Transactions of the Korean Nuclear Society Spring Meeting
Gyeongju, Korea, May 29-30, 2008

A Preliminary Verification and Validation (V&V) Methodology for the Artifacts Programmed with a Hardware Description Language (HDL)

Yong Suk Suh [a*], Jong Yong Keum [a], Je Youn Park [a], Ki Ho Jo [b], Chang Whan Jo [b]
[a]*Korea Atomic Energy Research Institute, 1045, Daedeokdaero, Yuseong, Daejeon 305-303, Republic of Korea*
[b]*Control Technology Research Institute, Samchang Enterprise Co., Ltd.,974-1 Goyeon-ri Woongchon-myon, Ulju-gun, Ulsan 689-871, Republic of Korea*
[*]*Corresponding author: yssuh@kaeri.re.kr*

1. Introduction

Nowadays, the FPGA (Field Programmable Gate Array) is widely used in various fields of industry. The FPGA was evolved from the technology of PLD (Programmable Logic Device). The FPGA provides more logic gates than the PLD, which integrates millions of programmable logic gates into a chip. It also provides a massive, fast and reliable processing performance. So, we can integrate a system's functions into a FPGA, which can be a SoC (System on Chip). Furthermore, we can make a FPGA-based DSP, which DSP functions are implemented with a FPGA. With these merits, the FPGA is also used in the nuclear industry. For example, the safety-critical I&C component is manufactured with the FPGA.

The FPGA is programmed with a HDL. The quality of the artifacts programmed with a HDL can impact on the quality of a FPGA. When a hazard fault exists in the artifact of a FPGA and is activated during its operation, an accident caused by the fault in the FPGA occurs. So, it is necessary to ensure the quality of the artifacts.

This paper, for the purpose of applying it to the SMART (System-integrated Modular Adavanced ReacTor) MMIS project, is to present a preliminary V&V methodology for HDL programmed artifacts. For this, we reviewed the following items:

- Characteristics of HDL programming
- Applicable requirements for a HDL program used for the safety-critical systems
- Fault modes of a FPGA

Based on the review, we establish the preliminary V&V methodology.

2. Characteristics of a HDL Programming

The HDL programming is to arrange logic gates such as AND, OR, NOT, FLIP-FLOP to describe behaviors of the gates in a FPGA. IEEE 1076-1993 for VHDL or IEEE 1364-2001 for Verilog are published as standards for this programming. It is programmed with an editing, synthesis, simulation and implementation sequence. There are two ways of editing a program: a text-based and a graphic-based. The text-based programming can be done with a pseudo-C syntax. The graphic-based programming can be done with a combination of logic gates. There are so many tools for the programming such as ISE, QUARTUS, LABVIEW, State Diagram Editor, SystemC, etc. Most tools provide both the text-based and the graphic-based programming environments. They also provide IP (Intellectual Property) to enhance the programming performance. Since a HDL programming is possible with the pseudo-C syntax, the following structured programming features can be achieved:

- Top-down design approach
- Modulization (or process-based design)
- Bottom-up build is also possible

Whereas the C programming is to obtain a computation result, the HDL programming is to describe an electrical signal flow in a chip via logic gates. In the HDL programming, the programmers usually use a simulation method to verify the correctness of their logics. There are several phases of simulation: RTL (Register Transfer Level) simulation with a synthesized program, functional simulation with netlists, and timing simulation with an image in a FPGA. The works of editing and simulating a HDL programming is iteratively and concurrently performed until the artifacts are ensured by the programmers.

3. Requirements for a HDL Program used for the safety-critical systems

The HDL program is a part of firmware. In IEEE 7-4.3.2-2003, the firmware is defined as "the combination of a hardware device and computer instructions and data that reside as read-only software on that device." From this sentence, we understand that the firmware includes a hardware device and software. The standard, however, does not describe design criteria only for the firmware; instead it states "V&V activities and tasks shall include system testing of the final integrated hardware, software, firmware, and interfaces." With the two statements above, V&V activities for the firmware should include those of the hardware and software. The hardware part of the firmware should be qualified in accordance with the design criteria of the hardware and the software part in accordance with the software design criteria. For the software design criteria, the standard states "Computer software shall be developed, modified, or accepted in accordance with an approved software quality assurance (QA) plan consistent with the requirements of IEEE/EIA 12207.0-1996." The standard also requires the performance of a configuration management, a commercial dedication, and a hazard analysis for the firmware.

4. Fault modes of a FPGA

We considered fault modes of a FPGA in terms of a hardware fault and a software fault. The hardware fault mode is usually related to a short or open in circuits of a

FPGA. This fault is caused by a manufacturing defect or degradation during an operation. Stuck-at-faults (Stuck-at-0 and Stuck-at-1) are these type of faults. Baraza[1] and Jun Seok Lee[2] presented mechanisms to detect these type of faults. The software fault mode is a type of design error caused by humans. In order to reduce the design error, we need to not only educate more reliable humans but also make the artifact more fault tolerable. Whatever we take, we must assume there exists at least a design error in an artifact. The purpose of the V&V is to detect the design error.

