A Self-Configuring TMR Scheme utilizing Discrepancy Resolution

Naveed Imran and Ronald F. DeMara
Department of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2362
Email: naveed@knights.ucf.edu, demara@mail.ucf.edu

***Abstract*—We employ output-discrepancy consensus to mitigate faulty modules of a *Triple Modular Redundant (TMR)* arrangement using dynamic partial reconfiguration. Traditionally, the fault-handling resilience of a TMR arrangement is limited to fault(s) in a single TMR instance over the entire mission duration. An additional permanent fault in any of two other TMR instances results in mission's failure. However, in this work, a novel *Self-Configuring approach for Discrepancy Resolution (SCDR)* is developed and assessed. In SCDR, the occurrence of faults in more than one module initiates the repair mechanism, then upon fault recovery, the system is configured into *Concurrent Error Detection (CED)* mode. The approach is validated by the complete recovery of a TMR realization of 25 stage Finite Impulse Response (FIR) filter implemented on a reconfigurable platform as a case study. The results show that a self-healing circuit can be realized exploiting the dynamic partial reconfiguration capability of FPGAs while requiring a streamlined operational datapath compared to TMR.**

***Keywords*-Evolvable hardware and dynamic reconfiguration. Reconfiguration techniques to improve fault tolerance.**

I. Introduction

Field Programmable Gate Arrays (FPGAs) are a suitable candidate for exploring fault tolerant techniques employing reconfigurable platforms [1],[2],[3],[4]. During runtime, the system can be reconfigured to avoid faulty resources. The dynamic reconfiguration capability of FPGAs is useful for building autonomous fault handling systems [2]. Fault tolerance of electronic circuits is an area of growing interest especially for mission-critical applications and generally for nanoscale devices. The reduced feature size of the future nanoscale chips will increase their vulnerability to transient as well as permanent faults. Besides manufacturing faults, the circuits are expected to face runtime faults on nanoscale [5]. Permanent faults especially due to aging effects [6] in sub-90nm technology are important to mitigate for mission sustainability.

Evolutionary techniques have been proposed in literature for achieving fault tolerance in FPGAs. The logic resources or interconnects are somehow represented by individuals of a population to be evolved. The genetic operators are performed to either repair the circuit or to create a robust circuit at design-time. These techniques require the knowledge of input-output values to evaluate the fitness criteria. However, for large circuits having many possible input-output values, the evaluation of every possible combination of input-output becomes an infeasible option. We employ a fitness criterion in which an absolute knowledge of the correctness of output values is not required.

Redundancy-based techniques are popular among fault tolerant community due to their fault detection, masking and isolation capability. An example is a TMR arrangement in which three replicated instances of a system concurrently operate to provide the system output via a majority voter. Even if one module becomes faulty, two healthy modules are able to provide throughput as long as no discrepancy is observed in their output. Yet, an additional fault in one of these two healthy modules can result in a mission's failure, or equivalently a discrepancy in the output of an operational pair (or CED pair) flags one of these two instances as *faulty* [1]. The observable faults are those which manifest their behavior during an evaluation period. The effect of evaluation period on fault detection in CED pair was examined in [7].

Under these conditions, we present a repair scheme for permanent faults in which the discrepancy between the output of a CED pair initiates the genetic recovery process. Upon fault detection in two instances of a TMR arrangement, the system is recovered through reconfiguration by repairing two instances. The followings are the advantages of the proposed repair mechanism:

- A self-healing TMR system is proposed with improved fault capacity. Initially, there are three instances of a module in operation, then in event of faults in two modules, the repair is performed to switch the system into duplex mode. Thus, in effect, the fault capacity of a TMR is improved.
- The fitness evaluation is performed using the actual inputs of the system, avoiding any test vectors. Thus, an optimal solution is sought in the relevant input subspace.
- The fault handling process does not require an explicit fault isolation phase. The faulty modules are consequently avoided by the evolutionary recovery process.
- Fault handling of sequential circuits is demonstrated. A considerable throughput is maintained during the fault recovery process.

