Int. J. Nonlinear Anal. Appl.
Volume12, Special Issue, Winter and Spring 2021 , 937-950
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2021.5526

Fast adder with the ability of multiple faults detection and correction

Hamid Tavakkolai[a], Gholamreza Ardeshir[a,*], Yasser Baleghi[a]

[a]*Dept. of Electrical and Computer Engineering, Babol Noshirvani University of Technology, Babol, Iran.*

(Communicated by Mohammad Bagher Ghaemi)

Abstract

Scaling makes digital circuits highly vulnerable to faults, hence, fault detection and correction in digital systems is crucial. This problem becomes more serious when the complexity and frequency of the system on a chip increase in order to achieve higher performance. Addition is one of the fundamental mathematical operators, which is also the basis of many other operations such as subtraction, multiplication, and addressing. Therefore, designing a fault-tolerant adder is a hot topic in VLSI circuits. In such circuits, the important issue of detecting and correcting multiple faults while using few hardware resources is of high interest from the aspect of circuit design optimization. In this paper, a fast adder is used, and the ability to detect and correct multiple errors is supplemented. In the proposed architecture, a self-testing full adder is designed based on multiplexers, and then this full adder is incorporated to design a self-testing multi-bit fast adder. Furthermore, a self-correcting full adder is designed using redundancy, and then this full adder is applied to design a self-correcting multi-bit fast adder. The syntheses and simulations of the proposed adders are performed for 8, 16, 32, and 64 bits. The obtained results show that the proposed adders outperform recent works in terms of the used hardware resources and the multiple faults detection and correction capability.

Keywords: Fast Adder, Fault-tolerant Adder, Self-testing Adder, Self-correcting Adder, Soft Fault, Multiple Faults.

1. Introduction

The probability of occurring faults in electronic circuits and components is always non-zero [10]. Designing circuits with the ability to detect and correct faults is of great importance for improving the

*Corresponding author

Email addresses: h.tavakolaee@iauamol.ac.ir (Hamid Tavakkolai), g.ardeshir@nit.ac.ir (Gholamreza Ardeshir), y.baleghi@nit.ac.ir (Yasser Baleghi)

Received: January 2021 *Accepted:* June 2021

reliability of different systems such as aircraft apparatus, biomedical devices, and satellites working in remote harsh environmental conditions [1]. In addition, fault-detecting and fault-correcting systems are required in Nano-electronic circuits, which are exposed to external interference like cosmic rays [14]. Therefore, the capability of detecting and correcting faults is considered as a priority for future electronic technologies.

In VLSI systems, faults are divided into two categories: soft faults (transient) and hard faults (permanent). Transient faults are caused by transient environmental factors such as cosmic rays and electromagnetic interferences. They may cause sudden changes in memory elements. Permanent faults are caused by changes in the structure of circuit elements [9]. If in a circuit just one transient fault or permanent fault occurs, then the circuit will have a single fault; otherwise, the circuit will have multiple faults.

As adders are key elements in digital systems, their optimization has become the target of many researches [17]. Accordingly, fault detection and correction approaches in order to improve the adder design in terms of speed, power, and area have been considered by many researchers. A circuit with the ability to detect and correct faults should do two tasks: 1- fault detection, 2- fault correction, and an efficient one should do these two tasks with minimum impact on the adder performance [19] and size.

Systems with the ability of fault detection and correction are typically based on the concept of redundancy. However, in general, the following techniques are applied at circuit level for fault detecting and correcting adders in VLSI design [15]

a) Multi-version method: in multi-version methods such as double modular redundancy (DMR), triple modular redundancy (TMR), and 5 modular redundancy (5MR), the structure of the circuit is repeated in several similar versions, and the inputs of the circuit are applied to all its versions. The outputs of these versions are compared with each other using a voting circuit [[3]-[2]], so that the highest voted output is chosen as the final output [13].

b) Self-checking logic structure: a faultless circuit generates a sequence of outputs corresponding to an input sequence. However, in a faulty circuit, the output sequence is invalid. Self-checking logic structures detect faults by comparing the produced output sequence with the input sequence that should be in correspondence with each other. Then, they correct the found faults by some redundant hardware. Currently, most of the researches are done based on this technique, as it uses fewer hardware resources than other methods. For example in [8], the Ripple carry (RC) adder and the Perfix Kogge-stone adder were combined in such a way to settle a trade-off between the computational speed and the used hardware. Then, using TMR redundancy in RC adder made the adder circuit fault-tolerant. In [18], the fault detection was performed using Parity Prediction and the fault correction was done by TMR in the Carry Tree part. In [1], using the input-output relationship made the full adder fault-tolerant, and then this full adder is applied in the Carry Select (CS) full adder.

c) Array structures: in this method, the adder circuit is divided into some similar blocks called array structures. When a fault occurs in a block, the faulty block is replaced by a redundant block. For example in [11], RC and CS adders became fault-tolerant by using iterative logic array (RLA). This approach creates redundant blocks along with array blocks, to replace faulty blocks with redundant ones through multiplexers. In [15], fault-tolerant systems inspire from unicellular creatures such as bacteria and bacterial colonies to develop electrical systems with the ability to correct faults. In this approach, the components of the circuit are addressed into their smallest component called cell. When a fault occurs in a cell, this faulty cell is replaced by a similar redundant cell using a suitable addressing mechanism.

