Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 22, 2009

Design and verification Process for Developing the FPGA-based Firmware for Nuclear Power Plants

Joon-Ku Lee, Jong-Ryong Keum, Yong-Suk Suh, Je-Yun Park
Instrumentation & Control and Human Factors Research Division
Korea Atomic Energy Research Institute
P.O.Box 105, Yuseong, Daejeon, 305-353, Korea Rep. of
Mail to: jklee@kaeri.re.kr

1. Introduction

IEEE 7-4.3.2-2003 standard is used as basic criterion for the design of computer-based safety critical systems for nuclear power plants (NPPs). This standard deals with both hardware and software design aspects, which establishes the overall design criteria of the systems. It defines the firmware as the combination of a hardware device and computer instructions and data that reside as read-only software in that device. From this definition, the firmware contains characteristics of both hardware and software. The standard states that hardware quality criteria are addressed in IEEE 603-1998 and software in IEEE/EIA 12207.0-1996. It also states that a computer development process shall include the development of computer hardware and software. These statements shall be applied to the design of a computer-based safety critical system which is especially designed with firmware for NPPs. In this paper, we present design & verification process of firmware for Nuclear Power Plants.

2. Characteristics of Firmware based on FPGA

The FPGA delivers tremendous advantages of a flexibility, productivity and performance to the industries. With these advantages, it can be applied to the nuclear industry. When we take an FPGA into account to apply it to the design of computer-based safety critical systems for NPPs, it is used as a chip to design the firmware. In order to ensure its application, it is necessary to investigate the FPGA-based design characteristics and identify the inherent hazards in the design.

Thanks to a high flexibility in the FPGA-based design, it is said that software engineers can design logic devices easily. This is not practically true because the time-delay characteristic of the devices is very complicated to the software engineers. The FPGA inherently has a glitch problem due to this time-delay, which can be a hazard. We cannot avoid adopting asynchronous design features in a design. These features cause a metastability to occur because the arrival times of the inputs are different and the clocks are slack. Therefore, logic device designers should be able to identify the timing problems in a design and to modify or optimize the design. This modification or optimization also makes the logic very complex, even messy. The more a complex logic is, the more likelihood a hazard occurs. In order to develop the FPGA-based systems for NPPs, verification process is more important than design process because many tools exist to support the design but verification should be done by human at present. Verifying a FPGA-based system is a very tedious and time-consuming works. The design process and verification process should proceed iteratively, and together, until the logic and devices are ensured so that required functions are guaranteed and no hazards are found. There are still limitations on the application of FPGA to the safety critical systems for NPPs such that FPGA is not good for a floating point processing. Nevertheless with these limitations, I&C designers of NPPs try to use FPGA nowadays. However we should keep in mind that FPGA has also indeterminism characteristics. It is necessary to collect and classify the hazards relating to the firmware design with FPGAs.

3. Design and Verification Process of Firmware

3.1 RTL Design Process using Top-down Approach

In the design and implementation stage, engineer use the top-down approach. System is composed into block or macro. And then engineer can develop the macro using HDL, schematic, FSM, or IP core.

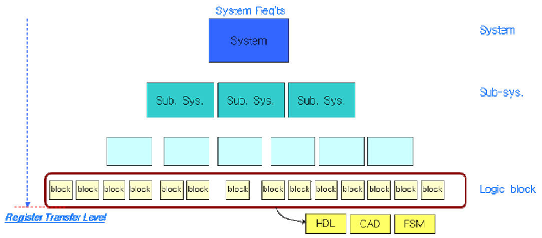

Fig. 1. Levelized hierarchy for RTL Design Process

3.2 Description Method using VHDL

There are three description method using VHDL. Generally, engineer use the mixed description.

1) Behavioral Description
   - Functional or algorithmic expression
   - Similar to high-level language programming
2) Data flow Description
   - Less abstractive than Behavioral description
   - Close to Boolean function, RTL, Gate description
   - Express operation of each logic circuit
3) Structural Description
   - Close to Hardware description

- Express not only each logic circuit but also interconnection

4) Mixed Description
   - Mixed use of above three descriptions
   - General method in design

3.3 Logic Synthesis

Logic synthesis is important process of FPGA design. Synthesis performs the translation, optimization, and mapping. In synthesis process, we must focus on making the shortest path and reducing a number of gates.

