Reliable Power Efficient Systems through Run-time Reconfiguration

Nahla Elaraby*†, Axel Jantsch*
*TU Wien, Vienna, Austria
{nahla.el-araby, axel.jantsch}@tuwien.ac.at
†Canadian International College-CIC, Cairo, Egypt

***Abstract*—We propose a methodology for optimizing the reliability, power, area, and performance of Field Programmable Gate Array (FPGA)-based systems at run time based on dynamic reconfiguration. A partial reconfiguration manager is designed to switch between different versions of the design according to real operating conditions. Reliability, Power, area, and performance are factors used for selecting the design version to be configured. Power and Reliability conditions are assessed through monitoring the system power state and the environment radiations with a radiation sensor, respectively. The experimental results show 40% reduction in power consumption while keeping an average reliability of 99.5%.**

***Index Terms*—FPGA, Reliability, Power, Run time optimization**

I. Introduction

Power management in FPGA is a challenge as they are power hungry devices especially with the huge increase in size, density and also the complexity of designs. Different power management techniques are adopted, mostly Power gating (PG) and the Dynamic Voltage and Frequency Scaling (DVFS). Traditional power gating used in Application Specific Integrated Circuits (ASICs) cannot be used the same way in FPGAs. Different techniques can be used to decrease power dissipation in FPGAs, starting from clock gating to floorplanning and routing techniques. Researchers have proposed to include PG in High Level Synthesis (HLS) where the portions with low activities are predicted and gating staff is inserted [1]. Widely used dynamic power management techniques optimize the power performance trade-off by switching between different power states. Also energy-accuracy trade-off in self powered devices can be optimized by utilizing multiple design points.

The second challenge for FPGAs is reliability. The static Random Access Memory (SRAM) based technology of FPGAs, increase their susceptibility to different types of faults [2], especially the Commercial Off-The-Shelf (COTS) FPGAs that are not radiation hardened. The usage of FPGAs is growing in different embedded systems in domains like aerospace, military and automotive, where reliability is a very important aspect. Thus, the need for reliability management and fault tolerance techniques is growing simultaneously. A wide range of fault detection and correction techniques [3], as well as fault tolerant methods analysis are available for FPGAs [4]. Furthermore, a lot of recent work has investigated methods for increasing the reliability and fault tolerance of FPGAs [5], [6]. Also Dynamic Reliability Management techniques use the radiation data and the flexibility offered by dynamic reconfiguration to switch between redundancy levels or different reliability architectures.

However, all reliability management techniques suffer from area and power overhead due to redundancy added to increase the reliability leading to expensive structures like Triple Modular Redundancy (TMR). Therefore developing a complete hardware run-time power management framework according to a compromise between reliability, power, area and performance could highly contribute to the design optimization process.

In this paper we propose an automatic run-time reconfiguration manager that can run on the processing core of an System-on-Chip (SoC) FPGA. The reconfiguration manager takes input from a radiation sensor, battery level indicator, and a run-time power monitoring unit. The latter measures the power of each system module periodically through monitoring the toggling activity of selected indicative signals in a multiplexed way. The reconfiguration manager decides to reconfigure the FPGA in a power efficient way while keeping the best possible reliability configuration. A butterfly structure is used as a use case to test the proposed technique. The main contribution of the proposed technique is enabling the use of expensive high reliability configuration with minimized overhead through FPGA Dynamic Partial Reconfiguration (DPR).

In the next section we discuss the related work from the state of the art. Section III explains the details of the proposed technique. The experimental results are presented in IV. Finally we conclude our work and clarify our future directions in section V.