In this paper, a fast adder circuit with the ability to detect and correct multiple faults is presented. At first, a self-testing self-correcting fast full adder is designed with the ability to detect and correct multiple faults using fewer hardware resources than other schemes. Then this full adder is applied in a multi-bit fast adder to build an adder capable of detecting and correcting multiple faults.
The rest of the paper is organized as follow: section 2 introduces the multi-bit fast adder [17] which is the basis of the next fault tolerant designs. The designed self-testing full adder and self-testing multi-bit fast adder are presented in section 3 and 4, respectively. Section 5 explains the multiple faults detecting self-correcting full adder. Section 6 describes the multiple faults detecting self-correcting multi-bit fast adder. Finally, the paper is concluded in section 7.

2. Multi-bit Fast Adder

A recently proposed multi-bit fast adder [17] is selected as the basis for the next fault-tolerant designs. A brief description of this fast adder is put forth in the following. Consider the circuit of a full adder at gate level in figure 1 If this circuit is applied to a 4-bit RC adder, the circuit shown in figure 2 is obtained.

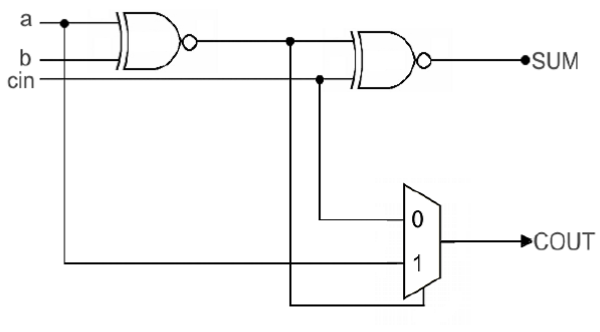

Figure 1: Full adder in gate level [4]

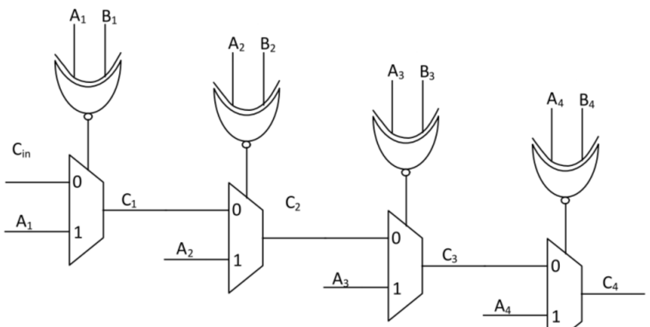

Figure 2: Generating carry out for 4-bit RC adder based on [4]

According to this circuit, the time required for calculating C_{out} (same C_4) carry is equal to the computational time of four multiplexers (neglecting $XNOR$ gates). In order to reduce the delay of the full adder, in [17] the operation of the adder is reconsidered as follow according to figure 2:

$$\begin{aligned} X_1 &= A_1 XNOR B_1 \\ X_2 &= A_2 XNOR B_2 \\ X_3 &= A_3 XNOR B_3 \\ X_4 &= A_4 XNOR B_4 \end{aligned} \tag{2.1}$$

Based on these equations, table 1 shows how Cout (same C_4) in figure 2 is calculated. As can be observed in table 1, in most cases the value of Cout can be calculated according to just inputs. To be more exact, only one case out of sixteen cases of table 1 requires also Cin to be known for calculating Cout. Based on table 1, the circuit shown in figure 3 can calculate Cout [17].

Table 1: The truth table for calculating the fourth carry bit of a 4-bit RC adder [17].

C_{out}	X_1	X_2	X_3	X_4
A_4	0 or 1	0 or 1	0 or 1	1
A_3	0 or 1	0 or 1	1	0
A_2	0 or 1	1	0	0
A_1	1	0	0	0
C_{in}	0	0	0	0

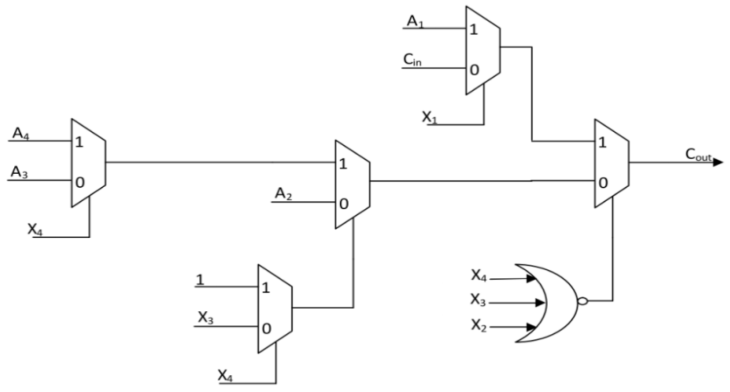

Figure 3: Circuit for calculating the fourth carry bit [17].