978-0-7695-4551-6/11 $26.00 © 2011 IEEE
DOI 10.1109/ReConFig.2011.5

IEEE computer society

II. Related Work

Hardware autonomy is desirable in space systems because manual intervention may be an infeasible option. Steiner et. al [8] proposed an autonomous system in which hardware computational resources are managed at runtime. Their demonstrated system can dynamically parse and synthesize digital circuit netlists, place-and-route on FPGA at a very fine granularity. It relies on a customized implementation tool. Our method operates at a coarse granularity of block-level in the circuit corresponding to pipelined stages of hardware core. The autonomous operation can be realized using dynamic reconfiguration capability, an on-chip micro-processor and the internal access port for reconfiguration. An autonomous operation of hardware is also desirable for other applications involving certain objectives such as power optimization. Mansouri et. al [9] used consensus information with the objective of optimizing power in a decentralized processors network.

Resource testing techniques for fault isolation of FPGA resources have been proposed in literature in the form of either offline testing or online testing [10]. In an offline *Built-In Self Testing (BIST)* method, all the active resources are released from their active functionality and a testing sequence is conducted to verify the correctness of these resources. On the other hand, *Online Testing* schemes may employ the dynamic reconfiguration capability of FPGA and tests can be performed during runtime. In Gericota et. al's approach [11], the active logic resources are concurrently replicated to support a runtime testing procedure. Dutt et. al [12] extended the BIST method to offline as well as online testing modes. The exhaustive evaluation of all the resources through test vectors may be a long process. Our scheme can be conceptualized as resource testing through actual inputs of the circuit.

Redundancy techniques for fault tolerance use redundant modules. The technique is widely used in mission-critical applications for fault detection as well as fault masking. As mentioned previously, *TMR* [13], [14] configuration involves three replicas of the design which are running at a time and the outputs are compared by a voting element. The majority voted output is passed through and becomes the actual output of the system. As the redundancy-based methods do not require exhaustive set of inputs, we have used it in our framework for fault detection. However, unlike the static modules of a conventional TMR setup, the instances in the SCDR scheme are dynamic in nature as the modules are allocated in active throughput datapath at runtime.

Dynamic partial reconfiguration capability of FPGAs has been explored for useful tasks by various researchers [15], [16], [17]. Fault recovery methods of FPGA-based designs usually exploit the reconfigurable nature of the device. After completion of the fault isolation phase, the faulty resources are avoided by reconfiguring the chip so that the design is relocated to a fault-free area. On the other hand, evolutionary techniques [3] such as *Genetic Algorithms (GAs)* have been employed to generate circuits at design-time which are robust to faults [3]. In the current work, the circuit is evolved at runtime to reach the desired level of functionality.

Hardware architectures for adaptive signal processing have been presented in literature. Estrada et. al [18] utilized reconfiguration capability of FPGA to adapt an FIR filter to the changing environment. The SCDR approach described here presents a fault- tolerant architecture of the filter. However, it may be noted, that the filter realization is only a case study. This paper validates the SCDR scheme on a pipelined sequential circuit.

III. The SCDR Approach

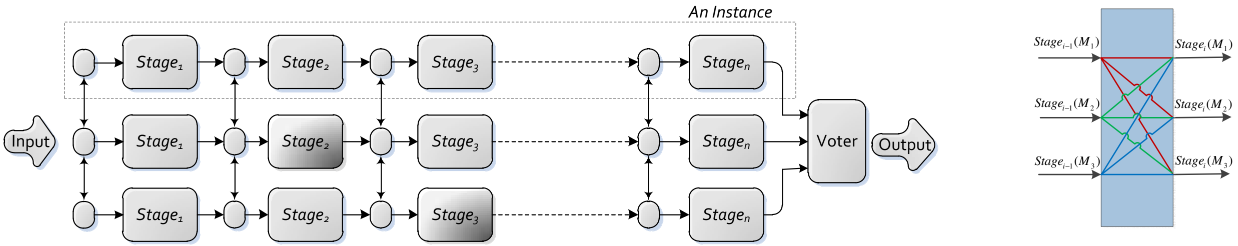
The TMR realization of a sequential circuit is illustrated in Fig. 1(a). A *Circuit Under Test (CUT)* is replicated 3 times resulting into $Instance_1$, $Instance_2$ and $Instance_3$ of the CUT. The majority voter computes the system's main output by enabling the majority output value. Faults are thus masked in the output if a single instance is affected.