II. Related Work

Power consumption is a challenging factor in FPGA designs. Accordingly, several approaches for power estimation, monitoring, and optimization exist in the literature. Some recent approaches are also investigating runtime power monitoring [7] [8] [9]. In [7] an adaptive online model is built by combining measurements of register-level switching activity and system-level power; also, an automated flow is used to select signals predicted to indicate high power consumption for power monitoring. An earlier work in [10] investigated the

runtime power scaling capabilities of ZYNQ devices and proposed accurate and fine-grained power control and monitoring techniques. The open standard power-management protocol Power Management Bus (PMBus) included in modern FPGAs, which facilitates the communication with power converters and other devices in a power system, was employed. A statistical methodology is presented in [11] for building accurate runtime power models using performance monitoring counters (PMCs) for mobile and embedded devices. The Reinforcement Learning (RL) approach is explored in [12], where RL automatically learns an optimal control policy to improve Network-on-Chip (NoC) energy efficiency. PG and DVFS are deployed to reduce both static and dynamic power simultaneously. The work in [13] presented an FPGA-based high-level power estimation methodology, providing a common power model for multiple Intellectual Properties (IPs). Another approach for power estimation in FPGAs in [14] considers early prediction of net activity and interconnect capacitance, where empirical prediction models are developed suitable for use in the synthesis of power-aware layouts, early power estimation/planning, and other applications. Also, [15] developed linear programming-based time slack algorithm EdTLC-NW, based on min-cost network flow to reduce runtime power. A hybrid technique for Slow-Transition Fast Level (STFL) signaling that creates a balance between power and bandwidth in the last level cache interface is proposed in [16]. A Computer-Aided Design (CAD) flow for high-level dynamic power estimation on FPGA is proposed by [8]. The method uses event counters to monitor the toggling activity for relevant signals selected using the Greedy Stepwise filter. A power model and monitors are automatically generated to achieve the best trade-off between accuracy and overhead. Later the authors presented a method in [9] based on data mining algorithms to monitor the toggling activities on relevant signals. A generic flow is proposed to produce a power model for any Register-Transfer Level (RTL) circuit on any technology. A decision-tree-based power modeling approach is employed in [17] to establish fine-grained hardware power monitoring on FPGA platforms. Another technique is proposed in [18] that considers energy harvesting devices running on a limited energy budget to recognize user activities over a given period to co-optimize the accuracy and active time through switching between multiple design points with different energy-accuracy trade-offs. This approach is somehow similar to our approach, but the trade-offs we used is power-reliability. A multi-objective dynamic power management technique is proposed in [19] to achieve better short-term smooth power consumption and long-term reliability based on fine-grained voltage and frequency scaling and PG per-core as actuators.

Runtime reliability management in FPGAs is also investigated in literature employing the flexibility offered by dynamic and partial reconfiguration [20] [21] [22] [23] [24] [25]. DPR was also employed in [26]. A framework is presented, giving the designer freedom to choose the level of redundancy and fault mitigation based on the level of hazards. Some task-based architectures are also used for reliability enhancement using DPR, including some Real Time Operating System (RTOS) with a scheduler [27] [28] [29]. The concept of Dynamic Reliability Management (DRM) is presented in [30] DRM tool flow comprising a design-time tool to automatically generate several implementations of a single hardware design at different reliability levels and consequently with different performance factors. In runtime, a ReconOS architecture and a multithreaded programming model are used to switch among different reliability configurations. The work of [31] shows that 7.7 and 3.7 times better performance through Dynamic Reliability Management (DRM), where the radiation data is interpreted and used to select the proper redundancy level varying the reliability levels for different modules through partial reconfiguration.

State of the art shows various approaches for power monitoring and management, besides investigating the trade-off of power against performance, accuracy, or area. Reliability enhancement and fault tolerance in FPGAs is also well investigated. However, there exists no complete framework for optimizing power versus reliability.

We propose a novel technique for power-reliability optimization that employs runtime module power monitoring data to reconfigure the FPGA with optimum reliability architecture dynamically.

III. Methodology

The proposed methodology aims to increase the overall system reliability while keeping the lowest possible power consumption level. A run time reconfiguration manager is designed to select and configure the FPGA with a bit stream from a saved library. Each bit stream in the library corresponds to a different reliability configuration. Figure 1 shows a block diagram of the proposed technique. The inputs for the reconfiguration manager indicate the power and reliability run time conditions. A run time power monitoring unit is implemented to monitor the power consumption of specific design modules periodically. In the following sections we describe each block in more detail.

