A Comparative Exploration About Approximate Full Adders for Error Tolerant Applications

M. Priyadharshni and S. Kumaravel(✉)

VIT, Vellore, Tamil Nadu, India
{priyadharshni.m,kumaravel.s}@vit.ac.in

Abstract. Arithmetic units such as adders and multipliers play an essential role in the performance of Digital signal processor (DSP) systems. The efficiency of the Processors are influenced by the speed and power consumption of arithmetic units. It is improved by adopting approximate computing in arithmetic units with acceptable degradation in the output. Approximate computing is an emerging topic in the past decades, it aims to achieve promising design approach with the sacrifices in computational quality for error resilient applications. Approximate computing can be adopted both in hardware level and software level of research. This paper provides an elaborative investigation about approximate computing on full adders which is explored at the hardware level. The approximation is applied to full adders either at gate level or transistor level. Further, ripple carry adder is designed for varying bit width with different degrees of approximation using these approximate Full adders. Ripple carry adder is estimated based on the structural analysis such as Area, Delay Product (ADP) and Power, Delay Product (PDP) and error matrix such as pass rate, error rate, Normalized Error Distance (NED) and Mean Error Distance (MED). The ripple carry adders are designed in Verilog HDL and stimulated in Synopsys Design Compiler (DC) using tsmc 65 nm standard cell library typical corner whereas, the error characteristics is done in MATLAB.

Keywords: Approximate computing · Approximate full adder · Error matrix · Ripple carry adder · Structural analysis

1 Introduction

DSP Processors are widely used in digital image/signal processing applications which are highly error tolerant. Error-tolerant applications will accept the output with some degrees of degradation since these applications are limited to human perception. The arithmetic and logical operations are mostly performed in DSP by using arithmetic units (ALU). The complexity of arithmetic and numerical

Supported by organization VIT.

operations are also determined by the ALU such as adders and multipliers. So the adders and multipliers play major in designing DSP processors where we can adopt the approximate computing. The main motivation of approximate computing [5] is to achieve best design approach with a reduction in the complexity but with the trade-off accuracy in the results for error resilient applications.

The approximation is done in two levels, both in the hardware level and software level of abstraction [5]. In the software level of approximation, some levels in the algorithms are eliminated and at the hardware level, the design of circuits is modified. By adopting the approximation at the hardware level, Approximate adders/multipliers are designed which are used in the processors to overcome the major drawback of DSP such as system complexity and power dissipation. Adders have attracted researcher's attention in designing approximation since adders are the most fundamental data operator. The key motivation in designing the approximate adders is to reduce the lifetime of carry propagation and the circuit complexity. The lifetime of carry propagation is depended on the length of the critical path. The longest lifetime of carry propagation is approximately equal to log2(n) [7] and less than log2(n)+12 [9]. In the worst case, lifetime of carry propagation is equal to N (rarely happens).

In this paper, approximate adders are evaluated based on their structural analysis and error metrics. Structural analysis are related to hardware related figures of merit like gate count [8], area, power and delay including the compound metrics such as area-delay product (ADP) and power-delay product (PDP). The error characteristics [4] of approximate adders are determined based on their Pass rate, error rate, error distance, mean error distance (MED) and normalized error distance (NED).

This paper makes the following contributions.

1. A collection of approximate adders which include recent designs are considered.
2. Approximate adders are evaluated based on their gate count, pass rate and error rate.
3. Further, the evaluation includes designing the ripple carry adder using approximate adders in the given set. Ripple carry adder is designed for varying bit width with different degrees of approximation.
4. Ripple carry adders are evaluated based on their design metrics and error metrics by using random 1 lakh test stimulus vectors.

2 Approximate Adder

Two methodologies are used in redesigning the full adder.

1. Approximation in the gate level of full adder.
2. Approximation in the transistor level of full adder.

2.1 Approximation in the Gate Level of Full Adder

Inexact Full Adders (INXA). Three inexact full adders [1] are implemented by applying the approximation in the gate level and they are implemented with less number of transistors. In these inexact adders one of the full adder output is approximated and the other output has the accurate result.

INXA1. This adder is designed by maintaining the accurate sum signal and the carry signal is approximated. The gate level implementation of INXA1 is shown in the Fig. 1 and the truth table is given in the Table 1 (col 14 and 15). In the truth table a tick mark represents the accurate output is equal to the exact output and the cross mark represents the unequal i.e erroneous outputs. The INXA1 produces the accurate sum and 2 error out of 8 cases in carry signal.

Gate Count Calculation: The implementation of 1-bit INAX1 requires 2 XOR gate and 1 NOT gate. For the sake of comparison, assume that 1 basic gate [8] is equal to 1 gate count and the number of gate count is used to determine the area of the adders. Then XOR gate is limited to 2-input basic gates (AND,OR and NOT), it requires 2 AND gate, 1 OR gate and 1 NOT gate [6]. Then INXA1 is implemented by using 4 AND gate, 2 OR gate and 2 NOT gate.