To improve the reliability, in the presence of multiple faults which may occur during long mission durations, adaptation of the datapaths can be applied. For this purpose, switches are inserted between various pipeline stages of the circuit to steer the datapath. Combinational circuits can also be partitioned into blocks. The configuration of these switches determine which module is selected as an active element of each instance. The SCDR concept is to avoid the faulty blocks and utilize the healthy blocks in the processing datapath.

In Fig. 1(a), the shaded blocks represent faulty stages. A *Module Switch (MS)* element, which can be realized as a multiplexer or a custom-routing circuit, is connected between every pipeline stage of a digital sequential circuit (or a block more generally) to steer the output of a block to one of three blocks in the following stage. A *Router Box* (RB) is a realization of the three MS elements, one for each instance of a TMR pathway. It supports six possible input-output pairings (i.e. 3!=6) as shown in Fig. 1(b). It can be implemented in hardware by either routing wires through partial reconfiguration or using multiplexers. The former has the advantage that no delay is introduced by the multiplexers in the datapath. However, it comes at the cost of reconfiguration latency during the fault recovery process. On the other hand, fault recovery time can be improved by introducing dedicated multiplexers at design time, but this introduces a drawback that the MS elements remain in the active throughput path at all times.

A. Encoding Representation of the TMR Pathways

In the presented SCDR scheme, an *evolutionary fault recovery* process is used to explore the search space of



(a) The TMR realization of a sequential circuit

(b) A Router Box

Figure 1. Circuit realization to employ the SCDR recovery mechanism

RB settings to obtain a more suitable TMR realization. An individual of the population represents one possible configuration of the TMR system. The number of variables in a GA individual is equal to the number of pipeline stages or number of blocks n in an instance of the TMR arrangement. Therefore, the length of a chromosome is n. Each variable can assume one of 6 possible values while each value corresponds to a unique configuration of the RB. The GA representation of the routing between each stage is exemplified in Fig. 2 where a chromosome's values correspond to different configurations of the RB. The representation of the GA problem is such that it exploits the dynamic partial reconfiguration capability of FPGA. The faulty blocks of a duplex can be swapped with healthy blocks of the third instance of TMR arrangement during runtime.

In the following, the scheme is presented in the context of a standard GA [19] to facilitate conventional analysis.

B. Fitness Function

Given many possible configurations of the system, it is difficult to ascertain which is superior in terms of desired functionality of the system. Many previous approaches store the input-output test vectors or truth-tables of the circuit. The SCDR avoids this approach because of two drawbacks: 1) the storage requirement would be prohibitive for large circuits 2). The application of test vectors may require the system taking offline. Therefore, we employ a previously developed *Competitive Runtime Reconfiguration* (CRR) approach [1] in which individuals are selected based upon their behavior consistency with the consensus of the other population members.

A *Discrepancy Value* (DV) is defined as the Euclidean distance between the outputs of two instances of a TMR in a given evaluation window. The objective is to minimize the DV between $Instance_1$ and $Instance_2$.

$$DV = \frac{||Y_1 - Y_2||}{\sqrt{E}} \qquad (1)$$

where
Y_1 = Output of the *Instance*$_1$
Y_2 = Output of the *Instance*$_2$
E = *Evaluation Window*

which is used to access the fitness of competing TMR pathways.