A. Run-time Power Monitoring Unit

The power consumption of basic modules or IP cores in run time can be done by counting the toggling of specific signals [7]. However counting the toggles of each signal from a module can strongly increase the area and power overhead. Therefore it is important to identify the most indicative signals that highly affect the power consumption for the monitored module. This can be done by sorting the signal activities based on previous generated switching activity files (.saif). The signals with the most activity are selected and an activity counter identifies the edges of these signals. The signal activity is used to calculate the power consumption using equation 1, where α_i is switching activity, C_i capacitance on the net i, V_{dd} is supply voltage and f is the operating frequency [32].

$$P_{dynamic} = \sum_{i \in N} \alpha_i C_i V_{dd}^2 f \tag{1}$$

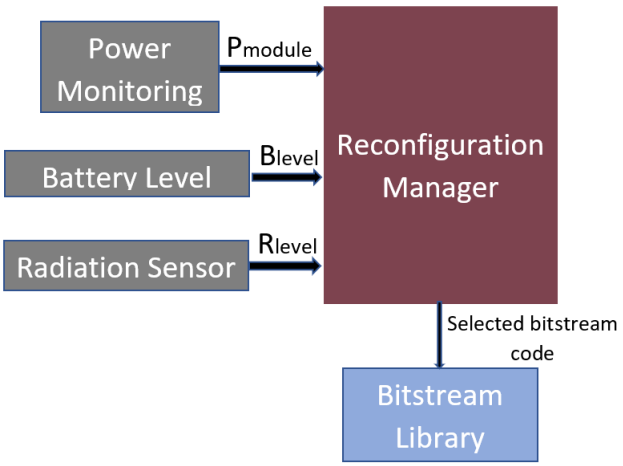

Fig. 1: Block Diagram for Proposed Mechanism

In this work a power monitoring module based on activity counters is implemented. The specific extracted set of indicative signals from each design module are multiplexed and passed to the run-time power monitoring module to be monitored periodically in a user specified time interval. The corresponding power value is calculated and passed to the configuration manager to be included in the optimisation function.

Signal Selection Technique: Signal selection starts with writing a testbench for the Xilinx Vivado power report, which should be as close as possible to the real world application. The next step is to synthesize and create the implementation of the module-to-be-monitored. Then adding the name of the Signal Activity file (saif) file in the simulation settings. Running the functional simulation on the implemented design creating the saif, followed by creating the power report by including the generated saif. Afterwards the signals and I/Os are sorted by highest to lowest switching rates then selecting the signals and I/Os with the highest activity.

B. Reconfiguration manager

A reconfiguration manager is designed to implement the logic needed for processing the inputs and generating the run-time decision for selecting the appropriate configuration bitstream. the configuration manager is implemented on the ARM core in the PS part of the Xilinx ZYNQ board. The functionality starts with deciding the suitable configuration, selecting the corresponding bitstream from the bitstream saved library, and finally reconfiguring the FPGA. The reconfiguration is done by the processing system via Processor Configuration Access Port (PCAP). The corresponding part is reconfigured automatically by reading the corresponding partial bitstream from an SD card.