5. A Preliminary V&V Methodology for a HDL program

We can assume that a HDL programming is the same case as we usually do in a generic programming. If this assumption is acceptable, the HDL programming should be qualified in accordance with IEEE/EIA 12207.0-1996. The fulfillment of IEEE/EIA 12207.0-1996 can be certified with CMMI-SW(Capability Maturity Model Integration-Software), SPICE(Software Process Improvement and Capability dEtermination), or ISO (International Organization for Standardization) 9001.[3] However, the CMMI-SW and the SPICE are not widely used in the nuclear industry.

As a minimum requirement, we require that HDL programmed artifacts be developed in accordance with the SMART MMIS SDLC (Software Development Life Cycle) that consists of the following phases: a concept, plan, requirement, design, implement, integration and validation, installation, and operation and maintenance.[4] In the SDLC, V&V activities are defined with the following activities: a review, test and analysis. These are required at each phase of the SDLC.

The most significant thing in a HDL programming is that it is hard to read the source code, to follow the flow of a signal, and to understand the whole package due to a highly complicated combination of logic gates. Because of these characteristics, it is required to perform the V&V activities to detect the faults.

There are two types of review activities: a walk-through and an inspection. The walk-through is an informal review activity, whereas the inspection is a formal activity. The performance of the review activities depends on the hardware experts. Checking the traceability between artifacts in a higher phase and artifacts in a lower phase is a key activity in the review.

As minimum testing activities, a unit testing, integration testing, and system testing should be performed. There are two types of techniques for each testing: a white-box testing and a black-box testing. The white-box testing focuses on the following coverages: a statement coverage to identify a dead code, a branch coverage to identify a faulty condition, and a toggle coverage to identify an unsigned value.[5] The black-box testing usually depends on the use of a simulation. The scenario for the black-box testing is as follows: applying a sequence of inputs, capturing the output response, comparing the response with an expected response, and analyzing the discrepancies between them. A test pattern (or a test vector) should be generated for the testing. During the simulation, a delay time should be checked throughout the signal flow from an input to an output. It is necessary to inject a test routine into a source code to check a physical defect such as the stuck-at-faults. This routine can be designed with a Boolean difference equation.

For a HDL programming, three types of modeling are applied as follows: a behavioral modeling including a concurrent modeling and a sequential modeling, a data flow modeling, and a structural modeling. And optimization technique such as a Karnaugh map is also applied in order to reduce the number of gates and the delay time in a circuit. The analysis activity checks if the optimization meets its requirements. During the analysis, logical proof is possible. However, it is hard to check the correctness of a timing sequence including the delay time. The performance of the analysis sorely depends on human fidelity.

6. Conclusions

A HDL programming is based on the knowledge of the hardware and software. The hardware knowledge is more important than software for the programming. The V&V for it should reflect this characteristic. This paper presents three activities for the V&V methodology: a review, test and analysis. Although this methodology is not applied to a practical FPGA design, it will be applied to a firmware design for the SMART MMIS safety systems. This paper does not present enough fault modes for a HDL program. We need to investigate and collect faults which have occurred during a HDL programming. It is necessary to study how the contents of a requirement specification can be traced through a design and implementation of the firmware.

REFERENCES

[1] Baraza, J.C., Gracia, J., Gil, D., Gil, P.J., “A prototype of a VHDL based fault injection tool: description and application”, Journal of Systems Architecture 47 (10), 2002, pp. 847–867.

[2] Jun Seok Lee, Man Cheol Kim, Poong Hyun Seong, Hyun Gook Kang, Seung Cheol Jang, “Evaluation of error detection coverage and fault-tolerance of digital plant protection system in nuclear power plants”, Annals of Nuclear Energy 33 (2006), pp 544–554.

[3] Yong Suk Suh, Heui Youn Park, Ki Sung Son, Ki Hyun Lee, Hyeon Soo Kim, “A Method to Improve the Software Acceptance Criteria for Nuclear Power Plants”, Trans. KNS Autumn Meeting, Korea, Oct. 2005.

[4] Yong Suk Suh, Jae Hong Park, In Soo Koo, Jong Myung Kim, Dong Cheol Park, Hyeon Soo Kim, “A Development of SDLC for MMIS of SMART Research Reactor”, Trans. KNS Autumn Meeting, Korea, Oct. 2004.

[5] David Dempster, Michael Stuart, “Verification Methodology Manual, 3rd Edition Techniques for Verifying HDL Designs”, (ISBN 0-9538-4822-1), Teamwork International, 2002.