C. Fitness Evaluation

For the purpose of evaluation, each individual of the population is instantiated on the chip and its fitness is updated after a temporal sliding window of input samples. In the SCDR approach, the individuals are evaluated subjected to the actual inputs of the system rather than synthetic test vectors. The input is applied to the TMR arrangement and the output of the individual instances is observed during the evaluation window period, E. Upon the completion of the evaluation window, a new configuration of the TMR is introduced into operation. Once all the individuals have been evaluated, a generation of the GA is complete. Thus, fitness scores of the individuals which is based upon their agreement/disagreement history, becomes available at the end of a generation. The SCDR proceeds next to select which configurations should be retained for subsequent operations.

D. Fitness Selection

Fitness Selection is the strategy of choosing individuals from the population to be parents which produce offspring that realize new arrangements. The fitness selection strategy in this paper is based on the rank selection [20]. The fitness score of the individuals is converted into the respective rank, then the probability of an individual being selected as a parent is inversely proportional to the square root of its rank. Recall that we want to minimize the fitness score which is the DV. Thus, the use of a rank selection strategy makes it more probable that the configurations having small DV would survive to produce new configurations in the next generation.

E. Genetic Operators

A *crossover* operator interchanges the segments of chromosomes between two parents to produce offspring. A

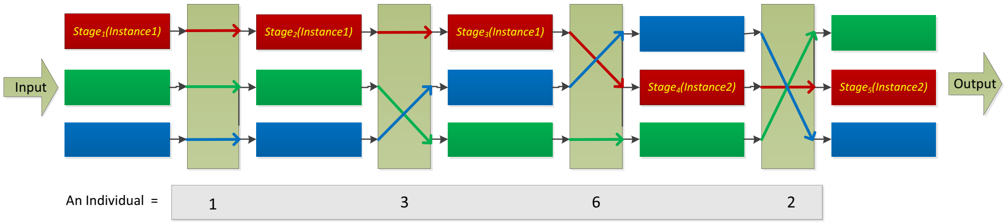


Figure 2. An example of the mapping between an individual and the realized configuration

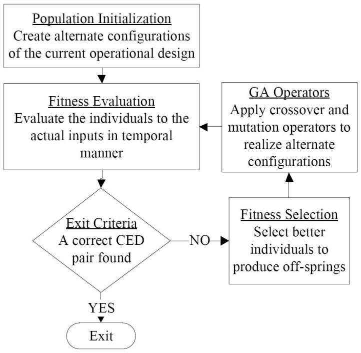

Population Initialization
Create alternate configurations of the current operational design
Fitness Evaluation
Evaluate the individuals to the actual inputs in temporal manner
GA Operators
Apply crossover and mutation operators to realize alternate configurations
Exit Criteria
A correct CED pair found
NO
Fitness Selection
Select better individuals to produce off-springs
YES
Exit

Figure 3. The evolutionary recovery process in the context of a standard GA [19]

	The 25 stages of an FIR filter																								
$Instance_1$					x	x	x																		
$Instance_2$																									
$Instance_3$																x	x	x	x	x					

x indicates a faulty stage

Figure 4. A faulty TMR configuration

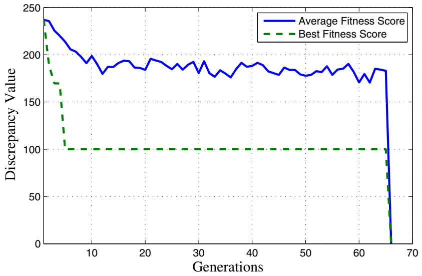


Figure 5. The consensus fitness history of the population

crossover rate specifies how many individuals go through the crossover operation in one population generation. The *mutation* operator relocates the chromosome parts within an individual to diversify the population.

In summary, the evolutionary process for the recovery mechanism is shown in Fig. 3. The exit criteria of the algorithm is the realizing of at least one configuration able supporting duplex mode with a full consensus, i.e., discrepancy-free outputs. The implication is that these configurations are those precisely which avoid known faulty resources as we discuss in the next section.