C. TMR

TMR [33] is the most popular reliability enhancement technique especially for FPGA based systems. Two versions

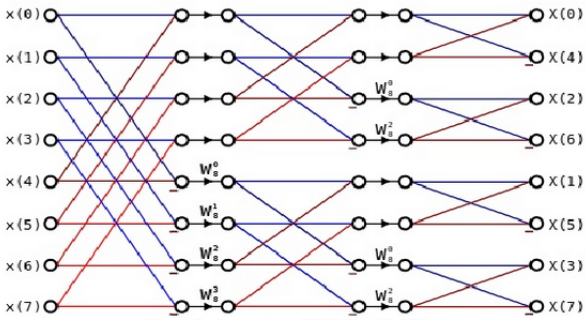

Fig. 2: FFT Butterfly structure

of each basic module in the target design are implemented, the simple basic version and the reliable version incorporating the TMR. Partial blocks are defined for each module to enable the switching between the two versions at run time without affecting the other modules. Switching between different configurations for different modules corresponds to a certain reliability level for the whole system. Equation 2 is used to calculate the reliability for a TMR structure, where R_M is the reliability of the module and R_V is the reliability of the voter.

$$R_{\text{TMR}} = (3R_M^2 - 2R_M^3)R_V \tag{2}$$

D. Use Case description

The FFT algorithm depends on a procedure of divide-and-conquer. The operation consists of dividing N coefficient points into smaller blocks in different stages [34]. The first stage computes with groups of two coefficients, producing N/2 blocks, each calculates the addition and subtraction of the coefficients scaled by the corresponding twiddle factors. The structure is called a butterfly for its cross-over appearance as shown in Figure 2 [35]. The following state of N/4 blocks uses the previous results, which will then combine the results of two previous blocks, combining four coefficients at this point. This process repeats until one main block is formed, with all N coefficients processed.

The Butterfly structure is used as a use case to test the proposed technique. Figure 3 shows the Butterfly structure implementation on Xilinx Vivado. The basic components are a complex twiddle multiplier, adder and subtractor. for each module of the three, the basic and TMR versions are designed and implemented using VHDL and synthesized on XILINX Zynq FPGA.

IV. Experimental Results and Discussion

The proposed methodology was applied to a butterfly structure to illustrate the technique. The butterfly structure consists of three modules: complex number Twiddle multiplier, adder, and subtractor. The design was implemented in different configurations employing the TMR as described in section III. Xilinx Zynq-7000 (xc7z020clg484-1) was used for implementation. Power analysis was performed. The results of power analysis was used to select the high activity signals to be connected to the run-time power monitoring unit. We used the