Total number of basic gates used in INXA1 = 9 basic gates.

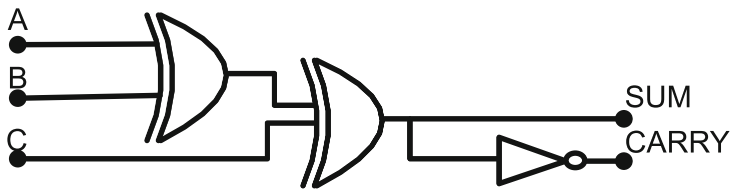

Fig. 1. INXA1

INXA2. INXA2 is formulated by maintaining the accurate results in the carry signal and approximate result in the sum signal. In INXA2, the second XOR gate in the exact full adder is replaced with OR gate. The gate level implementation of INXA2 is shown in Fig. 2. The truth table of INXA2 in Table 1 (col 16 and 17), which shows that the sum signal has 2 errors out of 8 cases whereas carry signal is remained with accurate output.

Gate Count Calculation: 1-bit INXA2 requires 1 XOR gate, 2 OR gate and 2 AND gate. When the XOR is limited to 2-input basic gates then INXA2 is implemented by using 4 AND gate, 1 NOT gate and 3 OR gate.

Total number of basic gates used in INXA2 = 8 basic gates.

INXA3. INXA3 is demonstrated by applying approximation in the sum signal and carry signal remains as exact. The gate level diagram of INXA3 is shown in Fig. 3 and the truth table is shown in Table 1 (col 18 and 19). INAX3 produces 2 errors out of 8 cases in the sum signal.

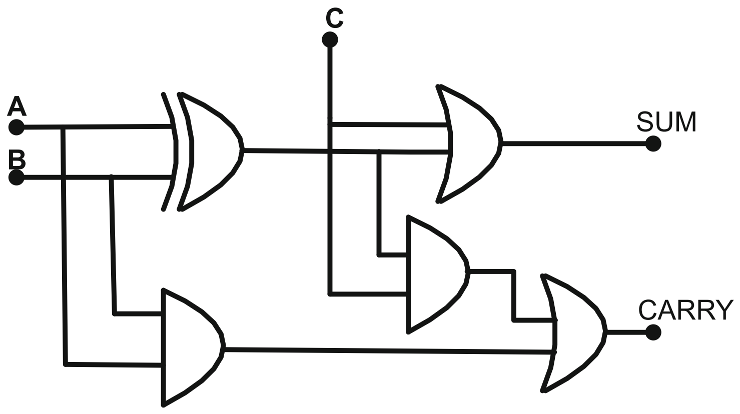

Fig. 2. INXA2

Gate Count Calculation: INXA3 utilizes 1 XOR gate, 2 AND gate, 1 OR gate and 1 NOT gate. After the limitation on XOR gate then INXA3 is redesigned by using 4 AND gate, 2 NOT gate and 2 OR gate.

Total number of basic gates used in INXA3 = 8 basic gates.

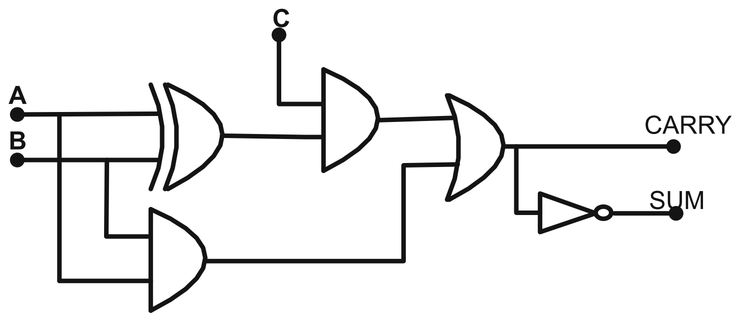

Fig. 3. INXA3

Approximate Full Adders (AFA). AFAs [2] are constructed by modifying the conventional Full adder in the gate level and they also designed as the output of carry signal is independent of carry input. The common approach used in these AFAs are to reduce the length of carry lifetime when it is implemented in the multibit adders. AFAs are implemented by applying approximation only in the carry signal.

AFA_1 and AFA_2. In AFA_1, the output of carry signal is considered as the input of A. In AFA_2, the input of B is considered as the output of carry signal. AFA_1 and AFA_2 produces 2 error out of 8 cases in the carry signal. The gate level logic implementation of AFA_1 and AFA_2 is shown in the Figs. 4 and 5 and the truth table of AFA_1 and AFA_2 is shown in the Table 1 (column 6 and 7) and (column 8 and 9) respectively.