In the circuit shown in figure3, the maximum computational time is equal to the computational time of three multiplexers. This time reduces to that of four multiplexers for an 8-bit adder, as shown in figure4.

The upper block in figure 4 is the circuit producing the output carries of the first 4 bits which is the

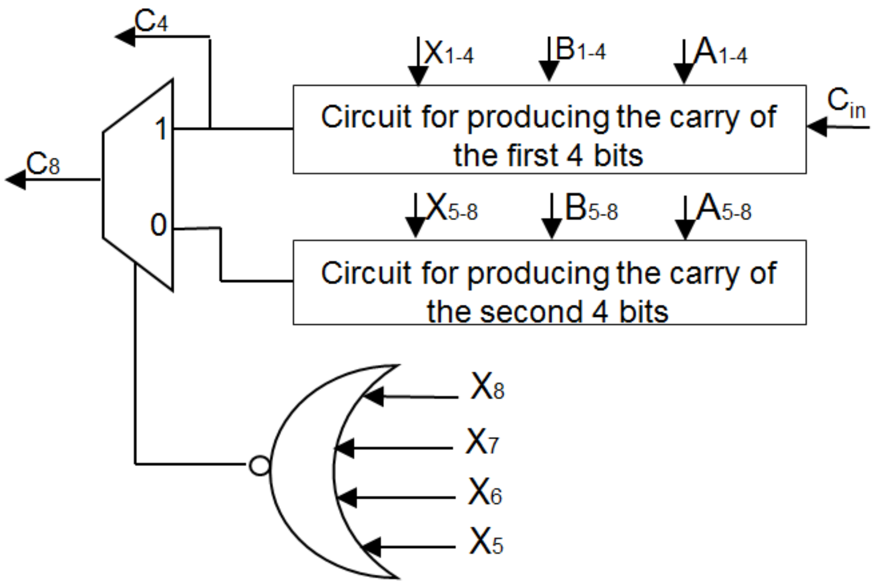
C4
X1-4
B1-4
A1-4
Circuit for producing the carry of the first 4 bits
Cin
1
C8
0
X5-8
B5-8
A5-8
Circuit for producing the carry of the second 4 bits
X8
X7
X6
X5

Figure 4: Circuit for calculating the eighth carry bit [17].

same as the circuit shown in figure 3, and it contains the input carry. While the bottom block in this figure is the circuit generating the output carry of the second 4 bits which is the same as the circuit shown in figure 5, and it has no input carry. Taking these two blocks together, the circuit generating the eighth bit carry is achieved. Similarly, the sixteenth bit carry with a maximum logic length of five multiplexers can be obtained. This procedure can be generalized to be applied to higher-bit adders. Using the circuit generating the fourth, eighth, and higher carry bits, it is possible to design

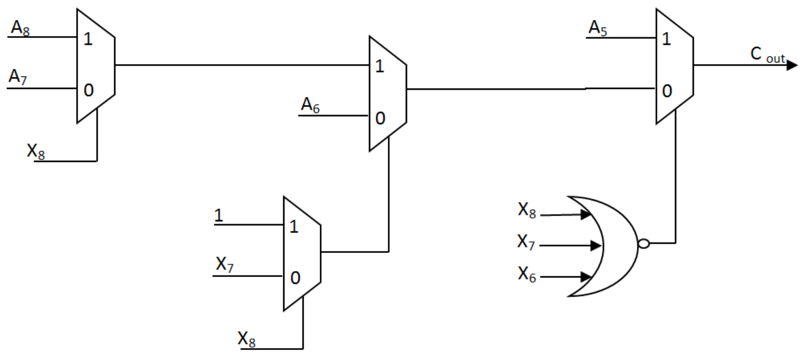

Figure 5: Circuit for calculating output carry of the second 4 bits used in the eighth bit carry producing circuit [17].

higher-bit fast adders. The design paradigm of these adders is shown in figure 6 as a block diagram. Figure 6 shows 16-bit adder [17] in which upper blocks are 4-bit RC adders and bottom blocks are the carry producing circuit as figure 4. Thus, the carry calculation process is speeded up by reducing the logic length.