IV. Experiment Design

A 25 stage FIR filter is implemented in Verilog HDL using the Xilinx ISE 9.2i development tool. An ML410 development board is used which contains Virtex-4 FPGA chip, Compact Flash interface, DDRAM, and UART. For experimental purpose, the GA was simulated on a desktop PC rather than using the PowerPC on-chip processor at this time. Although the PowerPC is a hardware overhead of the GA-based SCDR recovery process, yet it is not on the critical throughput path. Thus, the use of PowerPC for just recovery purpose is likely to be amenable to most applications. The logic resources utilized by one instance of the TMR arrangement are 2810 number of LUTs and 500 FFs excluding those needed by the bus macros which are needed for partial reconfiguration flow in Xilinx ISE 9.2i.

Then, the circuit is replicated three times to realize a TMR system. The effect of random *Stuck-At (SA)* faults for injection into different stages of the FIR filter was realized by modifying the LUT contents in the post place-and-route simulation model. This simulates faults in different pipeline stages of a sequential circuit. An instance of the experiment is shown in Fig. 4, where faults are injected in the three stages of the $Instance_1$ and 5 stages of the $Instance_3$. The experiment objective is to achieve fault recovery of two instances constructing a CED pair.

V. Simulation Results

A. Intrinsic Hardware Evaluation using SCDR

The fitness history of the recovery process is shown in Fig. 5. These plots depict the average behavior and best behavior of the arrangements in each generation. Even with use of this realistic case study consisting of 25 stage FIR filter utilizing 2810 LUTs and with 8 faulty stages simultaneously present, the GA is able to converge to a minimum score within 70 generations. Here, the population size used was 50, the crossover rate was 0.8, and the size of the evaluation window was 100 input samples.

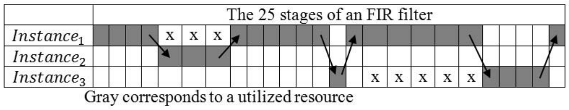

The 25 stages of an FIR filter
Instance1
Instance2
Instance3
x x x
x x x x x
Gray corresponds to a utilized resource

Figure 6. A repaired instance in the new configuration

The zero value fitness score of an individual corresponds to a TMR configuration in which two instances completely agree in terms of their output. A repaired instance after the genetic recovery process is shown in Fig. 6 where arrows depict the selected data pathway. It can be observed that this instance avoids any faulty resource, although any knowledge of faulty behavior of the resources was made unavailable. This demonstrates that consensus-based fitness evaluated over a sufficient window can provide a good approximation of the actual fitness of the system thus identifying faulty modules.

The amplitude spectrum of the input signal contains two periodic sine waves of frequency 50 Hz and 100 Hz. The cut-off frequency of the low-pass FIR filter is set to 75Hz. The spectrum of the output signal from the filter is shown in Fig. 7. After fault injection, the output spectrum is also given in Fig. 7 which is different than what is desired for the filter functionality. In addition, Fig. 7 depicts the output of the preferred recovered TMR arrangement after completing the SCDR fault recovery process. To quantify the recovery quality, the *Signal-to-Noise Ratio* (SNR) is defined by the ratio of signal power, P_x to the noise power, P_e. In this case, the SNR is computed by:

$$SNR = 10 * log_{10} \frac{P_x}{P_e} \tag{2}$$

The difference signal is defined by:

$$e(t) = x(t) - y(t)$$

where
$x(t)$ = Input signal to the filter
$y(t)$ = Output signal from the filter

It is evident from Fig. 7 that the SNR measure of the signal after fault recovery process of the system is identical to the original fault-free system. Thus, the desired functionality of the FIR filter is retained.

