Advances in Engineering Software 82 (2015) 14–24

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

# Automated design space exploration of multi-cycle transient fault detectable datapath based on multi-objective user constraints for application specific computing

Anirban Sengupta *, Saumya Bhadauria

*Computer Science & Engineering, Indian Institute of Technology, Indore, India*

ARTICLE INFO

*Article history:*
Received 24 October 2014
Received in revised form 3 December 2014
Accepted 21 December 2014
Available online 10 January 2015

*Keywords:*
Multi cycle
Transient
Fault detectable
Design space exploration
Power
Delay
Bacterial foraging

ABSTRACT

A novel automated design space exploration (DSE) approach of multi-cycle transient fault detectable datapath based on multi-objective user constraints (power and delay) for application specific computing is presented in this paper. To the best of the authors' knowledge, this is the first work in the literature to solve this problem. The presented approach, driven by bacterial foraging optimization (BFO) algorithm provides easy flexibility to change direction in the design space through tumble/swim actions if a search path is found ineffective. The approach is highly capable of reaching true Pareto optimal curve indicated by the closeness of our non-dominated solutions to the true Pareto front and their uniform distribution over the Pareto curve (implying diversity). The contributions of this paper are as follows: (a) novel exploration approach for generating a high quality fault detectable structure based on user provided requirements of power-delay, which is capable of transient error detection in the datapath; (b) novel fault detectable algorithm for handling single and multi-cycle transient faults.

The results of the proposed approach indicated an average improvement in Quality of Results (QoR) of >9% and reduction in hardware usage of >23% compared to recent approaches that are closer in solving a similar objective.



## 1. Introduction

DSE in HLS includes searching an optimal datapath from a set of assorted design alternatives of equivalent functionality which offer higher performance, and lower power expenditure with complete fault reliability attribute. The aim of the exploration approach is to reduce the large and complex design space into a set of feasible design solutions meeting multiple designs performance along with certain orthogonal issues like runtime and QoR [5,8–13,23,26–28].

However, optimizing area, power and performance remains no longer sufficient now. Specifically, for current generation of systems which demand designs (especially for space applications where radiation induced faults are highly possible) that requires ability to detect errors occurring due to transient faults (such as single event upsets). Transient faults are radiation induced faults which are non-permanent in nature. These nonrecurring faults can be caused by energized particles, environmental noise or electromagnetic interference. The duration of such faults is in order of a few picoseconds [16,24]. The occurrence of transient faults is due to recent advancements in technology where the packing of millions of transistors on a single chip have become more feasible. The increase in density per unit area is negatively impacting the device and overall systems reliability by making it susceptible to transient faults or the single event upsets (SEU) [31] especially in space applications. Therefore, to achieve high reliability of the systems, fault detectability [31–36] should be considered as design metric (or constraint) during multi-objective DSE in HLS [25,34].

### 1.1. Motivation/background on using BFOA

*Summary:* BFOA uses a simplified framework and is less sensitive than other evolutionary techniques such as particle swarm optimization (PSO) and genetic algorithm (GA), yet capable of reaching high quality optimal solution. Therefore, it is better adaptable to the problem at hand which is considered intractable and NP hard. The detailed motivation for adopting BFOA is provided below:

A regular DSE process with power and delay as design objective is considered intractable [26]. Inclusion of multi-cycle fault detectability as additional design objective would intricate the mechanism and therefore shall require advanced search algorithm to

* Corresponding author.
*E-mail address:* asengupt@iiti.ac.in (A. Sengupta).

http://dx.doi.org/10.1016/j.advengsoft.2014.12.008
0965-9978/

find a quality design solution. Methodologies such as BFOA would be considered suitable for such notorious problems because other evolutionary algorithms such as GA, hybrid GA and PSO do not provide flexible options for guided/adaptive searching such as change in directions when a certain search path is found unproductive. Further, PSO is known to be a highly sensitive algorithm; therefore failing to clinically pre-tune the parameters often would result in convergence to local optima [38]. Additionally, due to multiple loops involved in BFOA such as chemotaxis and dispersal as well as options of tumble/swim (helping to change directions when required), the framework of this algorithm provides the flexibility to be configured in an proficient way for eliciting efficient search behavior during multi-dimensional DSE. BFOA comprises of primarily of two major steps: chemotaxis and dispersal for locomotion of bacterium. Using locomotion mechanisms (such as flagella) bacteria can move around in their environment, sometimes moving chaotically (tumbling and spinning), and other times moving in a directed manner that may be referred to as swimming. The algorithm was designed for application to continuous function optimization problem domains. Given the loops in the algorithm, it can be configured in numerous ways to elicit different search behavior. It is common to have a large number of chemotaxis iterations, and small numbers of the other iterations [37]. The steps on how locomotion movement of bacterium is employed for DSE of the given problem are described in Section 3.3.