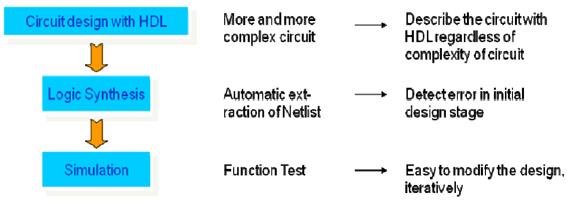
Circuit design with HDL
Logic Synthesis
Simulation
More and more complex circuit
Describe the circuit with HDL regardless of complexity of circuit
Automatic extraction of Netlist
Detect error in initial design stage
Function Test
Easy to modify the design, iteratively

Fig. 2. Logic Synthesis Process

3.4 Issue in developing the verification process

Software fault is a type of design error caused by human. In order to reduce the design error, we need to not only educate more reliable human but also make artifact more fault tolerable. Whatever we take, we must assume there exists at least a design error in an artifact. The purpose of V&V is to detect the design error.

3.4.1 Potential hazards in the design process

In the design process, Potential hazards is below

1) Schematic : wrong interconnection, non-existing logic
2) HDL : definition of the wrong number of input/output in entry stage, wrong data flow, unreachable statements
3) FSM : existence of non-reliable state, wrong transition between states, unreachable states
4) IP Core : use of unverified or uncertified IP cores, functionality of IP core is not fully ensured

3.4.2 Potential hazards in FPGA

In design and implementation stage, potential hazards are below. mostly, timing hazard.

1) Clock slack is the loose state of clock timing
2) Clock skew is caused by different arrival time in different flip flop with same clock
3) Setup, hold time violation is mostly, caused by asynchronous timing
   -Setup time violation exists unless that input is stable after clock edge changes
   -Hold time violation is caused by input-data-change when clock edge changes
4) Glitch is the inherent characteristics of logic design caused by asynchronous timing
5) Meta stability is the undefined state , when output of flop-flop is not 0 or 1, mostly caused by setup/hold time violation
6) Time variation such as setup, hold time violation may be caused by device heating

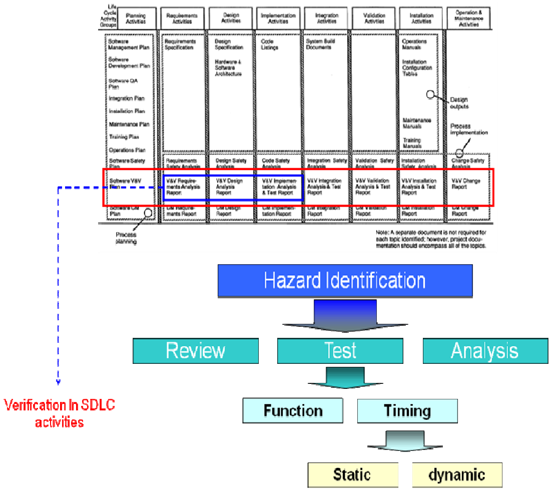

Fig. 3. Software Life Cycle activities in NUREG-0800 & Hazard Identification

3.5 Hazards In Optimization

It is important to reduce the hazards. The optimization techniques are setup time & hold time fixing.
Setup time fixing techniques are below.

1) Reduce combinational logic delay by minimizing number of logic levels
2) Split the complex combinational logic
3) Implement Pipelining
4) Use double syncronizer using flip flops

Hold time fixing techniques are below.

1) Can be fixed by adding delays on input ports, if delay time is needed
2) Adjust clock speed

4. Conclusions

We presented a design & verification process for a firmware-based on the designs for Nuclear Power Plants. We also dealt with the characteristic of the firmware due to this time-delay and so on. When we design the firmware for NPPs, we must know characteristics of FPGA and define the requirements & specification of application of the FPGA, exactly. In the design stage, top-down approach will enhance the generic design process.

REFERENCES

[1] Y.S.Suh, J.Y.Keum, J.Y.Park, K.H.Jo, C.W.Jo, A Preliminary Verification and Validation Methodology for the Artifacts Programmed with a Hardware Description Language(HDL), Transaction of the Korean Nuclear Society Spring Meeting, 2008.