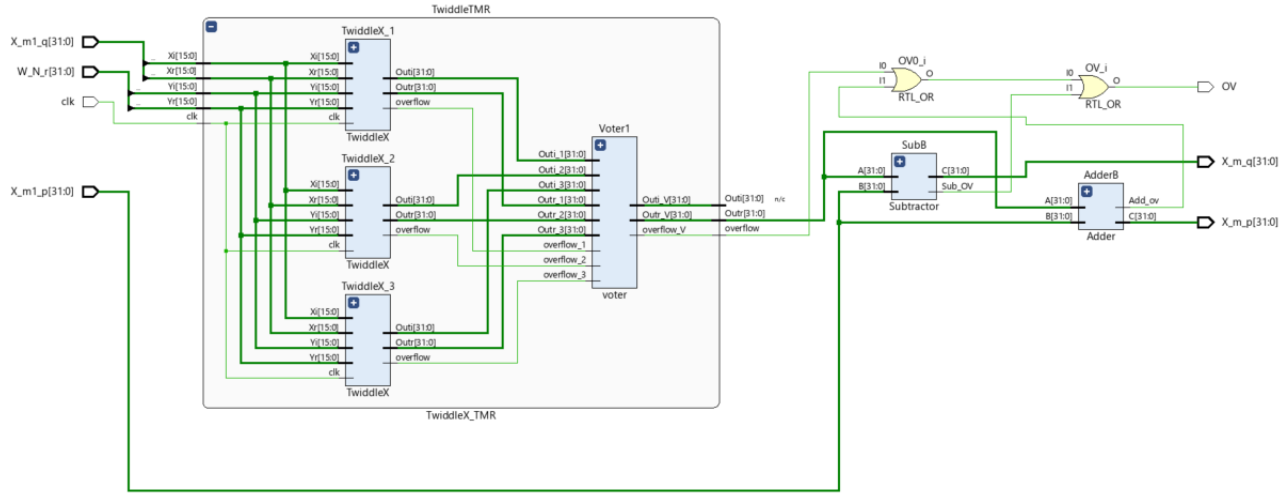

Fig. 3: Butterfly Schematic

TABLE I: Implementation Results

Configuration	Power(W)	Area (LUT)	Reliability
All Basic	0.663	339	0.9936
T_Basic A_Basic S_TMR	0.919	467	0.9937
T_Basic A_TMR S_Basic	0.919	467	0.9937
T_Basic A_TMR S_TMR	1.191	646	0.9983
T_TMR A_Basic S_Basic	1.445	659	1
T_TMR A_Basic S_TMR	1.717	838	1
T_TMR A_TMR S_Basic	1.717	838	1
All TMR	1.989	1017	1

reliability model provided by [36] as shown in equation (3). R_{FPGA} is the reliability of a design running on an FPGA, SER_{design} is the soft error rate of the design and λ is the failure rate of the FPGA device (failure rates are taken from the Xilinx device reliability report [37]). The use case is consisting of Twiddle multiplier connected in series with an adder and subtractor connected together in parallel, so we use equation 3 to calculate the reliability of each block then equations 4 and 5 for series and parallel connections to calculate the overall reliability in each of the implemented configurations.

$$R_{\text{FPGA}} = e^{-\lambda * t} * e^{-SER_{design}(t) * t} \quad (3)$$

$$R_S = R_1 x R_2 x R_3 x .....x R_n \quad (4)$$

$$R_P = 1 - [(1 - R_1)x(1 - R_2)x(1 - R_3)x....x(1 - R_n)] \quad (5)$$

Table I shows power, area, reliability results for different configurations of the butterfly structure. T is the Twiddle module, A is the adder module, and S is the subtractor module; T_basic A_TMR S_TMR means that the twiddle has the basic implementation while both the adder and subtractor include triple modular redundancy (TMR) implementations.

The reduction in power is calculated as a percentage from the maximum power consumption. We show the simulation results in (figure 4) where the power reduction and the average reliability are plotted for 10 simulation runs.

V. Conclusion and Future Work

In this paper, we propose a technique for dynamic optimisation for FPGA based designs. The flexibility offered by FPGAs and their dynamic and partial reconfiguration properties is used to optimise power and reliability automatically at run time. TMR is very expensive in terms of power and area, however our proposed technique employs the partial reconfiguration to enable automatic switching to TMR subject to the operation environment to provide the highest reliability with minimum cost over the total operation time. We showed through the results of our experiments that the power consumption can be decreased by 40% while keeping 99.5% average reliability.

In future work we plan to incorporate an optimising automatic solver in the reconfiguration manager running on the ARM processing system, so it can better select the optimal configuration at run time. An automated design flow tool will be developed to generate the different architectures of the design similar to the one proposed by [31].

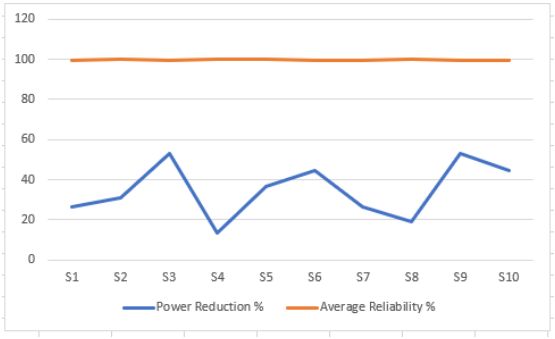

Fig. 4: Power consumption for different simulation runs

REFERENCES

[1] E. Choi, C. Shin, T. Kim, and Y. Shin, "Power-gating-aware high-level synthesis," in *Proceeding of the 13th international symposium on Low power electronics and design (ISLPED '08)*, pp. 39–44, 2008.