B. Faults-Aware Simulation Paradigm

To further evaluate the SCDR scheme, a simulation was performed in which the absolute fitness was assessed in the presence of knowledge about the faulty modules locations. The objective cost to be minimized is the utilization of a minimum number of faulty blocks on data pathway , and is defined by:

$$cost = \sum_{i=1}^{3} \sum_{j=1}^{n} Stage_j(Instance_i).FS \tag{3}$$

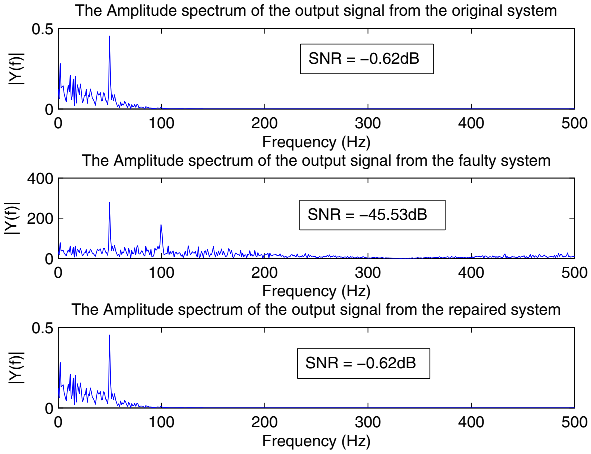


Figure 7. The amplitude spectrum of the output signal

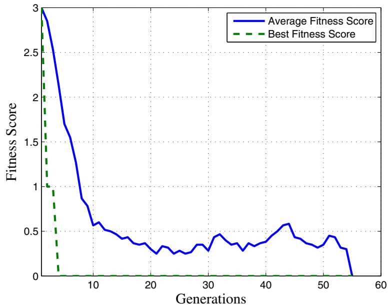


Figure 8. The absolute fitness history of the population

where
n = Number of pipeline stages in the circuit
$Stage_j(.).FS$ fitness state $\in \{0, 1\}$
In the above set, '0' corresponds to *Healthy* condition, and '1' corresponds to *Faulty* condition of a module.

Fig. 8 shows that although average behavior of the population for various generations is same in both the SCDR and the faults-aware simulation cases (i.e., Fig. 5 and Fig. 8), the best individual is found in smaller number of generations in the later case. It is due to the fact that the knowledge about faulty nature of the modules is made available to the fitness function and the GA readily evolves towards the objective of minimum number of faulty resources utilization. This case, however, utilizes ideal information which may not be available during fielded missions.

C. Performance Bound Comparison to Exhaustive Search

Finally, we compare with a case when all the possible configurations are evaluated exhaustively instead of employing a GA. As there are 3 instances, each with n variables, the total number of configurations combinations to be evaluated would be $N_E = n^3$.

For n=25 stage circuit, the upper bound on the required number of evaluations N_E becomes 15,625 which is much larger than the number of evaluations taken by the GA to reach the solution. As shown in Fig. 5, the population size of 50 is able to bring the correct configuration in 66 generations, thereby necessitating only 3,300 evaluations indicating effective recovery for the 25 stage FIR design. Future work will be to assess and improve recovery time which may be significant for mission-critical and safety-critical circuits. In addition, the focus will be to conduct generic analysis in presence of variable number of pipeline stages and granularity of fault handling.

VI. Conclusion

The SCDR adaptive fault-handling scheme is proposed for reconfigurable fabric based systems. It avoids resource testing through test vectors and utilizes discrepancy information during normal throughput computation. The fault coverage includes the utilized logic resources. The improved SNR as a result of the recovery scheme compared to that of a faulty system demonstrates the effectiveness of the approach. We evaluated the scheme using a typical application that is decomposable into distinct pipelined stages. Although, the case study is a very regular circuit, there is no loss of generality as long as the circuit can be fitted into various PRR stages. A future extension can be the development of a fault-tolerant pipelined microprocessor core using the SCDR scheme.

References

[1] R. F. DeMara and K. Zhang, "Autonomous FPGA fault handling through competitive runtime reconfiguration," in *Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on*, 29 2005, pp. 109 – 116.

