A Study on the Differences in Verification and Validation Methodology between FPGA and Software

Dongil Lee [a*]
[a]*KHNP, Central Research Institute, 70, 1312-gil, Yuseong-daero, Yuseong-gu, Daejeon, 34101, South Korea*
[*]*Corresponding author: diturtle@khnp.co.kr*

1. Introduction

In the past, FPGA Verification and Validation (V&V) of NPPs (Nuclear Power Plants) used software V&V methods until the FPGA became a chip. By performing software verification without considering the FPGA as hardware, the hardware characteristics of the FPGA are not considered.

The software V&V method used to verify the hardware characteristics was verified by adding a System on Chip (SoC) design and verification method. This approach has resulted in differences in the V&V methods and activities.

In this paper, we will shed light on the V&V methods of software and FPGAs, and discuss important points in FPGA verification.

2. V&V Methods Comparison

This section discusses software V&V and SoC design and verification, and explain FPGA verification in the nuclear power field.

2.1 V&V of Software (IEEE 1012)

Software V&V are performed as management, acquisition, supply, development, operation, and maintenance processes, and V&V activities are performed for each process. The software level is determined by using Software Integrity Level (SIL) system provide IEEE 1012. And it is determined that the performed strength of the V&V tasks and the documentation scope according to the SIL [1, 2].

V&V activities and tasks can be performed by several organizations, including development organizations, business management organizations, quality organizations, and V&V organizations.

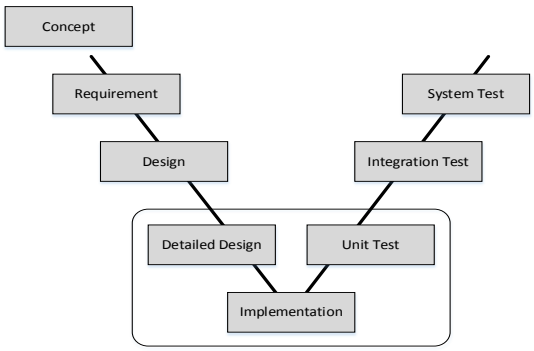

Fig. 1. V-model of software development

Software design and verification are performed with a V-model. Verification at the design stage consists of each stage (requirement, design, implementation), and validation consists of unit, integration, and system testing.

2.2 General FPGA Design and Verification in SoC

SoC design methods are divided into front-end and back-end methods. The front-end is the process of designing and synthesizing gate-level using Hardware Description Language (HDL) from the design specifications, and the back-end is the process of composing the synthesized netlist into the layout directly related to the semiconductor fabrication process.

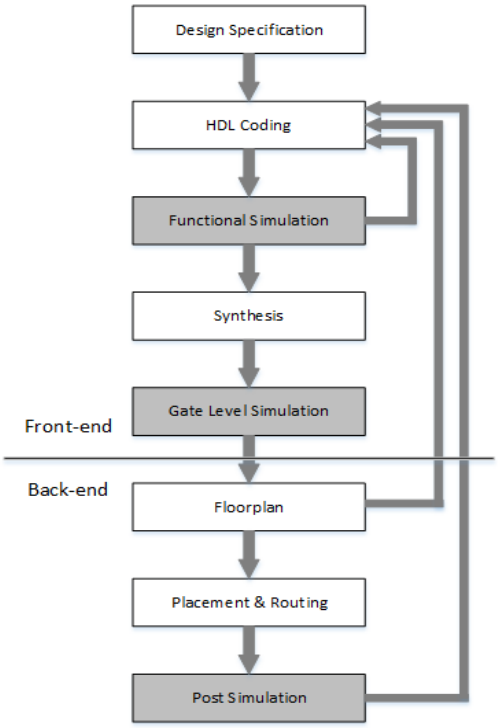

Fig. 2. SoC design flow

Fig. 2 shows the overall flow of the SoC design.

In the Design Specification phase, the design concept and configuration (function, performance, interface, etc.) are made and HDL is used in HDL-Coding phase.

