A Study on the Verification and Validation of Programmable Logic Component in A Nuclear Power Plant

G. Y. Park[a], C. H. Jung[b] and D. I. Kim[a]
a Instrument & Control Division, Korea Institute of Nuclear Safety, 19 Kusong-dong, Yuseong, Daejeon, Korea.
b Research Division, Korea Institute of Nuclear Safety, 19 Kusong-dong, Yuseong, Daejeon, Korea.
{k703pgy, k148jch, dikim}@kins.re.kr

1. Introduction

The usage of programmable logic component, i.e., Complex Programmable Logic Device (PDL) or Field-Programmable Gate Array(FPGA), has been becoming popular in a commercial application because of its flexibility. As a part of this trend, the up-to-date component adopted in a nuclear facility is gradually increasing. Therefore, it is important to verify functions of the programmable logic component.

The objective of this study is provide a verification plan of programmable logic design appropriate to the nuclear Instrument & Control (I&C) system. To achieve our goal, we focus on the verification of safety-critical programmable logic design, i.e. avionics, and the guideline for nuclear safety system.

This paper represents the first phase result of the project to develop a verification and validation (V&V) method for the programmable logic components. Next section depicts a relationship between a formal development process and a verification process. And the conclusion and the future work is represented in the final section.

2. Programmable Logic Design Process

A software and hardware technology are employed in developing an application adopting programmable logic component. That is why programmable device is a hybrid device [1,2]. Verifying the component for commercial application is based on the hardware design as well as the software process. Additionally, the programmable logic design for nuclear facility requires compliance with the software design guidelines such as IEEE Std 7-4.3.2, "IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations."

This section focuses on the programmable logic design development process and the verification. Fig. 1 shows the verification process and the development process as a part of the software design.

2.1 Development Process

The development process mainly comprises two parts. The first part is generating behavioral code and verifying it. Functions implemented in the behavioral code should comply with specification requirements, and the behavioral code verification ensures its Design Correctness [3]. The other is a generating Register Transfer Level (RTL) code and the verification [4,5]. The RTL code is a functional logic using synthesizable constructs with a higher hierarchical level of abstraction than a gate level. A RTL code simulation ensures Design Performance [6,7], i.e., signal integrity or static timing, before synthesis process. A designer using programmable logic components has responsibility for the code generation and its verification within the development process. Furthermore, intermediate products during development process have influence on the verification process such as a testvector generation or a testbench design.

2.2 Verification Process

The verification process is implemented in four stages. The first stage is selecting features from the specification and the selected items are assigned a priority. In the next stage, the items generally fall into groups based on the similarity of its attributes such as similar configuration or granularity [5]. And each group forms a testcase. In the order of that priority, the

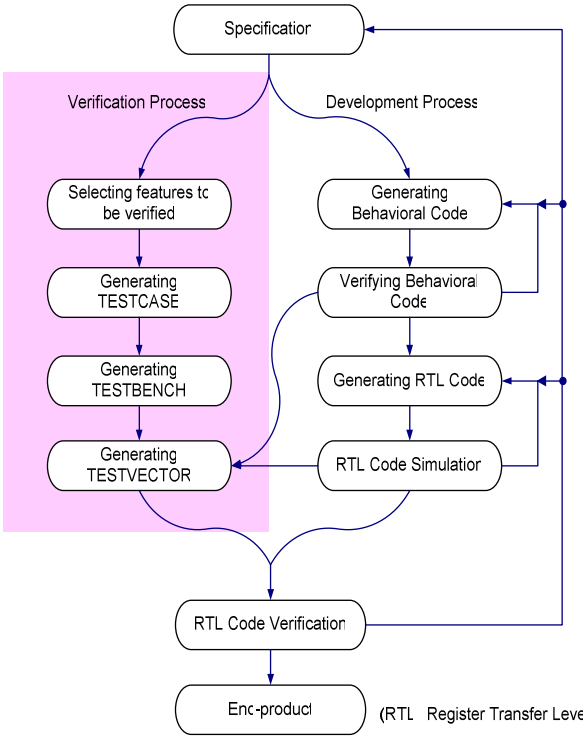

Figure 1. Relationship between the development process and the verification process.

important items should be included in the testcases. Similar to the testcases, a testbench comprises testcases which has similar attributes in the third stage. As shown in Fig. 1, the last stage of verification process is generating testvectors. The testvectors is stimuli for testbenchs to validate and verify functional correctness of the design as well as its design performance. The intermediate products such as the behavioral code verification results or the RTL code simulation result are included in generating the testvectors. The testbenches with the testvectors verify the function and the performance of the final RTL codes.

2.3 Efforts to End-product

Contrary to the software verification, The verification process of programmable logic is not independent from the development process. Namely, they are in a pseudo-relationship. The verification of software is in the form of 'black-box verification. Contrary, the programmable logic design verification occasionally has the form of gray-box or white-box verification. For example, it is impossible to verify the correctness of a complex embedded processor in the black-box verification. Therefore, verification engineers and developers should share the responsibility for the verification of end-product.

As shown in Fig. 1, the verification process is not independent from the development process. For example, intermediate products of the development process have an effect on generating a testvector. Furthermore, the programmable logic design may be modify for increasing the testability or verifying the design easily such as an insertion of scan chain. In the functional design aspect, the scan chain is not the functional requirement, but the additional design for verification.

3. Conclusion

The programmable logic designer and the verification engineer have responsibility for the design correctness and verification of the design, respectively. Moreover, they share the responsibility to ensure the performance of the end-product. The verification engineer can change the design for reducing the verification complexity and the designer is influence in the result of verification process.

There have been many research on software tools or verification languages which reduce the verification efforts. Furthermore, the conflict may occur between the existing guidelines for the nuclear safety system design and the verification process for the programmable logic design. We focus on finding these conflictions and resolving the problems. There have been developing a guideline to ensure the performance of programmable logic design as well. Through the result of previous researches on this topic and the efforts to resolve these conflictions, we provide a general verification plan of programmable logic design appropriate to the nuclear I&C system.

REFERENCES

[1] J. A. Cercone, M. A. Beims, and K. G. McGill, Verification and Validation of Programmable Logic Devices, National Aeronautics and Space Administration, 2004.
[2] DOT/FAA/AR 95/31, Design, Test, and Certification Issues for Complex Integrated Circuits, Federal Aviation Administration, 1996.
[3] T. D. Tessier, Rethinking Your Verification Strategies for Multimillion-Gate FPGAs, XAPP408, Xilinx, 2002.
[4] J. Bergeron, Writing Testbenches: Functional Verification of HDL Models, Kluwer Academic Publishers, 2000.
[5] J. Bergeron, Writing Testbenches using SystemVeliog, Springer, 2006.
[6] IEEE Std 1076.6, IEEE Standard for VHDL Register Transfer Level (RTL) Synthesis.
[7] IEEE Std 1800, IEEE Standard for SystemVerilog-Unified Hardware Design, Specification, and Verification Language.