Real-Time System Fault-Tolerant Scheme Based on Improved Chaotic Genetic Algorithm

Jie Wang[1,2(✉)], Junjie Kang[1,2], and Gang Hou[1,2]

[1] School of Software Technology,
Dalian University of Technology, Dalian 116620, China
wangjiel003@163.com
[2] Ubiquitous Network and Service Software Laboratory in Liaoning,
Dalian 116620, China

Abstract. Traditional evolutionary fault-tolerant scheme can effectively repair circuit faults, but for large-scale integrated circuits, the evolution process consumes a lot of time and it is difficult to meet the real-time requirements. In this paper, a real-time system fault-tolerant scheme based on improved chaotic genetic algorithm is proposed. The scheme uses a built-in test detection mechanism with feedback to detect the running state of the circuit in real time. When a fault occurs, normal system operation is maintained by the fault compensation mechanism. At the same time, the system uses the evolution repair mechanism to repair the faulty circuit. Evolution process uses an improved chaotic genetic algorithm, which can quickly converge to obtain a repair circuit through adaptive chaotic crossover and mutation. This paper builds a fault-tolerant system on the FPGA. In the experiment, the fault is randomly injected into circuit so that to simulate the actual circuit fault. The proposed algorithm and fault-tolerant scheme are used to verify the self-repairing ability of the system. The experimental results show that under real-time constraints, the repair rate of the fault circuit reaches 94%.

Keywords: Fault-tolerant · Evolution hardware · Chaotic genetic algorithm

1 Introduction

With the continuous development of electronic information technology, integrated circuits are widely used in aerospace and communications. Due to the increasing size and complexity of integrated circuits, microelectronic systems may malfunction in complex and variable environments. For example, in areas such as aerospace and deep-sea exploration where reliability is critical, if faults occur and the electronic system do not have self-healing capabilities, the entire electronic system will face a serious situation [1]. Therefore, it is very important to design a circuit system with fault tolerance.

According to different implementation methods, the fault tolerance of electronic systems is divided into active fault-tolerant control and passive fault-tolerant control [2]. Passive fault-tolerant control utilizes redundant hardware resources to implement fault repair, such as three-mode redundancy. However, for large scale integrated

M. Jia et al. (Eds.): WiSATS 2019, LNICST 281, pp. 145–156, 2019.
https://doi.org/10.1007/978-3-030-19156-6_14

circuits, hardware redundancy is difficult to implement in many practical circuit systems and fault tolerance is limited. Active fault-tolerant control has better fault tolerance than passive fault-tolerant control. Active fault-tolerant control completes fault repair by fault adjustment or signal reconstruction online adjustment to ensure system stability and reliability after faults occur [3]. Evolution hardware technology is an effective active fault-tolerant control. The adaptive and self-repairing features of the evolution hardware can make the fault circuit automatically adjust the circuit structure to achieve self-repair [4]. However, because the evolution process consumes a lot of time and the evolution efficiency is not high, it cannot complete the evolution operation of the entire electronic system. Considering the real-time requirements of system fault tolerance, after fault occurs, if the fault circuit cannot complete the self-repair in real time, normal operation of the system will be affected a lot.

Aiming at the above problems, this paper proposes a real-time system fault-tolerant scheme based on improved chaotic genetic algorithm. The fault-tolerant scheme uses the Built-in Test technology to monitor the circuit system in real time. When the system detects the fault, the faulty circuit is differentiated through the Built-in Test to quickly find the fault source. After the fault source isolation is completed, the system uses the fault compensation mechanism to maintain the normal operation of the circuit, and adopts an efficient chaotic genetic algorithm to quickly obtain the repair circuit, so as to realize the online real-time repair.

2 Related Work

Programmable logic devices such as FPGAs are widely used in integrated circuit design due to the features of high integration and flexible design. However, due to the complex environmental changes, programmable logic devices used in aerospace are susceptible to radiation, then faults may occur in the circuit system. Therefore, fault-tolerant research on programmable logic devices has been widely concerned by researchers.

