In this project, the proposed approximate configurable adder is better than the existing Carry Look Ahead adder. The proposed adder design-based approximation concept using reversible logic gates, and its configurability of accuracy is realized by masking the carry propagation at runtime. The experimental results demonstrate that the proposed adder delivers significant speedup with a small area and less power overhead than CLA. Furthermore, compared with previously studied configurable adders, the experimental results demonstrate that the proposed adder achieves the original purpose of delivering an unbiased optimized result between area without sacrificing accuracy. It was also found that the quality requirements of the evaluated application were not compromised.

Then we evaluated the power consumption, critical path delay, and design area for each of these implementations. Compared with the conventional CLA, with 1.95% mean relative error distance (MRED), the proposed adder reduced power consumption and critical path delay by 42.7% and 56.9%, respectively. We provided a crosswise comparison to demonstrate the superiority of the proposed adder. Moreover, we implemented two previously studied configurable adders to evaluate power consumption, critical path delay, design area, and accuracy. We also evaluated the quality of these accuracy configurable adders in a real image processing application.

FUTURE SCOPE:

Approximate computing is a computation technique which returns a possibly inaccurate result rather than a guaranteed accurate result, and can be used for applications where an approximate result is sufficient for its purpose. In future These kind of adders used in image processing, filters and cryptographic applications for security reasons. One example of such situation is for a search engine where no exact answer may exist for a certain search query and hence, many answers may be acceptable. Similarly, occasional dropping of some frames in a video application can go undetected due to perceptual limitations of humans. Approximate computing is based on the observation that in many scenarios, although performing exact computation requires large number of resources, allowing bounded approximation can provide disproportionate gains in performance and energy, while still achieving acceptable result accuracy

REFERENCES

[1] S. Cotofana, C. Lageweg, and S. Vassiliadis, “Addition related arithmetic operations via controlled transport of charge”, IEEE Transactions on Computers, vol. 54, no. 3, pp. 243-256, Mar. 2005.

[2] V. Beiu, S. Aunet, J. Nyathi, R. R. Rydberg, and W. Ibrahim, “Serial Addition: Locally Connected Architectures”, IEEE Transactions on Circuits and Systems-I: Regular projects, vol. 54, no. 11, pp. 2564-2579, Nov. 2007.

[3] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Quality programmable vector processors for approximate computing", 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp. 1-12, Dec. 2013.
[4] A. B. Kahng, and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs", IEEE/ACM Design Automation Conference (DAC), pp. 820-825, Jun. 2010.
[5] R. Ye, T. Wang, F. Yuan, R. Kumar, and Q. Xu, "On Reconfiguration- Oriented Approximate Adder Design and Its Application", IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 48-54, Nov. 2013.
[6] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-Power digital signal processing using approximate adders", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124-137, Jan. 2013.
[7] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-Inspired imprecise computational blocks for efficient VLSI implementation of Soft-Computing applications", IEEE Transactions on Circuits and Systems I: Regular projects, vol. 57, no. 4, pp. 850-862, Apr. 2010.
[8] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO: modeling and analysis of circuits for approximate computing", IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 667-673, Nov. 2011.
[9] NanGate, Inc. NanGate FreePDK45 Open Cell Library, http://www.nangate.com/?page_id=2325, 2008 [10] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders", IEEE Transactions on Computers, Vol. 62, no. 9, pp. 1760-1771, Sep. 2013.
[11] M. S. Lau, K. V. Ling, and Y. C. Chu, "Energy-Aware probabilistic multiplier: Design and Analysis", International Conference on Compliers, architecture, and synthesis for embedded systems, pp. 281-290, Oct. 2009.
[12] T. Yang, T. Ukezono, and T. Sato, "A Low-Power Configurable Adder for Approximate Applications", 19th International Symposium on Quality Electronic Design, Mar. 2018.