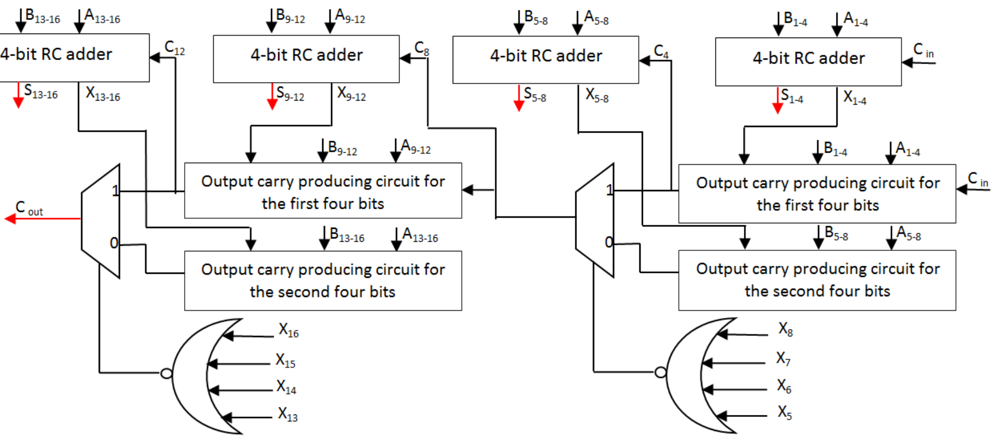

Figure 6: Block diagram of 16-bit fast adder[17].

3. Proposed Self-testing Full Adder

One main challenge regarding the design of fault-detecting and fault-correcting adders is the huge amount of used hardware resources. In the following, some circuits are proposed to tackle with this challenge with more suitable solutions. Table 2 shows the truth table of a full adder. As can be

Table 2: The truth table of a full adder

A	B	C_{in}	S	C_{out}
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

observed from table 2, only in the first and last cases, C_{out} and S outputs and are the same, while in other cases they are complements. According to this, a circuit is proposed which does not change C_{out} in the first and last cases, but in other cases produces the complement of C_{out}. The proposed circuit should have these two characteristics:

(1) The circuit inputs should be the same as the adder inputs.

(2) The calculation paths for S and C_{out} should be independent. This way, if a fault occurs in calculating S, it does not affect C_{out} and vice versa.

First, we assume that calculation paths for S and C_{out} are independent. The following equations are obtained according to table 2.

$$if \quad C_{out} = S \Rightarrow \text{Error} = S \quad \text{XNOR} \quad C_{out} \tag{3.1}$$
$$if \quad C_{out} \neq S \Rightarrow \text{Error} = S \quad \text{XNOR} \quad \bar{C}_{out} \tag{3.2}$$

The conditions held for these equations can be considered as follow:

$$if \quad A \quad \text{XNOR} \quad B = 0 \Rightarrow \text{Error} = S \quad \text{XNOR} \quad \bar{C}_{out} \tag{3.3}$$
$$if \quad A \quad \text{XNOR} \quad B = 0 \quad \& \quad B \quad \text{XNOR} \quad C_{in} = 0 \Rightarrow \text{Error} = S \quad \text{XNOR} \quad \bar{C}_{out} \tag{3.4}$$
$$if \quad A \quad \text{XNOR} \quad B = 0 \quad \& \quad B \quad \text{XNOR} \quad C_{in} = 1 \Rightarrow \text{Error} = S \quad \text{XNOR} \quad C_{out} \tag{3.5}$$

Until the Error signal is equal to 1, the circuit is faultless. Once, the Error signal becomes zero, it indicates a fault in the circuit. Using these equations, the self-testing full adder circuit shown in figure 7 is obtained.

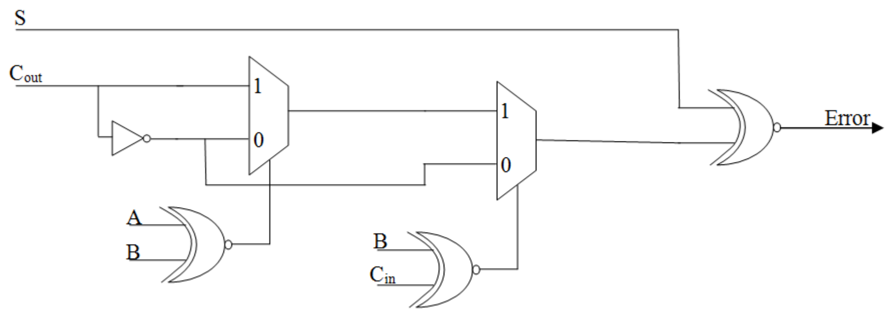

Figure 7: Proposed circuit for fault detection in full adder circuit

If the circuit shown in figure 7 is applied to the full adder of figure 1 [4], the circuit shown in figure8 is obtained.