The most common fault-tolerant method used in FPGAs is the Triple Modular Redundancy technology (TMR) [5]. TMR does not require an additional fault detection circuit, and the correct output of the system is determined only by voting, thereby quickly completing the fault repair. Sari et al. combined TMR with scrubbing technology to improve system reliability by reducing the execution time of configuration memory scrubbing [6]. But it was difficult to utilize redundant resources for the entire electronic system. And for the problem of on-chip embedded memory failure, Patil et al. [7] proposed a scheme for detecting and repairing faults in multi-port memories by using built-in self test, thereby reducing resource area consumption. Similar to redundancy technology, another popular fault-tolerant technology is reconstruction technology. The reconstruction technology pre-stores the configuration circuit of the original system. After detecting the fault, it reconstructs the fault circuit with the saved

configuration information, so that the circuit resumes normal operation [8, 9]. However, the reconstruction technology needs to reconfigure the entire electronic system. Every time a small fault occurs, the entire system will be reconfigured, which will generate a lot of time overhead. In order to meet real-time constraints, Nazar [10] proposed a partial reconfiguration scheme to repair circuit faults and reduce repair time consumption. But this scheme relies on fine-grained fault detection, which is difficult to meet complex circuit systems.

Evolution fault-tolerant technology does not require a large amount of additional redundant resources, nor does it require pre-stored system configuration information. It can adjust the circuit structure to adaptively repair faulty circuits, significantly reducing hardware resource consumption and repair time consumption. Yang et al. [11] proposed a three-module redundancy architecture based on evolution mechanism. This scheme evolves the system circuit into a redundant module with different structures through an interactive two-stage mutation evolution strategy, which can effectively solve the common mode fault problem, but the hardware resources are more expensive. Zhang et al. [12, 13] proposed a new self-repair technology based on evolution hardware and compensation balance technology, which can realize fault self-repair of various circuits and devices through dynamic configuration. However, if the same problem occurs multiple times in different time periods, the system will repeat the evolution repair, which takes time. In response to this problem, Liu et al. [14, 15] proposed a real-time system fault-tolerant strategy based on evolutionary hardware, which accesses the configuration library when multiple faults occur in the same circuit, and reduces the repair time by combining the configuration library and the similarity evolution algorithm. However, the problem is that the repair constraint duration of the system circuit cannot be accurately predicted, and maintaining the configuration library also adds extra time and resource consumption.

The above fault-tolerant schemes have good fault tolerance after experimental verification. But since the specific fault repair constraint period cannot be obtained, the real-time performance is relative. If the repair circuit cannot be obtained within the constraint period, even if the repair circuit is finally generated, fault tolerance fails too, so a fault-tolerant scheme with efficient real-time and fault tolerance must be designed.

3 Real-Time Fault Tolerant System

Since different circuit systems have different complex structures, each circuit unit has different repair constraint time after a circuit failure occurs. In the case that the constraint time cannot be obtained, it must be ensured that the normal operation of the circuit cannot be stopped when the system is repaired, and the repair should be as fast

as possible. This paper presents an online real-time repair scheme. This scheme detects circuit faults through built-in test. After the fault occurs, the detection mechanism finely locates and isolates the fault source. Then it uses the fault compensation mechanism to maintain the normal operation of the system. At the same time, it runs the evolution repair mechanism to quickly generate the repair circuit, and finally realizes the online real-time repair of the system and improves the reliability of the system.

3.1 Fault Detection Mechanism

The premise of fault tolerance is that after the circuit fails, the system can quickly detect the fault area and take corresponding repair measures. This paper proposes a detection mechanism of Built-in test (BIT) with feedback. By running the BIT, the system periodically detects the working conditions of the various components during the operation of the circuit. As shown in Fig. 1, the input parameter is used to compare and analyze with the truth table. The system records the circuit results of each cycle for the feedback mechanism. If there is no fault, the system runs normally. If a fault occurs, the system generates a feedback signal, and the circuit re-runs and get the result, and compares the two running results with the truth table to determine whether it is a transient fault. If the fault is a transient fault, the system is operating normally. If it is not a transient fault, the system generates a fault signal and isolates the fault source.