Gate Count Calculation: Both AFA_1 and AFA_2 are realized by using 2 XOR gate. Due to the limitation, they bring about by using 4 AND gate, 2 OR gate and 2 NOT gate.

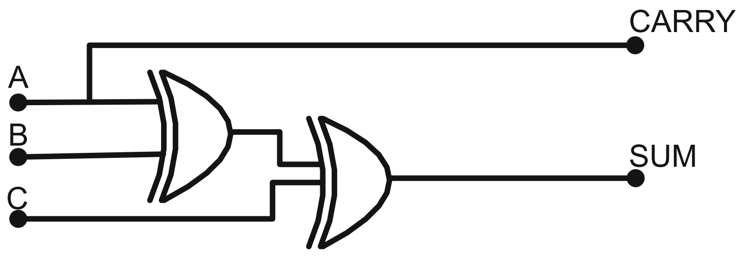

Fig. 4. AFA_1

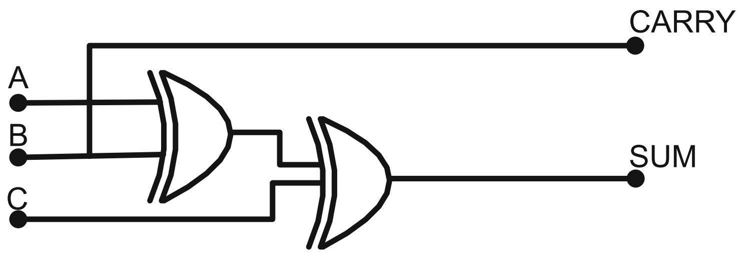

Fig. 5. AFA_2

The total number of basic gates used in AFA_1 and AFA_2 = 8 basic gates.
AFA_3. In AFA_3, the output of carry signal is viewed as A.B. The truth table of AFA_3 is shown in the Table 1 (Column 10 and 11) and gate level implementation is shown in the Fig. 6.

Gate Count Calculation: AFA_3 is performed by using 2 XOR gate and 1 AND gate. After the limitation applied to the XOR gate, AFA_3 is realized by using 5 AND gate, 2 OR gate and 2 NOT gate.

The total number of basic gates used in AFA_3 = 9 basic gates.
AFA_4. In AFA_4, the output of carry signal is executed as A+B. The gate level circuit diagram is shown in Fig. 7 and the truth table is shown in Table 1 (column 12 and 13).

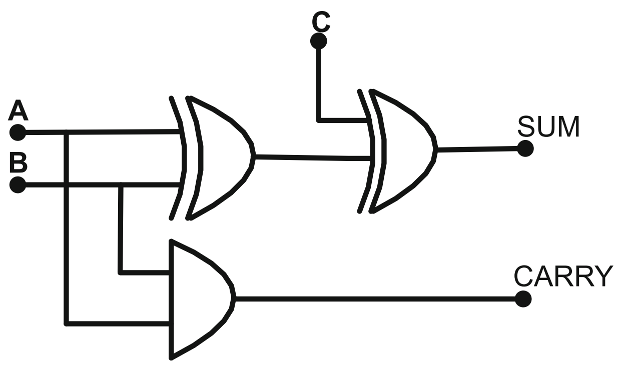

Fig. 6. AFA_3

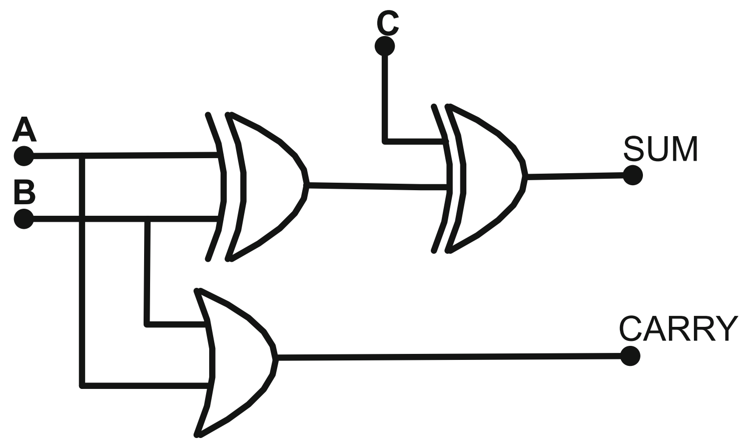

Fig. 7. AFA_4

Gate Count Calculation: 2 XOR gate and 1 OR gate are used to construct AFA_4. 4 AND gate, 3 OR gate and 2 NOT gate are utilized to construct AFA_4 after the limitation applied over the XOR gate.

The total number of basic gates used in $AFA_4 = 9$ basic gates.

Table 1. Truth table of approximate full adders.