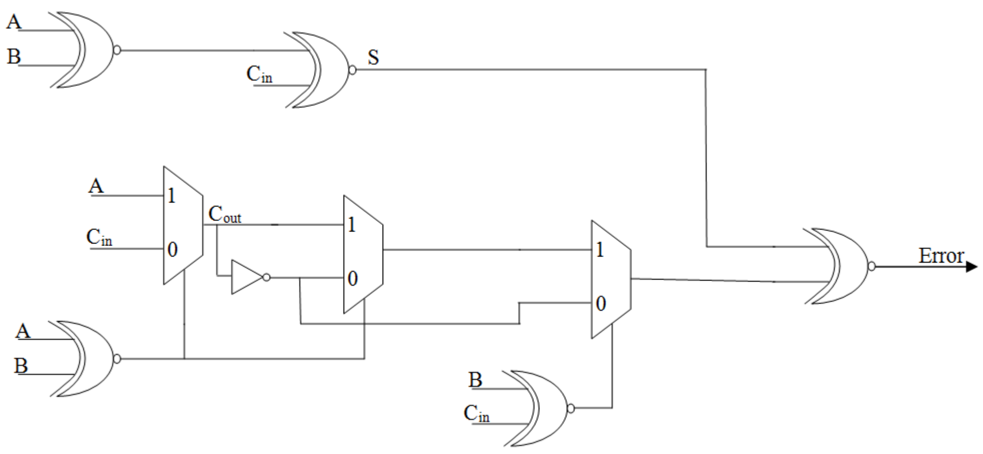

Figure 8: Proposed circuit for fault detection in the full adder of [4]

As seen in figure 8, the A XNOR B gate for calculating C_{out} is the same as this gate in detection part. This way, not only the number of used gates decreases, but also the calculation paths for C_{out} and S are independent. If the circuit of a self-testing full adder is considered as a cell, then figure 9 shows this cell.

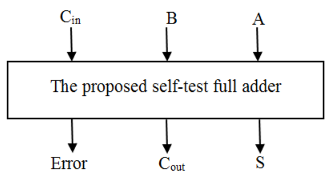

Figure 9: Proposed self-testing full adder cell

Table 3 compares the proposed self-testing full adder with other adders in terms of the number of used transistors. This table contains both the standard full adder with 28 transistors and the full adder of [16] with 10 transistors. The multiplexers used in these schemes are made up of 2 and 6 transistors.

Table 3: Comparison of the proposed full adder with other adders in terms of the number of used transistors

	Number of fault tolerant transistors			The percent of overhead redundancy		
	FA 10T	FA 28T Mux 2T	FA 28T Mux 6T	FA 10T	FA 28T Mux 2T	FA 28T Mux 6T
Proposed scheme	28	46	56	%180	%64	%100
DMR	34	70	70	%240	%150	%150
[14]	34	58	58	%240	%107	%107
[18]	40	-	-	%300	-	-
[19]	-	48	58	-	%71	%107

As can be seen in table 3, the proposed scheme uses fewer hardware resources than other schemes, which indicates the efficiency of the proposed scheme.

4. Proposed Self-testing Multi-bit Fast Adder

The proposed self-testing full adder block is used in the multi-bit fast adder to make it a self-testing multi-bit fast adder. This means that the 4-bit block contains four full adders and the circuit for generating the fourth carry bit. The self-testing circuit is as figure 10.

As can be observed in figure 10, for creating a self-testing carry generation circuit, the carry of the fourth full adder (FA) block is used, and then the XNOR gate is applied for fault detection.

This approach has a little effect on the speed of the proposed fast adder. The fast adder performs its routine operations, and, after adding, it just detects the faulty block. This issue is completely different in fault correction. Since, in fault correction, once a fault is detected it should be corrected

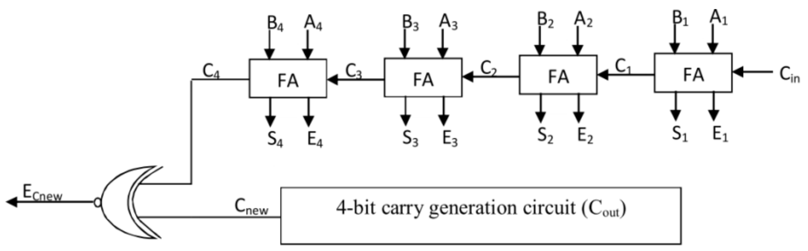

Figure 10: Self-testing block for 4-bit fast adder

immediately, which in turn slows down the addition. To tackle with this problem, we can use TMR approach in the carry generation unit of the 4-bit block not to reduce the speed of the fast adder. However, this leads to the increased hardware resources. We investigate this issue in more details in fault correction section.

5. Proposed Multiple Faults Self-correcting Full Adder

After fault detection, the fault should be corrected so that the circuit returns to its normal mode. Fault correction is always accompanied with redundancy, i.e. if a block is faulty, it should be replaced by its corresponding redundant block. We use this technique in the proposed circuit, i.e. a control circuit replaces the faulty block with its corresponding redundant block.
Consider the proposed self-testing full adder in section 4. In this section, a redundancy-based structure is proposed to make the full adder, self-correcting. The proposed self-correcting full adder is shown in figure 11