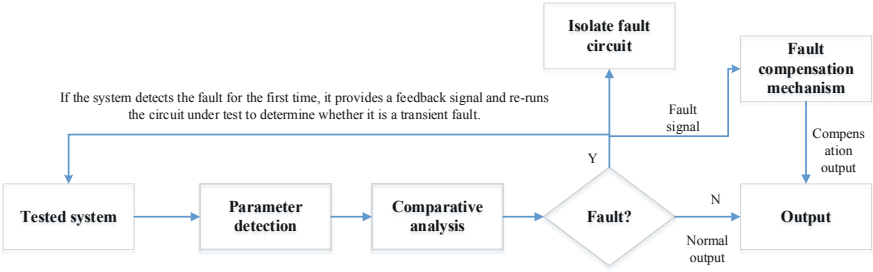

Fig. 1. BIT fault detection mechanism with feedback

BIT technology has circuit status detection and fault detection to improve the accuracy of fault diagnosis and locate fault sources. However, there are defects in the detection of transient faults. The BIT detection mechanism with feedback proposed in this paper can effectively deal with transient faults, and reduce the diagnosis and repair time.

3.2 Fault Compensation Mechanism

After fault detection locates the fault source circuit, it is necessary to isolate the fault circuit. At this time, the system still needs normal operation. Due to the complicated structure of the circuit system, it is impossible to predict the repair constraint period of the faulty circuit. If the fault repair cannot be completed within the constraint period, the operation of the system is directly affected, and even leads to serious consequences.

Based on the above reasons, this paper proposes a fault compensation mechanism. After receiving the fault signal, the system uses the fault compensation mechanism to maintain the normal operation of the circuit, and then uses the BIT to isolate the fault circuit to provide security for fault repair. Figure 2 shows the fault compensation mechanism. The system uses the truth table for fault detection and fault compensation. When the fault occurs, the fault detection mechanism sends a signal C1 to the multi-channel analog switch MUX. Then BIT isolates the faulty circuit. Signal C0 triggers the fault compensation mechanism. Truth table provides the correct output data, and the compensation mechanism outputs the correct data through the MUX to ensure the normal operation of the system.

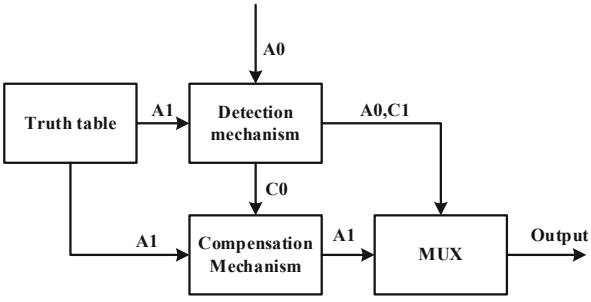

Fig. 2. Fault compensation mechanism

While the repair mechanism is running, the system can operate normally. It is not necessary to predict the repair constraint duration of the circuit, and the faulty circuit can be reconfigured after the rapid evolution, so that the online real-time repair of the fault can be realized.

3.3 Evolution Repair Mechanism

System fault tolerance is a real-time process. If the evolution mechanism cannot obtain the repair circuit under the real-time constraints, the system function will be directly affected. For the evolution hardware fault tolerance technology, it is necessary to obtain a repair circuit within the fault repair time of the system. In the case of unknown problems, running evolution algorithms occupies most of the repair time, so a good evolution algorithm plays a key in improving the real-time performance of the evolution hardware fault-tolerant technology. Genetic algorithms are the most commonly

used algorithms for evolution hardware. It uses the fitness value to guide the search process without requiring the target function to be continuous. And its search process starts from the solution group, not from the point. However, the population of the genetic algorithm is randomized at the time of initialization, and the quality of the individual cannot be guaranteed. A large part of the population is far away from the optimal solution. And because the crossover and mutation operations of genetic algorithms are random, the performance of individuals after hybridization and mutation cannot be guaranteed to be better than the original. Finally, the convergence speed and accuracy of the algorithm may be affected. Based on the above reasons, this paper proposes an improved chaotic genetic algorithm (ICGA). ICGA improves the evolution rate and local search ability of the algorithm by improving the process of selection, intersection and mutation of genetic algorithm, thereby improving the efficiency of evolution repair.