Inputs			Exact outputs		AFA_1		AFA_2		AFA_3		AFA_4		INXA1		INXA2		INXA3	
x1	x2	x3	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum
0	0	0	0	0	0 ✓	0 ✓	0 ✓	0✓	0✓	0✓	0✓	0✓	0 ✓	0 ✓	0 ✓	0✓	0✓	1✗
0	0	1	0	1	0 ✓	1 ✓	0 ✓	1✓	0✓	1✓	0✓	1✓	1 ✗	1 ✓	0 ✓	1✓	0✓	1✓
0	1	0	0	1	0 ✓	1 ✓	1 ✗	1✓	0✓	1✓	1✗	1✓	0 ✓	1 ✓	0 ✓	1✓	0✓	1✓
0	1	1	1	0	0 ✗	0 ✓	1 ✓	0✓	0✗	0✓	1✓	0✓	1 ✓	0 ✓	1 ✓	1✗	1✓	0✓
1	0	0	0	1	1 ✗	1 ✓	0 ✓	1✓	0✓	1✓	1✗	1✓	0 ✓	1 ✓	0 ✓	1✓	0✓	1✓
1	0	1	1	0	1 ✓	0 ✓	0 ✗	0✓	0✗	0✓	1✓	0✓	1 ✓	0 ✓	1 ✓	1✗	1✓	0✓
1	1	0	1	0	1 ✓	0 ✓	1 ✓	0✓	1✓	0✓	1✓	0✓	0 ✗	0 ✓	1 ✓	0✓	1✓	0✓
1	1	1	1	1	1 ✓	1 ✓	1 ✓	1✓	1✓	1✓	1✓	1✓	1 ✓	1 ✓	1 ✓	1✓	1✓	0✗

2.2 Approximation in the Transistor Level of Fulladder

The reduction in circuit complexity, node capacitance, and dynamic power dissipation are achieved by removing some of the transistors from the transistor level representation of the accurate full adder. For the sake of comparison, all the approximate full adders are represented in gate level and approximate adders with minimum error are considered for the exploration.

Approximate Mirror Adder. Different Approximate adders are acquired by extracting some of the transistors from the economical representation of a full adder called mirror adder (MA) [3]. When removing some transistors from MA it results shorter delay (i.e) faster charging and discharging of node capacitance is achieved.

AMA1. AMA1 is implemented by using eight transistors which are accomplished by removing some of the transistors from conventional MA. It is done by verifying that extraction process does not result in the short or open circuit in AMA1. AMA1 produces 2 errors out of 8 cases in the sum signal and one error out of 8 cases in the carry signal.

Gate Count Calculation: The AMA1 is achieved by using 3 AND gate, 2 OR gate and 1 NOT gate.

The total number of basic gates used in AMA1 = 6 basic gates (Fig. 8).

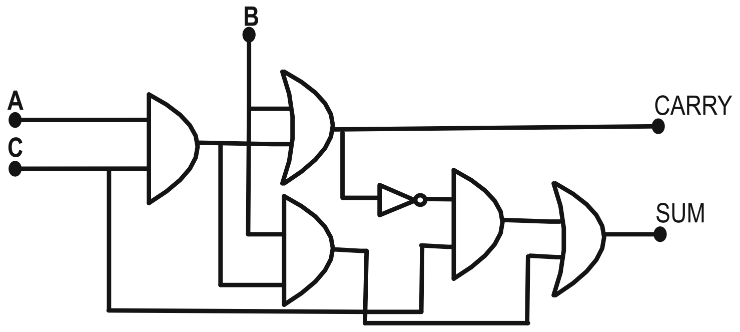

Fig. 8. *AMA*1

AMA2. AMA2 is implemented as carry signal is computed first then a buffer is added to get the sum signal. The output of sum signal is equal to complimentary of carry signal for 6 cases out of 8 cases with an error in 2 cases. The output carry signal remains as accurate. The truth table of AMA2 is shown in table and gate level representation is shown in the figure.

Gate Count Calculation: The gate level characterization of AMA2 is based on its Transistor representation and it is achieved by using 2 AND gate, 2 OR gate and 1 NOT gate.

The total number of basic gates used in AMA2 = 5 basic gates (Fig. 9).

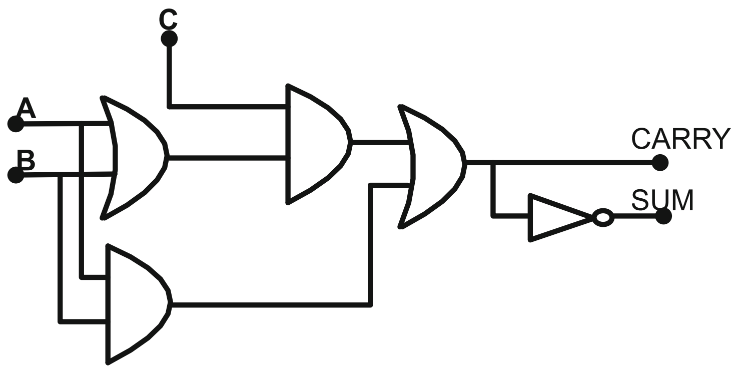

Fig. 9. *AMA*2

Approximate XNOR/XOR Based Adder (AXA). AXA [10] is proposed by approximating XOR/XNOR based exact full adder. AXA achieves less transistor count and reduction in power dissipation with the trade-off accuracy when compared to the exact full adder.