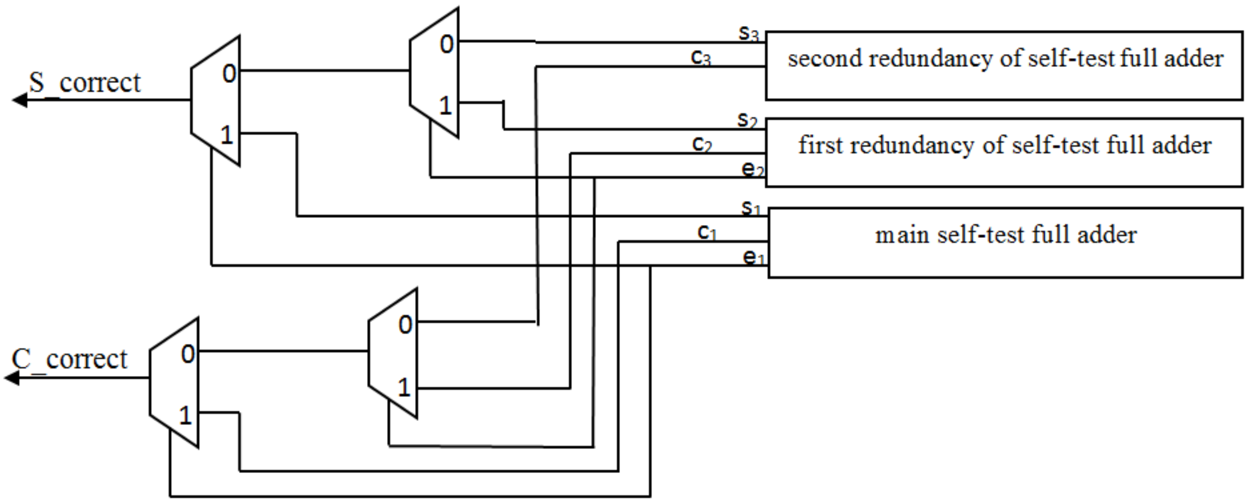
S_correct
C_correct
0
1
s3
c3
second redundancy of self-test full adder
s2
c2
e2
first redundancy of self-test full adder
s1
c1
e1
main self-test full adder

Figure 11: Proposed multiple faults self-correcting full adder

As can be seen in figure 10, when the circuit is faultless, (e_1 and e_2 are equal to 1), then the main self-testing full adder outputs its results. If the main block is faulty, then signal e_1 becomes zero and the main block is replaced by the first redundant block. If the main block and the first redundant block both become faulty, then e_2 becomes zero and the second redundant block enters the circuit. Therefore, the circuit of figure 11 has the ability to correct dual faults. One of the advantages of the

proposed circuit in figure 10 over other dual faults self-correcting circuits is its very compact voting circuit using few hardware resources.
Table 4 shows the comparison of the proposed self-correcting full adder with other adders. This table contains both the standard full adder with 28 transistors and the full adder of [16] with 10 transistors. The multiplexers used in these schemes are made up of 2 and 6 transistors.
As can be seen in table 4, the proposed adder circuit outperforms other adders.

Table 4: Comparison of the proposed full adder with other adders in terms of the number of used transistors

	Number of fault tolerant transistors			Percent of dual faults correction area		
	FA 10T	FA 28T Mux 2T	FA 28T Mux 6T	FA 10T	FA 28T Mux 2T	FA 28T Mux 6T
Proposed scheme	74	128	164	%89	%94	%85
5MR	210	300	300	%23	%46	%46
[20]	104	131	170	%14	%32	%24

6. Proposed Multiple Faults Self-correcting Multi-bit Fast Adder

The proposed self-correcting full adder is used in the multi-bit fast adder to make it capable of correcting multiple faults. The TMR approach is applied to make the fourth bit carry generation circuit self-correcting. Figure 12 shows the self-correcting circuit for generating the fourth carry bit of the fast adder. The voting circuit is obtained from [2].

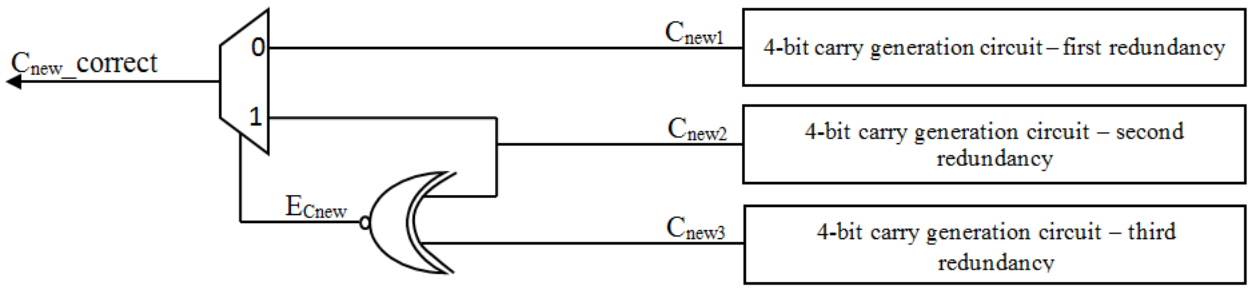
Cnew_correct
0
1
ECnew
Cnew1
Cnew2
Cnew3
4-bit carry generation circuit – first redundancy
4-bit carry generation circuit – second redundancy
4-bit carry generation circuit – third redundancy

Figure 12: TMR approach for self-correcting fourth carry bit generation circuit

Using circuits shown in figure 11 and figure 12, the multi-bit fast adders with the ability to correct multiple faults are designed as depicted clearly in figure 13.