Selection

The initial group determines the search efficiency of the optimal solution to some extent. The genetic algorithm usually randomly generates the initial population within a given range, and the randomly generated population has discrete, random and irregular characteristic. In order to improve the ergodicity of the initial population and increase the diversity of the population, The elite retention mechanism is adopted for the 10% group with the highest fitness in the group, and the other 90% use roulette to make the selection. The roulette method will select individuals with high fitness from 90% of the population and we repeat the process until the population is full. The selection method combining elite retention strategies with roulette can ensure that each generation's optimal solution can enter the next generation of populations, accelerating the convergence of genetic operations.

Intersection

The traditional genetic algorithm adopts single-point crossover or uniform crossover, which leads to slow convergence in the early stage of evolution, and it is easy to destroy the excellent mode of chromosomes and reduce the evolution efficiency. To improve algorithm efficiency, ICGA uses adaptive chaotic crossover. The crossover operator is continuously adjusted according to the change in chromosome fitness of the individual. As shown in Eq. 1, the number of individual intersections individuals is obtained by calculating individual fitness values and population fitness values. As shown in Eq. 2, then we use the Tent chaotic map with strong traversal uniformity to generate intersections that are evenly distributed across chromosomes. The 10% of individuals with large initial chromosome adaptation values need to be judged. If the offspring fitness value is smaller than the parental fitness value, the parental chromosome is still retained.

$$Pc = \begin{cases} k_1, f \le f_{avg} \\ k_2(f_{\max} - f)/(f_{\max} - f_{avg}), f > f_{avg} \end{cases} \tag{1}$$

$$x^{n+1} = \begin{cases} 2x^n, 0 \le x^n \le 0.5 \\ 2(1 - x^n), 0.5 \le x^n \le 1 \end{cases} \tag{2}$$

K_1 and k_2 are constants, f is the current fitness value of the chromosome, f_{max} is the current maximum fitness value of the population, and f_{avg} is the average fitness value of the population, and x is a random number in the interval [0, 1], and the repeated operation generates a uniform random number. Then we map the random numbers to the feasible range of the chromosome length to get the intersection position of the chromosome.

Mutation

In the later stage of evolution, the average fitness of the population is close to the optimal chromosome fitness. At this time, we need to fine-tune the population chromosomes through the mutation operation to solve the local optimal problem. Similar to the crossover process, ICGA uses adaptive chaotic mutation. By comparing the fitness of the population's optimal chromosome with the current chromosome's, the mutation numbers of the current individual is calculated. ICGA continuously searches the surrounding area of the chromosome and gradually reaches the optimal chromosome fitness. The mutation number is calculated by the following formulas, where N is the length of the chromosome, F_{max} is the maximum fitness specified by the user, F_i is the fitness of the ith chromosome, P_{max} is the maximum mutation rate specified by the user, L_{max} is the maximum mutation number, and N_i is the mutation number of the ith chromosome. From formulas (3) and (4), we first calculate the difference between the fitness of the ith chromosome and the maximum fitness, and then obtain the proportion of the difference. Finally, we combine the proportion and the mutation rate to obtain the mutation number of the ith chromosome. It can be seen that the greater the proportion is, the smaller the chromosome fitness is, and the more the mutation number will be.

$$L_{\max} = N * P_{\max} \tag{3}$$

$$N_i = L_{\max} * \frac{F_{\max} - F_i}{F_{\max}} \tag{4}$$

When the variant numbers are obtained, ICGA generates random mutation positions through the Tent chaotic map and maps the random numbers to the feasible region of the chromosome length. In the early stage of evolution, ICGA quickly achieves a higher level of average fitness of the population through crossover and mutation operations. Later, since the individual's chromosome fitness is close to the maximum fitness value, the number of variants is also reduced. In this way, the optimal solution is searched in the solution space, and each variation only retains high-quality individuals, which can make the algorithm break through the limitation of the local optimal solution. The algorithm flow of ICGA is as follows.