AXA1. AXA1 is implemented with 6 transistors, 4 transistor for XNOR and 2 for pass transistor logic. In AXA1 the output of XNOR gate is equal to the output of the sum signal which leads to 4 errors out of 8 cases in sum signal whereas carry signal is remained accurate by cascading the output of XNOR gate with pass transistor logic.

Gate Count Calculation: The gates utilized for implementing AXA1 is 1 XOR gate, 1 NOT gate, 1 OR gate, and 2 AND gate. When the XOR is limited to the 2-input basic gate, the total gates utilized for AXA1 is 2 NOT gate, 2 OR gate, and 4 AND gate.

The total number of basic gates utilized for AXA1 = 8 basic gates (Fig. 10).

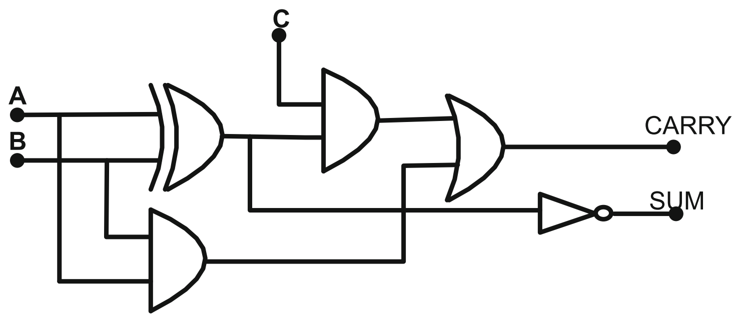

Fig. 10. $AXA1$

AXA2. The design of AXA2 is the extension of AXA1. In order to reduce the error in the sum signal, a pass transistor configuration (2 transistors) is used, which leads to 2 errors out of 8 cases in sum signal. The carry signal remained same.

Gate Count Calculation: AXA2 requires 1 XOR gate, 1 NOT gate, 1 OR gate, and 3 AND gate for the gate level characterization of AXA2. Due to the limitation, 2 NOT gate, 2 OR gate and 5 AND gate are utilized for the demonstration of AXA2.

The total basic gates used in AXA2 = 9 basic gates (Fig. 11 and Table 2).

3 Structural Analysis of Approximate Adder

Ripple Carry adders for n = 20 are designed using approximate adders with different degrees of full adder approximation (DFA) for example: 12:8 the first 8 LSB adders are approximate adders and the remaining adders are accurate

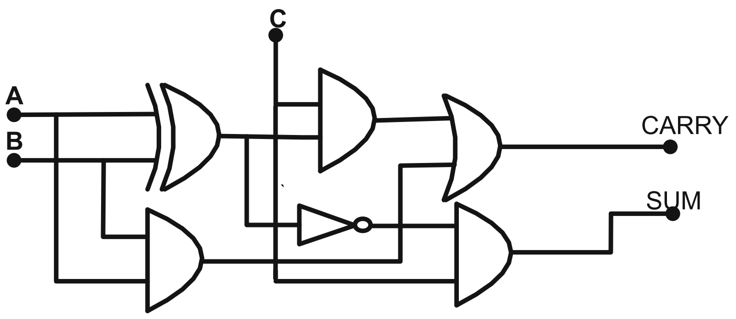

Fig. 11. $AXA2$

Table 2. Truth table of approximate full adders.

Inputs			Exact outputs		AMA1		AMA2		AXA1		AXA2	
x1	x2	x3	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum	Carry	Sum
0	0	0	0	0	0 ✓	0 ✓	0 ✓	1✗	0✓	1✗	0✓	0✓
0	0	1	0	1	0 ✓	1 ✓	0 ✓	1✓	0✓	1✓	0✓	1✓
0	1	0	0	1	1 ✗	0 ✗	0 ✓	1✓	0✓	0✗	0✓	0✗
0	1	1	1	0	1 ✓	0 ✓	1 ✓	0✓	1✓	0✓	1✓	0✓
1	0	0	0	1	0 ✓	0 ✗	0 ✓	1✓	0✓	0✗	0✓	0✗
1	0	1	1	0	1 ✓	0 ✓	1 ✓	0✓	1✓	0✓	1✓	0✓
1	1	0	1	0	1 ✓	0 ✓	1 ✓	0✓	1✓	1✗	1✓	0✓
1	1	1	1	1	1 ✓	1 ✓	1 ✓	0✗	1✓	1✓	1✓	1✓