After performing functional verification tests on the RTL (Register Transfer Level) configuration through a Functional Simulation, the RTL configured through the

synthesis process is changed to a Gate level netlist by applying restrictions, rules, and options (timing, size, power) of FPGA.

When the synthesis is completed, a Gate Level Simulation is performed using the information file of the timing delay of the signal. Place and route according to the circuit are then designed through Floor Plan and Placement & Routing. Finally, the final firmware is programed in the FPGA and a Post Simulation is performed to verify whether the design is abnormal.

2.3 FPGA Design and Verification in Safety Industry System (IEC 62566)

Since the hardware characteristics may be missed in the verification method using the traditional software V&V, IEC 62566 proposed a verification method by integrating the SoC design method and the software V&V [3].

The SoC design method employs a waterfall method, but the traditional V-model method is adopted in the nuclear power field. The choice of the V-model in the nuclear power felid is more focused on verification than design.

IEC 62566 suggests a V-model that reflects the SoC verification method in the implementation phase of the existing software V-model because all the specificity (hardware characteristic) of FPGA is not reflected in the general software verification method.

Therefore, IEC 62566 is presently applied to verification in domestic NPPs.

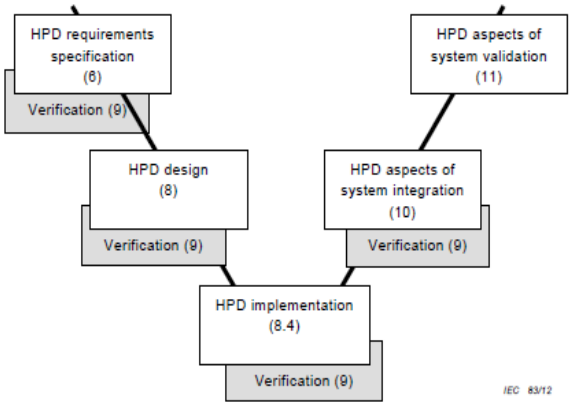

Fig. 3. IEC 62566-2012 V-model

3. Consideration of FPGA V&V

Generally, the source of FPGA (VHDL, VerilogHDL, etc.) is regarded as a software and is verified until it becomes firmware after being compiled and is programmed in FPGA.

At this time, if HDL is regarded as software only and verification is performed, the hardware characteristic may be missed, which may cause an error. The following lists some considerations to note in FPGA verification.

3.1 Coding Guide

HDL coding guidelines should be verified. Codes that do not follow the coding guidelines can cause false synthesis [4].

3.2 FSM (Finite State Machine) Verification

Logical verification of the FSM design is mandatory, and the FSM must specify all states.

3.3 Arrangement of Combination and Sequential Logic

Synchronous design should be performed by arranging sequential logic after combination logic.

3.4 Appropriateness of Test Bench Design

In FPGA design, the test bench for verification rather than design is the most important aspect. Functional simulation and timing simulation including delay information should be logically designed.

3.5 Signal Timing Analysis using Signal Waveform

All signals have a delay, and the delay of the signal affects data acquisition. Data integrity must be ensured by analyzing the signal waveform from the test bench.

3.6 Tool Version, Compile options

Management is important because the results vary depending on the option and version of the tool provided by the vendor.

4. Conclusions

FPGA is hardware, but HDL used for implementation is similar to the syntax of general programming language. Therefore, verification of the software aspect is necessary, but verification of the hardware characteristic is the most important aspect of the FPGA verification process.

In order to minimize FPGA design errors, FPGA V&V should verify characteristics of the FPGA and of the signals between FPGA and its surrounding ICs.

REFERENCES

[1] IEEE 1012, “IEEE Standard for Software Verification and Validation”, p.10, 2004
[2] IEEE 1012, “IEEE Standard for Software Verification and Validation”, Annex B, 2004
[3] IEC 62566, “Instrumentation and control important to safety - Development of HDL-programmed integrated circuits for systems performing category A functions”, p16, 2012
[4] NUREG/CR-7006, “Review Guidelines for Field-Programmable Gate Arrays in Nuclear Power Plant Safety Systems”, pp13-41, 2006