ICGA

```
1  begin
2    Initialize P(g);
3    Fitness(g)←CalculateFit(P(g));
4    if (get the solution) break;
5    while g< g_max do
6     begin
7      P′(g)← Select(P(g), Fitness(g));
8      CrossoverNum(g)← CalCossover[P′(g)];
9     CrossoverPosition(g): get crossover positions through Tent chaotic map;
10     P″(g)← Crossover(P′(g), CrossoverPosition(g));
11    Fitness(g)←CalculateFit(P(g));
12     while i <N do
13      begin
14      CurrentFit ← calFit(chrom[i]);
15      MutationRate(i) ← MutRate_max *(Fit_max -CurrentFit)/ Fit_max;;
16      MutationNum(i) ← MutationRate(i)*N;
17      while j < MutationNum(i)
18      begin
19        MutationPosition(i):Get mutation positons through Tent chaotic map;
20      end
21      P‴(g)← Mutation(P″(g), MutationPosition(i));
22        if(CalculateFit(P″(g)) < CalculateFit(P‴(g)))
23          P″(g)← P‴(g);
24         i ← i+1;
25      end
26    Fitness(g)←CalculateFit(P″(g));
27    if (get the solution) break;
28    g← g+1;
29   end
30 end
```

4 Implementation

The experiment uses Xilinx's Zynq UltraScale zcu102 development board. We use 2-bit multiplier and 8-bit parity checker as experimental objects for fault-tolerant evolution. In this experiment, the fitness evaluation of the evolution hardware uses the internal evolution method, that is, each chromosome in the group is downloaded to FPGA for fitness evaluation, thereby accelerating the calculation speed of the fitness and reducing the time consumption caused by the fitness calculation. Considering the reconfiguration time of chromosomes and the computational cost of fitness, this experiment uses Virtual Reconfigurable Circuits (VRC) [16], which implements more efficient chromosome coding operations, simplifies the complexity of the evolution target circuit, and reduces computing costs. The important parameters of the algorithm

and circuit in the experiment are shown in Table 1. It can be seen that virtual reconfigurable circuit of the 8-bit parity checker uses an 8 * 4 matrix with a total of 32 configuration function blocks (CFB). Each CFB contains two inputs and one output. The inputs of column 0 are from the circuit input, and the outputs are used as the inputs of the column 1. After the calculations are completed in the column 3, we can get the final circuit results based on the outputs of the last column. Therefore, after configuring the chromosome as VRC, we can calculate all the circuit inputs and get all the actual outputs. By comparing the actual outputs with the theoretical outputs in the truth table, we can finally get the fitness value of the chromosome.

Table 1. Algorithm and circuit structure parameters

Parameters	8-bit parity checker	2-bit multiplier
VRC scale	8 * 4	8 * 4
Chromosome length, bit	291	300
Population size	30	30
Maximum fitness	256	64
Largest generation	4000	10000

In order to make full use of the software and hardware cooperation characteristics of ZYNQ, in the programmable logic part, we configure the VRC module of the experimental circuit, and build the fault detection module and fault compensation module; then in the processing system, we execute various evolution algorithms to get chromosomes of each generation; finally we transmit the chromosomes to the VRC module in the programmable logic part through the AXI bus to complete the fitness calculation.

We use a fixed fault injection to simulate an actual circuit fault. In the experiment, we keep the inputs of some CFBs in the virtual reconfigurable circuit at 0 or 1 to achieve the effects of circuit failure, resulting in circuit failure. After replacing the fault location repeatedly, the experimental process is as similar as possible to the failure caused by environmental problems, which improves the credibility of the experiment.

When the fault occurs, system determines the fault circuit through the BIT detection mechanism with feedback, and then uses the fault compensation mechanism to maintain the normal operation of the circuit, and simultaneously stimulates the evolution repair mechanism to generate the repair circuit, and finally realizes the online fault repair. We use particle swarm optimization (PSO), simulated annealing (SA), standard genetic algorithm (SGA), adaptive genetic algorithm (AGA) and improved chaotic genetic algorithm (ICGA) as the evolution algorithms to repair the proposed fault-tolerant system. The experimental results are shown in Figs. 3 and 4.