adders. The designed adders are coded in Verilog HDL and simulated using Verilog compiler simulator (VCS). These different adders are synthesized in Synopsys Design compiler (DC) using TSMC 65nm standard cell library typical corner. The switching activity of ripple carry adder for 1 lakh random test stimulus vectors are recorded in the .saif file and used in DC for detailed power analysis. Area, Power, and delay obtained from the DC compiler, which promotes the comparative study of the approximate adders. The structural analysis of approximate adders is based on their gate count, Area and Delay Product (ADP), Power and Delay Product (PDP) and Area and Power Product (APP). Table 3 gives the detailed analysis of ripple carry adder with different of approximation (DAR). ADP, PDP, APP values are represented in Figs. 12, 13 and 14 respectively. AFA shows attains lowest values in the compound metrics by maintaining the equal gate count when compared to other approximate adders in the set.

4 Error Analysis of Approximate Adder

Pass rate, error rate, mean error distance (MED) and normalized error distance (NED) are the error metrics considered for the evaluation of error characteristics of the approximate adder. Error distance is given as difference between the exact results and accurate results. MED is the mean value of error distance.

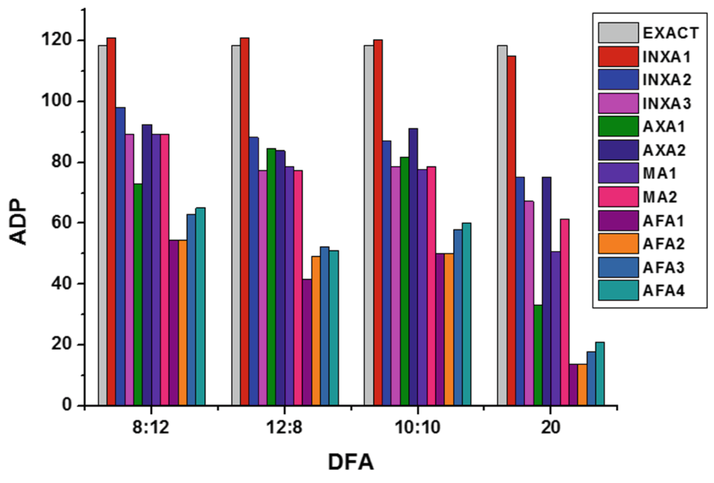

Fig. 12. ADP

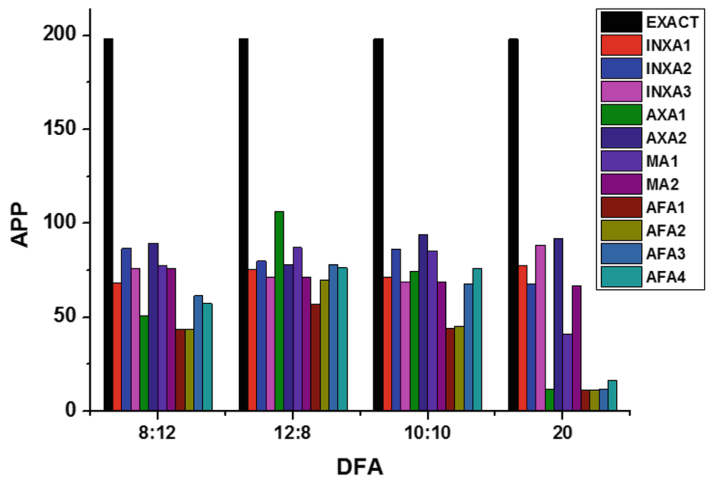

Fig. 13. APP

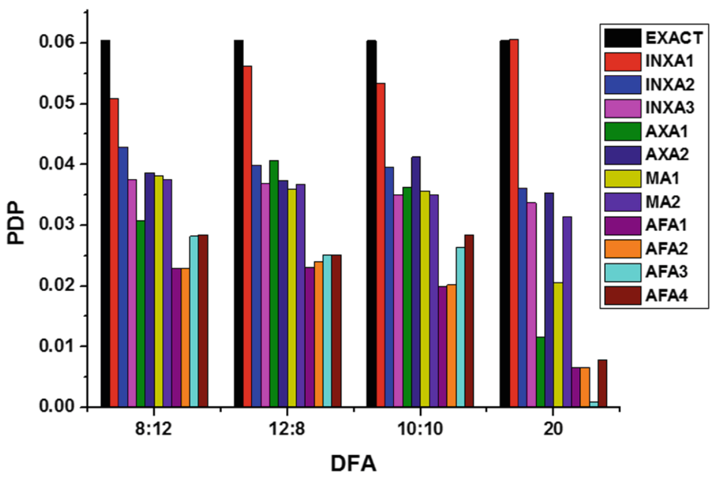

Fig. 14. PDP

Table 3. Structural comparison of ripple carry adder using approximate adders for different degrees of approximation