[2] M. Kvas, S. Valach, and P. Fiedler, "Reliability and Safety Issues of FPGA Based Designs," *IFAC Proceedings Volumes*, vol. 45, no. 7, pp. 201 – 206, 2012. 11th IFAC,IEEE International Conference on Programmable Devices and Embedded Systems.

[3] R. R. Fernanda Lima Kastensmidt, *Fault-Tolerance Techniques for SRAM-Based FPGAs*. Springer US, 2006.

[4] E. Stott, P. Sedcole, and P. Cheung, "Fault tolerance and reliability in field-programmable gate arrays," *IET Computers Digital Techniques*, vol. 4, pp. 196–210, May 2010.

[5] A. Kourfali and D. Stroobandt, "In-circuit fault tolerance for FPGAs using dynamic reconfiguration and virtual overlays," *Microelectronics Reliability*, vol. 102, p. 113438, 2019.

[6] V. M. G. Martins, P. R. C. Villa, R. Travessini, M. D. Berejuck, and E. A. Bezerra, "A dynamic partial reconfiguration design flow for permanent faults mitigation in FPGAs," *Microelectronics Reliability*, vol. 83, pp. 50 – 63, 2018.

[7] J. J. Davis, E. Hung, J. M. Levine, E. A. Stott, P. Y. K. Cheung, and G. A. Constantinides, "Kapow: High-accuracy, low-overhead online per-module power estimation for fpga designs," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 11, pp. 2:1–2:22, Jan. 2018.

[8] M. Najem, P. Benoit, F. Bruguier, G. Sassatelli, and L. Torres, "Method for dynamic power monitoring on fpgas," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–6, Sep. 2014.

[9] M. Najem, P. Benoit, M. El Ahmad, G. Sassatelli, and L. Torres, "A design-time method for building cost-effective run-time power monitoring," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, pp. 1153–1166, July 2017.

[10] A. F. Beldachi and J. L. Nunez-Yanez, "Accurate power control and monitoring in zynq boards," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–4, Sep. 2014.

[11] M. J. Walker, S. Diestelhorst, A. Hansson, A. K. Das, S. Yang, B. M. Al-Hashimi, and G. V. Merrett, "Accurate and stable run-time power modeling for mobile and embedded cpus," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, pp. 106–119, Jan 2017.

[12] H. Zheng and A. Louri, "An energy-efficient network-on-chip design using reinforcement learning," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2019.

[13] A. Lakshminarayana, S. Ahuja, and S. Shukla, "High level power estimation models for fpgas," in *2011 IEEE Computer Society Annual Symposium on VLSI*, pp. 7–12, July 2011.

[14] J. H. Anderson and F. N. Najm, "Power estimation techniques for fpgas," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 1015–1027, Oct 2004.

[15] Y. Lin, Y. Hu, L. He, and V. Raghunat, "An efficient chip-level time slack allocation algorithm for dual-vdd fpga power reduction," in *ISLPED'06 Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, pp. 168–173, Oct 2006.

[16] P. Behnam and M. N. Bojnordi, "Stfl: Energy-efficient data movement with slow transition fast level signaling," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2019.

[17] Z. Lin, W. Zhang, and S. Sharad, "Decision tree based hardware power monitoring for run time dynamic power management in fpga," in *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–8, Sep. 2017.

[18] G. Bhat, K. Bagewadi, H. G. Lee, and U. Y. Ogras, "Reap: Runtime energy-accuracy optimization for energy harvesting iot devices," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2019.

[19] A. M. Rahmani, M. Haghbayan, A. Miele, P. Liljeberg, A. Jantsch, and H. Tenhunen, "Reliability-aware runtime power management for many-core systems in the dark silicon era," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, pp. 427–440, Feb 2017.

[20] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Enhanced fpga reliability through efficient run-time fault reconfiguration," *IEEE Transactions on Reliability*, vol. 49, no. 3, pp. 296–304, 2000.