It can be seen from the figure that the fault-tolerant scheme proposed in this paper can repair circuit faults well, but different repair algorithms have different efficiency. With the increase of generations, ICGA can change the crossover operator and mutation operator according to the fitness of the chromosome. In the early stage of evolution, the average fitness of the population is quickly reached a higher level. In the

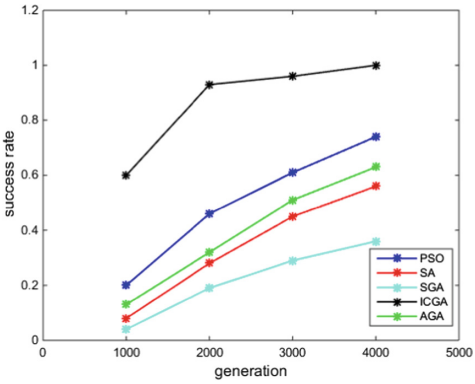

Fig. 3. Results of 8-bit parity

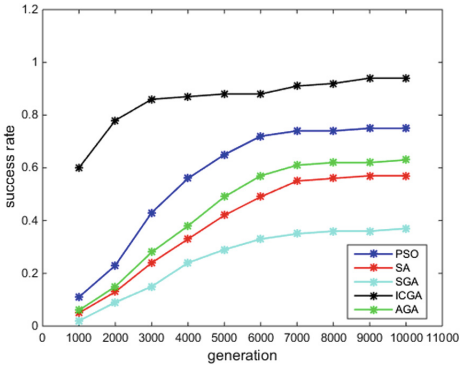

Fig. 4. Results of 2-bit multiplier

later stage, since the individual's chromosome fitness is close to the maximum fitness value, the number of mutations is also reduced accordingly. In this way, the optimal solution is gradually searched in the solution space, and each variation only retains high-quality individuals, which can make the algorithm break through the limitation of the local optimal solution. Figure 3 shows the repair results of the 8-bit parity checker. We performed 30 fault injections on the 8-bit parity circuit. The best repair result is ICGA, and the repair success rate can reach 93% within 2000 generations. Figure 4 shows the experimental results of a 2-bit multiplier. For a more complex 2-bit multiplier, ICGA can complete 60% fault repair within 1000 generations and 90% repair rate within 4000 generations. Compared with other algorithms, ICGA improves the real-time performance of fault-tolerant schemes most obviously.

Figure 5 shows the repair circuit and chromosome coding structure of 8-bit parity checker. Due to the randomness of the evolution hardware repair technology, the optimal chromosome may not be obtained in the end, but the adaptive value of the

iterative local optimal chromosome is very different from the optimal chromosome, which can satisfy most of the circuit functions. Therefore, the local optimal chromosome can still be used to configure the repair circuit, and if the fault is detected again, the evolution repair is performed again.

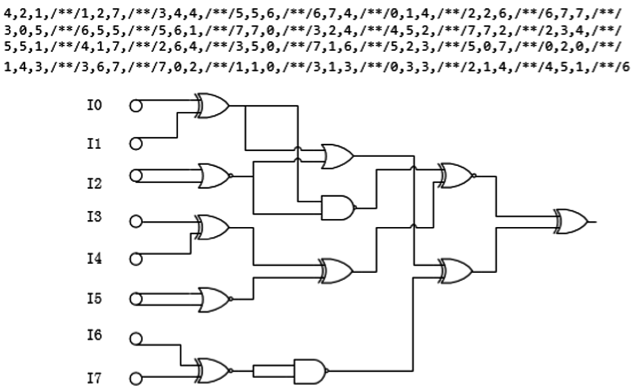

Fig. 5. The repair circuit of 8-bit parity checker

5 Conclusion

Evolution fault-tolerant technology can adjust the circuit structure of the programmable logic device to adaptively repair circuit faults, but the evolution process takes a lot of repair time. Aiming at the real-time problem of evolution repair, this paper proposes a real-time system fault-tolerant scheme based on improved chaotic genetic algorithm. The scheme detects the state of the circuit in real time through the BIT detection mechanism with feedback. When the fault occurs, the system uses the fault compensation mechanism to maintain the normal operation of the circuit, at the same time, stimulate the evolutionary repair mechanism. Through the improved chaotic genetic algorithm, the scheme greatly improves the rate and success rate of evolution repair. Ultimate, the solution realizes online repair of fault circuits, which improves the reliability and stability of the system.