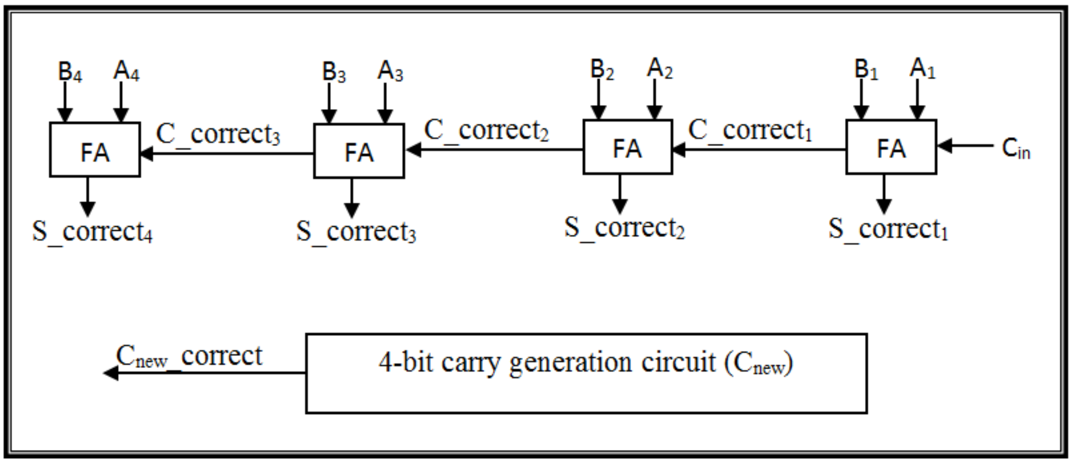

Figure 13: 4-bit fast adder block with the ability to correct multiple faults

7. Simulations and Syntheses

All the proposed circuits are simulated in ModelSim V6.3. When the circuit is faulty, multiple faults are corrected properly. The syntheses of the proposed circuits for 8, 16, 32, and 64 bits are done in linux and by Synopsys Design compiler software C-2009.06-SP5 version.
Table 5 shows the syntheses results of the multi-bit fast adder without fault tolerance, self-testing multi-bit fast adder, and self-correcting multi-bit fast adder respectively for 8, 16, 32, and 64 bits.
Table 6 shows number of transistor for proposed self-testing multi-bit fast adder with DMR design.

Table 5: Comparison of non-fault-tolerant multi-bit fast adder, self-testing multi-bit fast adder, and self-correcting multi-bit fast adder in terms of delay, area, and power consumption

Adder type	8-bit adder			16-bit adder			32-bit adder			64-bit adder		
	power (mW)	area	Critical Path Length	power (mW)	area	Critical Path Length	power (mW)	area	Critical Path Length	power (mW)	area	Critical Path Length
Non-fault tolerant fast adder	4.54	15091	0.94	6.29	35043	2.24	9.97	69900	4.56	13.63	142813	9.32
Self-testing fast adder	10.90	41739	2.78	11.86	87578	5.70	16.70	175157	11.64	22.66	338049	22.76
Self -correcting fast adder	23.01	131591	8.03	27.12	254507	16.19	33.13	507478	33.23	58.41	986008	65.02

Table 7 also shows the number of transistors used in the proposed self- correcting adder with the TMR and 5MR design. table 8 and 9 also show percentage of area correction self-testing and self-correcting adder based on the number of transistors. The percentage of area correction is calculated according to the number of transistors, in that the number of transistors in part of the circuit that is capable of correcting the fault is divided into all transistors of the circuit.

Table 6: Comparison of proposed self-testing adder and DMR design in term of number of transistor

	8bit	16bit	32bit
Proposed non-fault-tolerant fast adder	96	240	480
Proposed Self-testing fast adder	244	544	1072
DMR	308	700	1400

Table 7: Comparison of proposed self- correcting adder, TMR and 5MR designs in term of number of transistor

	8bit	16bit	32bit
Proposed Self--correcting fast adder	646	1408	2816
TMR	778	1712	3424
5MR	1888	4096	8192

Table 8: Comparison of proposed self-testing adder and DMR design in term of percentage of area correction

	8bit	16bit	32bit
Proposed Self-testing fast adder	98 %	93 %	93 %
DMR	62 %	62 %	62 %

Table 9: Comparison of proposed self- correcting adder, TMR and 5MR designs in term of percentage of area correction

	8bit	16bit	32bit
Proposed Self-correcting fast adder	89 %	87 %	87 %
TMR	38 %	38 %	38 %
5MR	25 %	27 %	27 %

The probability of multiple fault correction for proposed self-correcting adder, TMR and 5MR designs evaluated And shown in tables 10 and 11.