Designs	Degrees of approximation for N = 20	Area	Delay	Power
Exact	20	623.2	0.19	0.3179
	10:10	623.2	0.19	0.3179
	8:12	623.2	0.19	0.3179
	12:8	623.2	0.19	0.3179
INXA1	20	383.2	0.3	0.202
	10:10	400.8	0.3	0.1782
	8:12	402.8	0.3	0.1696
	12:8	402.8	0.3	0.1874
INXA2	20	375.2	0.2	0.1804
	10:10	436	0.2	0.1976
	8:12	445.2	0.22	0.1947
	12:8	420.4	0.21	0.1901
INXA3	20	420.4	0.16	0.2106
	10:10	392.8	0.2	0.0.1752
	8:12	425.2	0.21	0.1787
	12:8	386.8	0.2	0.1842
AFA1	20	151.2	0.09	0.0732
	10:10	334.4	0.15	0.1324
	8:12	321.2	0.17	0.1351
	12:8	383.2	0.13	0.1769
AFA2	20	151.2	0.09	0.0736
	10:10	334.4	0.15	0.135
	8:12	321.2	0.17	0.1352
	12:8	378	0.13	0.1851
AFA3	20	199.2	0.09	0.00962
	10:10	385.2	0.15	0.17534
	8:12	370	0.17	0.1658
	12:8	402.8	0.13	0.1931
AFA4	20	210.8	0.1	0.0785
	10:10	400.8	0.15	0.1895
	8:12	361.6	0.18	0.1583
	12:8	393.6	0.13	0.1937
AMA1	20	317.6	0.16	0.1285
	10:10	430.8	0.18	0.1979
	8:12	425.6	0.21	0.1817
	12:8	436	0.18	0.1996
AMA2	20	361.6	0.17	0.1847
	10:10	392.8	0.2	0.1752
	8:12	425.2	0.21	0.1787
	12:8	386.8	0.2	0.1841
AXA1	20	183.6	0.18	0.0645
	10:10	409.2	0.2	0.1813
	8:12	348	0.21	0.1464
	12:8	469.6	0.18	0.2264
AXA2	20	442	0.17	0.2079
	10:10	455.6	0.2	0.2062
	8:12	462.4	0.2	0.1932
	12:8	418.8	0.2	0.1869

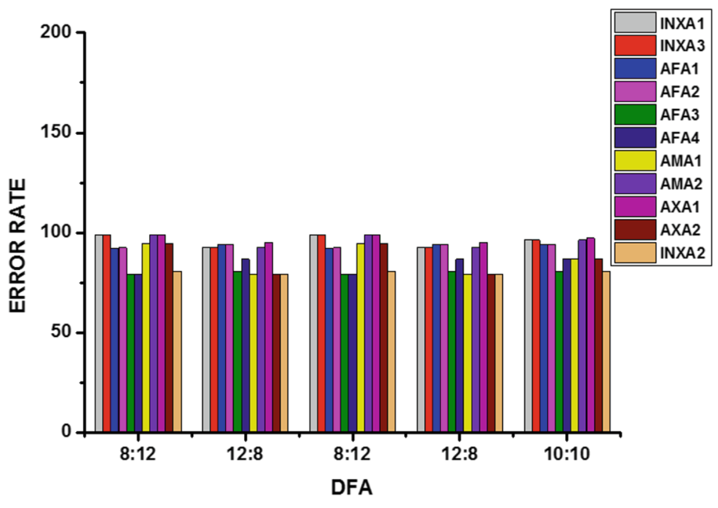

Fig. 15. Error rate.

$$NED = \frac{Mean\ value\ of\ Error\ Distance}{Magnitude\ of\ maximum\ possible\ error} \tag{1}$$

$$Pass\ Rate = \frac{Number\ of\ correct\ outputs}{Total\ number\ of\ outputs} \tag{2}$$

$$Error\ Rate = \frac{Number\ of\ incorrect\ outputs}{Total\ number\ of\ outputs} \tag{3}$$

1 lakh random test stimulus vector are generated and recorded for detailed error analysis, which is done in MATLAB. Pass rate and error rate of all the adders are represented in Figs. 15 and 16 respectively. Table 4 provides the NED values of ripple carry adder.