[21] G. I. Alkady, N. A. El-Araby, H. H. Amer, and M. B. Abdelhalim, "Using power consumption in the performability of Fault-Tolerant FPGAs," in *2016 5th Mediterranean Conference on Embedded Computing (MECO)*, pp. 55–58, June 2016.

[22] J. L. Nunes, J. a. C. Cunha, R. Barbosa, and M. Zenha-Rela, "Using partial dynamic fpga reconfiguration to support real-time dependability," in *Proceedings of the 13th European Workshop on Dependable Computing*, EWDC '11, (New York, NY, USA), p. 107–108, Association for Computing Machinery, 2011.

[23] M. Straka, J. Kastil, and Z. Kotasek, "Modern fault tolerant architectures based on partial dynamic reconfiguration in fpgas," in *13th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems*, pp. 173–176, 2010.

[24] G. I. Alkady, N. A. El-Araby, M. B. Abdelhalim, H. H. Amer, and A. H. Madian, "Dynamic fault recovery using partial reconfiguration for highly reliable FPGAs," in *2015 4th Mediterranean Conference on Embedded Computing (MECO)*, (Budva, Montenegro), pp. 56–59, IEEE, June 2015.

[25] G. I. Alkady, N. A. El-Araby, M. Abdelhalim, H. Amer, and A. Madian, "A fault-tolerant technique to detect and recover from open faults in FPGA interconnects," in *2014 14th Biennial Baltic Electronic Conference (BEC)*, (Tallinn, Estonia), pp. 69–72, IEEE, Oct. 2014.

[26] A. Jacobs, G. Cieslewski, A. D. George, A. Gordon-Ross, and H. Lam, "Reconfigurable fault tolerance: A comprehensive framework for reliable and adaptive fpga-based space computing," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 5, Dec. 2012.

[27] G. Wang, S. Liu, and J. Sun, "A dynamic partial reconfigurable system with combined task allocation method to improve the reliability of FPGA," *Microelectronics Reliability*, vol. 83, pp. 14–24, Apr. 2018.

[28] D. Sharma, L. Kirischian, and V. Kirischian, "Run-Time Mitigation of Power Budget Variations and Hardware Faults by Structural Adaptation of FPGA-Based Multi-Modal SoPC," *MDPI Computers*, vol. 7, p. 52, Oct. 2018.

[29] B. H. Sababha and Y. A. Alqudah, "A reconfiguration-based fault-tolerant anti-lock brake-by-wire system," *ACM Trans. Embed. Comput. Syst.*, vol. 17, Oct. 2018.

[30] J. Anwer, S. Meisner, and M. Platzner, "Dynamic reliability management: Reconfiguring reliability-levels of hardware designs at runtime," pp. 1–6, 12 2013.

[31] J. Anwer, S. Meisner, and M. Platzner, "Dynamic Reliability Management for FPGA-Based Systems," *International Journal of Reconfigurable Computing*, pp. 1687–7195, June 2020.

[32] Z. Lin, W. Zhang, and S. Sharad, "Decision tree based hardware power monitoring for run time dynamic power management in fpga," in *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1–8, 2017.

[33] U. Afzaal and J.-A. Lee, "A self-checking tmr voter for increased reliability consensus voting in fpgas," *IEEE Transactions on Nuclear Science*, vol. 65, no. 5, pp. 1133–1139, 2018.

[34] Y. S. Algnabi, R. Teymourzadeh, M. Othman, S. Islam, and M. V. Hong, "On-chip implementation of pipeline digit-slicing multiplier-less butterfly for fast fourier transform architecture," 2010.

[35] S. Jacob and V. V. Gopal, "Implementation of orthogonal frequency division multiplexing transceiver on fpga," 2014.

[36] O. Heron, T. Arnaout, and H.-J. Wunderlich, "On the reliability evaluation of sram-based fpga designs," in *International Conference on Field Programmable Logic and Applications, 2005.*, pp. 403–408, 2005.

[37] Xilinx, "Device Reliability Report," *Xilinx user guide*, 08 2015.