Funding. This work was supported by the National Nature Science Foundation of China (No.61472100) and Fundamental Research Funds for the Central Universities, China (No. DUT17JC26).

References

1. Jamshidpour, E., Poure, P., Saadate, S.: Photovoltaic systems reliability improvement by real-time FPGA-based switch failure diagnosis and fault-tolerant DC-DC converter. IEEE Trans. Industr. Electron. **62**(11), 7247–7255 (2015)
2. Najari, H., Harabi, R.E., Abdelkrim, M.N.: A graphical active fault tolerant control approach. In: 16th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, pp. 285–290. IEEE(2015)
3. Liu, L.J., Yu, W., Yu, Z.: Active fault-tolerant control design for a submarine semi-physical simulation system. Int. J. Control Autom. Syst. **16**, 2363–2372 (2018)
4. Jose, D., Tamilselvan, R.: Fault tolerant and energy efficient signal processing on FPGA using evolutionary techniques. In: Senthilkumar, M., Ramasamy, V., Sheen, S., Veeramani, C., Bonato, A., Batten, L. (eds.) Computational Intelligence, Cyber Security and Computational Models. AISC, vol. 412, pp. 155–164. Springer, Singapore (2016). https://doi.org/10.1007/978-981-10-0251-9_16
5. Nguyen, N.T.H., Cein, E., Diessel, O.: Scheduling voter checks to detect configuration memory errors in FPGA-based TMR systems. In: 2017 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, pp. 1–4. IEEE (2017)
6. Sari, A., Psarakis, M.: Scrubbing-aware placement for reliable FPGA systems. IEEE Trans. Emerg. Top. Comput. (2017)
7. Patil, S.R., Musle, D.B.: Implementation of BIST technology for fault detection and repair of the multiported memory using FPGA. In: Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, pp. 43–47. IEEE (2017)
8. Chatterjee, N., Mukherjee, P., Chattopadhyay, S.: A strategy for fault tolerant reconfigurable Network-on-Chip design. In: 2016 20th International Symposium on VLSI Design and Test. IEEE (2016)
9. Walker, J.A., Trefzer, M.A., Bale, S.J., Tyrrel, A.M.: PAnDA: a reconfigurable architecture that adapts to physical substrate variations. IEEE Trans. Comput. **62**(8), 1584–1596 (2013)
10. Nazar, G.L.: Improving FPGA repair under real-time constraints. Microelectron. Reliab. **55**(7), 1109–1119 (2015)
11. Yang, X., Li, Y., Fang, C., Nie, C., Ni, F.: Research on evolution mechanism in different-structure module redundancy fault-tolerant system. In: Li, K., Li, J., Liu, Y., Castiglione, A. (eds.) ISICA 2015. CCIS, vol. 575, pp. 171–180. Springer, Singapore (2016). https://doi.org/10.1007/978-981-10-0356-1_17
12. Zhang, J.B., Cai, J.Y., Meng, Y.F.: Fault self-repair strategy based on evolvable hardware and reparation balance technology. Chin. J. Aeronaut. **27**(5), 1211–1222 (2014)
13. Zhang, J.B., Cai, J.Y., Meng, Y.F.: Performance analysis of circuit fault self-repair strategy based on EHW and RBT. Beijing Hangkong Hangtian Daxue Xuebao/J. Beijing Univ. Aeronaut. Astronaut. **42**(11), 2423–2435 (2016)
14. Wang, J., Liu, J.W., Feng, B., Hou, G.: The dynamic evaluation strategy for evolvable hardware. In: Proceedings of the 2015 9th International Conference on Frontier of Computer Science and Technology, pp. 91–95. IEEE (2015)
15. Wang, J., Liu, J.W.: Fault-tolerant strategy for real-time system based on evolvable hardware. J. Circ. Syst. Comput. **26**(7) (2017)
16. Srivastava, A.K., Gupta, A., Chaturvedi, S., Rastogi, V.: Design and simulation of virtual reconfigurable circuit for a Fault Tolerant System. In: International Conference on Recent Advances and Innovations in Engineering. IEEE (2014)