A Basic Approach to Fault Tolerance of Data Paths of HLS-synthesized Systems and its Evaluation

Jakub Lojda, Zdeněk Kotásek
Brno University of Technology, Faculty of Information Technology, Centre of Excellence IT4Innovations
Bozetechova 1/2, 612 66 Brno, Czech Republic

{ilojda, kotasek}@fit.vutbr.cz

Keywords. High-level Synthesis, Data-Path, CatapultC, Fault Tolerance, Fault-Tolerant, Robot Controller, C++.

Abstract

As the complexity of digital systems grows alongside with the chip level integration, methods to synthesize digital circuits from a description on a higher level of abstraction are required more and more. Moreover, the chip level integration increase leads to higher susceptibility to faults. For some of electronic systems operating in environments with increased risk of failure occurrence (e.g. space, aerospace or medical systems), the requirement for reliable functionality is even more critical. One possible solution is to implement the so-called *fault tolerance* into these circuits. The concept of *fault tolerance* accepts the fact a fault can appear but the main goal of this approach is to ensure the fault will not propagate in the form of a wrong computation result nor show as a computation delay. The another approach, which is called *fault avoidance* solves the problem of faults through precise selection of more reliable components to build the system.

The higher level of abstraction can be achieved through the *High-level Synthesis* (HLS). The HLS is a methodology transforming an input description, that is usually in the form of an algorithm described in a higher-level programming language (e.g. *C++*) to its *Register Transfer Level* (RTL) representation. The output RTL is usually in the form of a synthesizable code (e.g. *VHDL*). Many methods to implement *fault tolerance* into HLS-synthesized systems exist, although these methods are based on a modification of the HLS methodology itself. In our approach, the input algorithm is modified in order to obtain the resulting RTL description in a *fault-tolerant* form. The overview of this approach is shown in Figure 1.

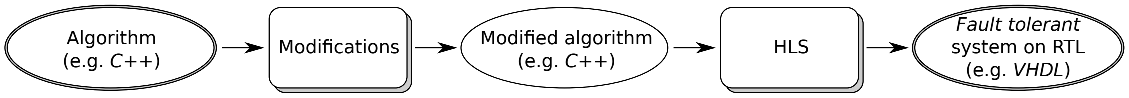
Algorithm (e.g. C++)
Modifications
Modified algorithm (e.g. C++)
HLS
Fault tolerant system on RTL (e.g. VHDL)

Figure 1: The basic approach to *fault tolerance* of HLS-synthesized systems data paths.

In the presentation, an effect of some of the main HLS optimization techniques to the susceptibility to failures of the resulting system will be evaluated. A basic data-types modifying approach, which increases the fault tolerance of the data-paths of the HLS resulting system will be presented as well. This approach will be demonstrated on the well known principle of the *Triple Modular Redundancy* (TMR).

Then an evaluation based on a case study utilizing a robot controller and the TMR principle with different levels of the HLS optimization techniques will follow. Although this approach is not limited to the *Field Programmable Gate Array* (FPGA) technology, the evaluation is based on the usage of an FPGA in conjunction with a platform for *fault-tolerant* systems properties evaluation. The evaluation platform we use in our research is based on the *functional verification* in combination with *fault injection*. It utilizes an FPGA in which the *Design Under Text* (DUT) is implemented and a PC that sends sensors data to the robot controller on the FPGA and also receives and evaluates the movements of the robot, while observing its ability to find and reach the target position in the simulated maze.

Paper origin

The original paper has been accepted and presented at International Conference on Field Programmable Technology in China [1].

Acknowledgment

This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II); project IT4Innovations excellence in science - LQ1602, ARTEMIS JU under grant agreement no 621439 (ALMARVI) and BUT project FIT-S-14-2297.

References

[1] J. Lojda, J. Podivínský, M. Krčma, and Z. Kotásek, “HLS-based Fault Tolerance Approach for SRAM-based FPGAs,” in *Proceedings of the 2016 International Conference on Field Programmable Technology*. IEEE Computer Society, 2016, pp. 297–298. ISBN 978-1-5090-5602-6.