Table 10: probability of multiple fault correction for proposed self-correcting adder

probability of multiple fault correction	8bit	16bit	32bit
1 fault	89 %	87 %	87 %
2 fault	79 %	66 %	65 %
3 fault	58 %	50 %	49 %
4 fault	46 %	38 %	37 %
5 fault	37 %	29 %	28 %

Table 11: probability of multiple fault correction for proposed self-correcting adder

probability of multiple fault correction	8bit	16bit	32bit
1 fault in TMR	38 %	38 %	38 %
2 fault in TMR	5 %	5 %	5 %
3 fault in TMR	0	0	0
1 fault in 5MR	25 %	27 %	27 %
2 fault in 5MR	6.25 %	7.29 %	7.29 %
3 fault in 5MR	0	0	0

8. Conclusion

In this paper, a self-testing self-correcting fast adder was presented. Due to the importance of the used hardware and the ability to detect and correct multiple faults in fault-tolerant schemes, these two criteria were considered to compare the proposed adders with other similar adders. The obtained simulation results indicate the superior performance of the proposed schemes than others.

References

[1] M. A. Akbar and J.-A. Lee, *Self-repairing adder using fault localization*, Microelectronics Reliability. 54 (2014) 1443-1451.

[2] T. Ban and L. Naviner, *Optimized robust digital voter in tmr designs*, in Colloque National GdR SoC-SiP, 2011.

[3] S. Habinc, *Functional triple modular redundancy (FTMR)*, Gaisler Research, 2002.

[4] Y. Jiang, A. Al-Sheraidah, Y. Wang, E. Sha and J.-G. Chung, *A novel multiplexer-based low-power full adder*, IEEE Transactions on Circuits and Systems II: Express Briefs. 51 (2004) 345-348.

[5] C. Khedhiri, M. Karmani, B. Hamdi and K. L. Man, *Concurrent error detection adder based on two paths output computation*, in Parallel and Distributed Processing with Applications Workshops (ISPAW), 2011 Ninth IEEE International Symposium on, 2 (2011), 27-32.

[6] C. Khedhiri, M. Karmani, B. Hamdi, K. L. Man, Y. Yang and L. Cheng, *A self-checking CMOS full adder in double pass transistor logic*, in International Conference on Engineers and Computer Scientst IMECS, 2012.

[7] R. V. Kshirsagar and R. M. Patrikar, *Design of a novel fault-tolerant voter circuit for TMR implementation to improve reliability in digital circuits*, Microelectronics Reliability. 49 (2009) 1573-1577.

[8] C. D. Martinez, L. D. Bollepalli and D. H. Hoe, *A fault tolerant parallel-prefix adder for VLSI and FPGA design*, in Proceedings of the 2012 44th Southeastern Symposium on System Theory (SSST), 2012, pp. 121-125.

[9] H. Moradian and J. A. Lee, *Self-repairing radix-2 signed-digit adder with multiple error detection, correction, and fault localization*, Microelectronics Reliability. 63 (2016) 256-66.

[10] L.B. Moraes and A. L. Zimpeck, *Evaluation of variability using Schmitt trigger on full adders layout.* Microelectronics Reliability. 1;88 (2018) 116-21.

[11] A. Mukherjee and A. S. Dhar, *Double-fault tolerant architecture design for digital adder*, in Students' Technology Symposium (TechSym), 2014 IEEE, 2014, pp. 154-158.

[12] A. Mukherjee and A. S. Dhar, *Design of a Self-Reconfigurable Adder for Fault-Tolerant VLSI Architecture*, in Electronic System Design (ISED), 2012 International Symposium on, 2012, pp. 92-96.

[13] A. Namazi, Y. Sedaghat, S. G. Miremadi and A. Ejlali, *A low-cost fault-tolerant technique for Carry Look-Ahead adder*, in 2009 15th IEEE International On-Line Testing Symposium, 2009, pp. 217-222.

[14] R. Rajaei, *Highly reliable and low-power magnetic full-adder designs for nanoscale technologies*, Microelectronics Reliability. 73(2017) 129-35.

[15] M. Samie, G. Dragffy, A. M. Tyrrell, T. Pipe and P. Bremner, *Novel bio-inspired approach for fault-tolerant VLSI systems*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 21(2013) 1878-1891.

[16] R. Shalem, E. John and L. John, *A novel low power energy recovery full adder cell*, in VLSI, 1999. Proceedings. Ninth Great Lakes Symposium on, 1999, pp. 380-383.

[17] H. Tavakolaee, Gh. Ardeshir and Y. Baleghi, *Fast Mux-based Adder with Low Delay and Optimized PDP*, Journal of AI and Data Mining. 7; 3 (2019) 385-392.

[18] M. Valinataj, *A novel self-checking carry lookahead adder with multiple error detection/correction*, Microprocessors and Microsystems, 38 (2014) 1072-1081.

[19] M. Valinataj , *Fault-tolerant carry look-ahead adder architectures robust to multiple simultaneous errors*, Microelectronics Reliability. 55 (2015) 2845-57.