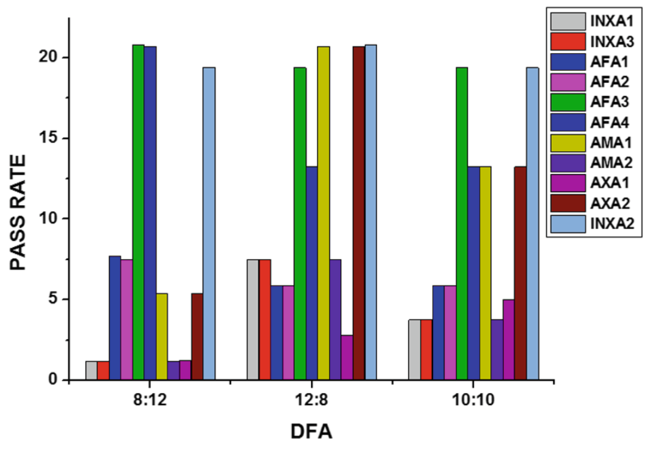

Fig. 16. Pass rate.

Table 4. Ned values of ripple carry adder for different degrees of approximation.

Designs	N = 20	N = 10:10	N = 8:12	N = 12:8
INXA1	4.685×10^{-1}	6.17×10^{-2}	1.8×10^{-3}	4.24×10^{-4}
INXA2	1.22×10^{-1}	8.336×10^{-1}	5.68×10^{-2}	4.104×10^{-2}
INXA3	4.678×10^{-1}	4.5×10^{-2}	1.6×10^{-3}	3.44×10^{-4}
AFA1	2.209×10^{-2}	2.209×10^{-2}	2.209×10^{-2}	4.21×10^{-3}
AFA2	3.15×10^{-2}	2.209×10^{-2}	2.209×10^{-2}	4.21×10^{-3}
AFA3	2.235×10^{-2}	2.209×10^{-2}	4.128×10^{-2}	2.23×10^{-3}
AFA4	3.15×10^{-2}	6.689×10^{-2}	4.128×10^{-2}	6.689×10^{-2}
AMA1	1.57×10^{-2}	2.97×10^{-2}	1.633×10^{-1}	4.2×10^{-4}
AMA2	4.678×10^{-1}	4.497×10^{-2}	1.6×10^{-3}	3.45×10^{-4}
AXA1	4.371×10^{-1}	7.05×10^{-3}	1.34×10^{-3}	2.27×10^{-4}
AXA2	3.07×10^{-2}	5.56×10^{-2}	1.6×10^{-3}	4.104×10^{-2}

5 Conclusion

Approximate adders are used in the DSP processor and in the error resilient application to reduce the circuit complexity with some acceptable degradation in the output. The approximation is applied to full adders either at gate level or at the transistor level. The probability of erroneous outputs in the approximate full adder at the gate level is less when compared to the probability of erroneous output in the approximate full adder at the transistor level. Further, the analysis is extended to the multi-bit adder, which is designed by using the approximate full adders in different degrees of approximation. The designed ripple carry adder is evaluated based on their Structural and error characteristics. Among all the approximate adders taken for the exploration, AFA1 and AFA2 shows a reasonable balance between the area and accuracy.

References

1. Almurib, H.A., Kumar, T.N., Lombardi, F.: Inexact designs for approximate low power addition by cell replacement. In: 2016 Design, Automation and Test in Europe Conference and Exhibition (DATE), pp. 660–665. IEEE (2016)
2. Dutt, S., Nandi, S., Trivedi, G.: Analysis and design of adders for approximate computing. ACM Trans. Embed. Comput. Syst. (TECS) **17**(2), 40 (2018)
3. Gupta, V., Mohapatra, D., Raghunathan, A., Roy, K.: Low-power digital signal processing using approximate adders. IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst. **32**(1), 124–137 (2013)
4. Liang, J., Han, J., Lombardi, F.: New metrics for the reliability of approximate and probabilistic adders. IEEE Trans. Comput. **62**(9), 1760–1771 (2013)
5. Mittal, S.: A survey of techniques for approximate computing. ACM Comput. Surv. (CSUR) **48**(4), 62 (2016)

6. Oklobdzija, V.: The Computer Engineering Handbook. Computer Engineering Series. CRC Press, Boca Raton (2001)
7. Parhami, B.: Computer Arithmetic: Algorithms and Hardware Designs. Oxford University Press, Inc., New York (2009)
8. Townsend, W.J., Swartzlander, E.E., Abraham, J.A.: A comparison of Dadda and Wallace multiplier delays. In: Advanced Signal Processing Algorithms, Architectures, and Implementations XIII, vol. 5205, pp. 552–561. International Society for Optics and Photonics (2003)
9. Verma, A.K., Brisk, P., Ienne, P.: Variable latency speculative addition: a new paradigm for arithmetic circuit design. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp. 1250–1255. ACM (2008)
10. Yang, Z., Jain, A., Liang, J., Han, J., Lombardi, F.: Approximate XOR/XNOR-based adders for inexact computing. In: 2013 13th IEEE Conference on Nanotechnology (IEEE-NANO), pp. 690–693. IEEE (2013)