## 2. Related work

So far in the literature, adequate attempts have been made to solve the DSE problem in HLS. The approaches developed so far aims at exploring the design space along with balancing some multi-conflicting issues during generation of the best possible solution (or Pareto front). Over the years the DSE process has evolved where the requirements specified by the user have also convoluted, ranging from simple area-delay tradeoff in initial years to complex power-delay tradeoff in recent years. Further, some HLS approaches included the consideration of fault security aspect with hardware redundancy, but without focusing on DSE of an optimized fault detectable design based on user power-delay constraint. Nevertheless, some recent approaches [1–6] are worth mentioning which are efficient DSE approaches/tools however without any consideration on fault aspect during HLS.

For example, authors in [1] have proposed simulated annealing (SA) DSE method called 'SALSA' for optimizing delay which uses many probabilistic search operators to enhance the performance of SA-based technique. Authors in [2] have applied GA to the binding and allocation phase. A specific crossover technique has been introduced which is based on force directed algorithm. The limitations of approaches [1,2] besides being unable to handle transient fault is that it does not consider power during design trade-off. In addition, Authors in [3] introduced a tool called AutoPilot for HLS. It performs C/C++/systemC-to-RTL synthesis. Although this tool performs area-performance-power tradeoff during DSE, however, the tool does not have ability to mask transient fault as well as only targets FPGA's. However, the approach presented in this paper is fault detectable as well as it handles power-performance constraints using novel adaptive bacterial foraging driven exploration approach. Further, authors, in approach [4] also used simulated annealing to generate optimum results; however, the work did not consider transient fault detection and correction during exploration. In [5], authors introduced a tool called SystemCoDesigner that offers rapid design space exploration with rapid prototyping of behavioral systemC models. An automated integrating approach is developed by integrating behavioral synthesis into their design flow. However, the approach besides being unable to handle transient fault, is also limited to area-delay tradeoff. Additionally, authors in [6] proposed an approach based on hierarchical and multiple clock domain HLS to target low power design on FPGA. The authors targeted FPGA unlike the proposed approach. Further, this work besides being unable to handle transient fault also does not consider power constraint as it only considers throughput. In addition, authors in [7] proposed a machine learning method for DSE which introduce a transductive experimental design that can wisely sample micro-architecture choices and use them for training in the learning model. The approach, besides being unable to detect faults does not consider power objective during exploration. But the proposed approach presented in this paper besides considering power and execution time (or delay) as design objective during exploration, also considers multi cycle transient faults during generation of an optimized datapath. Therefore, it ensures an optimal fault detectable datapath generation after exploration based on conflicting user constraints. Further, the proposed framework is also driven by an intelligent searching called bacterial foraging optimization algorithm which incorporates multiple loops such as chemotaxis tumble/swim to handle efficient guided searching during exploration.

In addition to above, there are works that only deal with the fault detection issue of the designs without ability to explore an optimized fault detectable datapath based on user specified power-delay, which makes the previous approaches significantly different than the one proposed in our paper. The mentioned fault secured approaches are discussed as follows preceded with the definition of fault detection: A circuit is referred as a fault detectable circuit; if for any single fault within the circuit, it either produces correct output or an incorrect one, thereby indicating that fault has occurred. Fault security/detectability is an attribute of a system that guarantees that either the result is correct, or that any observable error is reported. For example, in [17] authors use duplication of the control data flow graph (CDFG) and map the second onto the same hardware as the first, adding functional units (FUs) as needed. The technique uses the algebraic properties of associativity, distributivity, and commutativity to aid mobility in scheduling the duplicate CDFG and thus take better advantage of idle resources. The approach in [16] involves partitioning of the CDFG into regions or sub graphs. The authors presented a hardware redundancy based concurrent error detection (CED) approach which breaks the data dependences between the nodes. This is done to improve the sharing between normal and duplicate computations. The original and the duplicate computations which are represented by a region are performed on distinct hardware. This is done so that, every regions output can be compared to identify the faults within the regions. For this, voting on the results of the regions is done. In [15], a CED scheme is employed to detect and isolate the faults within a system while it is in use. In [18] authors investigated a method for exploring the tradeoff between the area and latency of the CED design in HLS. The approach sometimes used hardware redundancy or time redundancy or a combination of both to produce fault secure designs. Designs were made secure on the basis of check pointing introduced in the system. Instead of adding extra FUs for fault detection, they use re-computation on the same hardware using different allocations. Therefore approaches [15–18] are all fault detectable approaches (using hardware redundancy) with no provision of producing an optimized fault detectable datapath system based on conflicting power and delay constraint of user.

## 3. Proposed methodology

### 3.1. Motivation

Given a sample scheduled data flow graph (SDFG) in Fig. 1. It shows a scheduled data flow graph of an application which uses