[2] R. F. DeMara, K. Zhang, and C. A. Sharma, "Autonomic fault-handling and refurbishment using throughput-driven assessment," *Appl. Soft Comput.*, vol. 11, pp. 1588–1599, March 2011.

[3] D. Keymeulen, R. S. Zebulum, Y. Jin, and A. Stoica, "Fault-tolerant evolvable hardware using field-programmable transistor arrays," *Reliability, IEEE Transactions on*, vol. 49, no. 3, pp. 305–316, 2000.

[4] E. Stott, P. Sedcole, and P. Cheung, "Fault tolerant methods for reliability in FPGAs," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, 8-10 2008, pp. 415 –420.

[5] W. Rao, C. Yang, R. Karri, and A. Orailoglu, "Toward future systems with nanoscale devices: Overcoming the reliability challenge," *Computer*, vol. 44, no. 2, pp. 46 –53, Feb. 2011.

[6] C. Bolchini and C. Sandionigi, "Fault classification for SRAM-Based FPGAs in the space environment for fault mitigation," *Embedded Systems Letters, IEEE*, vol. 2, no. 4, pp. 107 –110, dec. 2010.

[7] K. Zhang, R. F. Demara, and C. A. Sharma, "Consensus-based evaluation for fault isolation and on-line evolutionary regeneration," in *in Proceedings of the International Conference in Evolvable Systems (ICES'05)*, 2005, pp. 12–24.

[8] N. Steiner and P. Athanas, "Hardware autonomy and space systems," in *Aerospace conference, 2009 IEEE*, march 2009, pp. 1 –13.

[9] I. Mansouri, F. Clermidy, P. Benoit, and L. Torres, "A run-time distributed cooperative approach to optimize power consumption in MPSoCs," in *SOC Conference (SOCC), 2010 IEEE International*, sept. 2010, pp. 25 –30.

[10] M. Abramovici, J. M. Emmert, and C. E. Stroud, "Roving STARs: an integrated approach to on-line testing, diagnosis, and fault tolerance for FPGAs in adaptive computing systems," in *Evolvable Hardware, 2001. Proceedings. The Third NASA/DoD Workshop on*, 2001, pp. 73–92.

[11] M. Gericota, G. Alves, M. Silva, and J. Ferreira, "Reliability and availability in reconfigurable computing: A basis for a common solution," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 16, no. 11, pp. 1545 –1558, Nov. 2008.

[12] S. Dutt, V. Verma, and V. Suthar, "Built-in-self-test of FPGAs with provable diagnosabilities and high diagnostic coverage with application to online testing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, no. 2, pp. 309 –326, Feb. 2008.

[13] C. Carmichael, "Triple module redundancy design techniques for virtex FPGAs," *Xilinx Application Note: Virtex Series*, 2006.

[14] K. Morgan, D. McMurtrey, B. Pratt, and M. Wirthlin, "A comparison of TMR with alternative fault-tolerant design techniques for FPGAs," *Nuclear Science, IEEE Transactions on*, vol. 54, no. 6, pp. 2065 –2072, dec. 2007.

[15] A. Jara-Berrocal and A. Gordon-Ross, "VAPRES: a virtual architecture for partially reconfigurable embedded systems," in *Proceedings of the Conference on Design, Automation and Test in Europe*, 2010, pp. 837–842.

[16] M. Kuehnle, A. Brito, C. Roth, K. Dagas, and J. Becker, "The study of a dynamic reconfiguration manager for systems-on-chip," *Annual Symposium on VLSI*, 2011.

[17] L. Shannon, "Leveraging reconfigurability in the design process," *International Conference on Field Programmable Logic and Applications*, vol. 0, pp. 731–732, 2005.

[18] S. Lopez-Estrada and R. Cumplido, "Hardware architecture for adaptive filtering based on energy-CFAR processor for radar target detection," *IEICE Electronics Express*, vol. 7, no. 9, pp. 628–633, 2010.

[19] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[20] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1985, pp. 101–111.