Equivalence Checking between Pre-synthesis and Post-synthesis Programs by Using VIS

Jong-Hoon Lee [a*], Junbeom Yoo [a], Jong Gyun Choi [b], Jang-Soo Lee [b]
[a]*Division of Computer Science and Engineering, Konkuk Univ., 1 Hwayang-dong, Gwangjin-gu, Seoul, Korea*
[b]*Korea Atomic Energy Research Institute, 111, 989 Daedeok-daero, Yuseong-gu, Daejeon, Korea*
[*]*Corresponding author: kirdess@konkuk.ac.kr*

1. Introduction

PLC (Programmable Logic Controller) [1] has been widely used to implement real-time Software in PRSs (Reactor Protection Systems). Recently, there have been attempts to implement software in RPSs by using FPGA (Field-Programmable Gate Array) [2]. In PLC-based Software development, the design programs are translated into implementation programs, and behavioral equivalence between the design and implementation is demonstrated by formal method based technique. In FPGA-based software development, the design programs are also synthesized into implementation programs. However, in this process, testing and simulation based comparison techniques are mainly used. This paper proposes a formal method based technique to demonstrate behavioral equivalence between pre-synthesis and post-synthesis programs with VIS (Verification Interacting with Synthesis) verification system [3]. We translated into BLIF-MV which is front-end of VIS, from Verilog and EDIF netlist which synthesized from the same Verilog by an automatic synthesis tool.

2. Background

2.1 FPGA-based Software Development

FPGA is an integrated circuit designed to be configured by a customer or a designer after manufacturing in field. The FPGA configuration is generally specified using a HDL such as Verilog HDL and VHDL. Many vendors provide various types of FPGA in industry. *Xilinx*, *Altera*, *Actel*, *Semiconductor*, *Lattice*, *Sypress*, *QuickLogic* and *Atmel* are the examples. Nuclear industry has tried to use the products of *Altera* and *Actel*.

A typical FPGA-based software development process is consists of requirements analysis, design and implementation phases. Software requirements are analyzed and refined in requirements analysis and preliminary design phases. In design phase, we need to manually program the design in HDLs, and then HDLs are synthesized into gate-level netlists.

After programming HDLs, an FPGA is produced mechanically thorough several steps, such as optimization, synthesis, mapping, placement & routing, configuration and downloading. EDA (Electronic Design Automation) tools provided by FPGA vendors such as '*Altera Quartus II*' [4] and '*Actel Libero IDE*' [5] support all step seamlessly and mechanically. They also provide systematic verification and simulation facilities for each synthesis step.

2.2 EDIF

EDIF (Electronic Design Interchange Format) [6] is a vendor neutral format in which to store netlists and schematics. It was one of the first attempts to establish a neutral data exchange format for the EDA industry. The version of EDIF developed from 2 0 0 till the final version 4 0 0 [7~9]. The newer versions of EDIF have a richer feature set, however the FPGA vendors seems to have standardized on EDIF 2 0 0. Furthermore, EDIF from one vendor does not quite work with another vendors, the reason is that each tool of FPGA vendors reads and writes different style of EDIF.

2.3 VIS

VIS [3] is a system for formal verification, synthesis, and simulation of finite state systems. It has formal equivalence checking of combinational and sequential circuits. It provides *vl2mv* tool [10], which translates a subset of Verilog into an intermediate format BLIF-MV [11]. VIS also provides *edif2blif* tool, which translated an EDIF netlist into BLIF-MV. However, *edif2blif* does not work with EDIF from automatic synthesis tool which provided by FPGA vendors, hence the different style of EDIF as explained in section 2.2.

3. Formal Equivalence Checking with VIS

This section explains proposed technique. Section 3.1 describes an overview of proposed technique. Section 3.2 shows a result of equivalence checking with VIS.

3.1 An Overview of Equivalence Checking with VIS

FPGA vendors provide their own EDA tool which include automatic synthesis tool. An automatic synthesis tool synthesis from HDLs into netlists, then testing and simulation based comparison techniques are mainly used to demonstrate behavioral equivalence between HDLs and netlists. However, the vendor provided automatic translator should demonstrate functional

correctness rigorously. To accomplish this need, we propose a formal method based verification technique for FPGA-based software for nuclear industry.

Our approach is to confirm correctness of synthesis by using formal equivalence checking of VIS. A proposed process as depicted in Fig. 1 is consists of formal equivalence checking and two ways of translation. One way is translation from Verilog program into BLIF-MV which is the front-end of VIS by *vl2mv*. The other way is translation from EDIF netlist from EDIF netlist which is synthesized by automatic synthesis tool into BLIF-MV. If two programs are equivalent, we can confirm correctness of the synthesis.

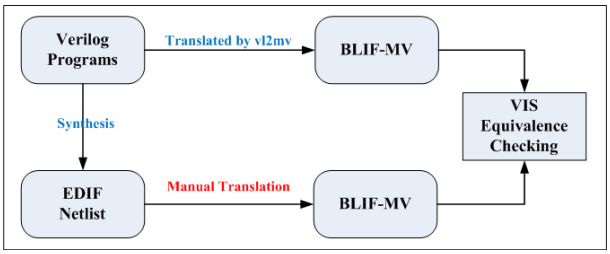

Fig. 1. An overview of proposed technique.

3.2 A Result of Equivalence Checking with VIS

To confirm effectiveness of a proposed technique, we performed a case study with a part of a prototype-version of RPS BP (APR-1400) [12]. It designed with FBD program, for this reason, we translated FBD program via '*FBDtoVerilog*' which developed in our former researches [13][14]. '*FBDtoVerilog*' is an automatic translator that translates into behaviorally equivalent Verilog programs from FBD program.

We synthesized a Verilog program by using '*Synplify Pro*' [15] automatic synthesis tool which included in '*Actel Libero IDE*' [5]. '*Synplify Pro*' can generate EDIF as an output format of synthesized netlist. Then we manually translated into BLIF-MV from synthesized EDIF netlist. Another BLIF-MV is translated from a Verilog program by using *vl2mv*. Then we demonstrated behavioral equivalence between two BLIF-MVs. Fig. 2 is a result of VIS equivalence checking, it shows two programs are equivalent.

```
vis> read_blif_mv from_edif.mv
Warning: Some variables are unused in model th_LO_SG1_LEVEL_Trip_Logic.
vis> flatten_hierarchy
vis> seq_verify from_verilog.mv
Networks are sequentially equivalent.

vis>
```

Fig. 2. A result of VIS equivalence checking.

3. Conclusion

This paper proposes a formal method based technique to confirm correctness of synthesis by using equivalence checking of VIS verification system. In order to confirm the effectiveness of the proposed technique, we performed a case study with a part of prototype version of the RPS BP, and demonstrated a behavioral equivalence between Verilog HDL and EDIF post-synthesis netlist.

ACKNOWLEDGEMENTS

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the Development of Performance Improvement Technology for Engineering Tool of Safety PLC (Programmable Logic Controller) program supervised by the KETEP (Korea Institute of Energy Technology Evaluation and Planning) (KETEP-2010-T1001-01038) and a grant from the Korea Ministry of Strategy, under the development of the integrated framework of I&C conformity assessment, sustainable monitoring, and emergency response for nuclear facilities.

REFERENCES

[1] Wikipedia: Programmable logic controller. http://en.wikipedia.org/wiki/Programmable logic controller
[2] Wikipedia: Field-programmable gate array. http://en.wikipedia.org/wiki/FPGA
[3] R. K. Brayton, G. D. Hachtel, A. Sangiovanni-Vincentelli, F. Somenzi, A. Aziz, S. T. Cheng, S. A. Edwards, S. P. Khatri, Y. Kukimoto, A. Pardo, S. Qadeer, R. K. Ranjan, S. Sarwary, T. R. Shiple, G. Swamy, T. Villa, VIS : A System for Verification and Synthesis., The 8th International Conference on Computer Aided Verification(CAV), p.428-432, 1996.
[4] Altera: Altera Quartus II. http://www.altera .com/products/software/
[5] Actel: Actel Libero IDE. http://www.actel.com/products/software/
[6] Wikipedia: Electronic design interchange format. http://en.wikipedia.org/wiki/EDIF
[7] EIA: Electronic Industries Association: Electronic design interchange format version 2 0 0, 1988.
[8] IEC: International Electrotechnical Commission: Electronic design interchange format version 3 0 0, 1993.
[9] IEC: International Electrotechnical Commission: Electronic design interchange format version 4 0 0, 1996.
[10] S. T. Cheng, G. York, R. K. Brayton, VL2MV: A Compiler from Verilog to BLIF-MV, HSIS Distribution, 1993.
[11] R. K. Brayton, BLIF-MV: An Interchange Format for Design Verification and Synthesis, Electronics Research Laboratory, College of Engineering, University of California, 1991.
[12] Korea Atomic Energy Research Institute (KAERI), Software Design Specification for Reactor Protection System, KNICS-RPS-SD231 Rev.02, 2006.
[13] J. Yoo, J. H. Lee, S. Jeong, S. Cha, FBDtoVerilog: A Vendor-independent Translation from FBDs into Verilog Programs, The 23rd International Conference on Software Engineering and Knowledge Engineering(SEKE), p.48-51, 2011.
[14] D. A. Lee, J. Yoo, J. S. Lee, Equivalence Checking between Function Block Diagrams and C Programs using HW-CBMC, The 30th International Conference on Computer Safety, Reliability and Security, p.397-408, 2011.
[15] Synopsys: Synplify Pro, http://www.synopsys.com/tools/implementation/FPGAimplementation/FPGAsynthesis/pages/synplifypro.aspx