Verification Process of Behavioral Consistency between Design and Implementation programs of pSET using HW-CBMC

Dong-Ah Lee*, Jong-Hoon Lee and Junbeom Yoo
Division of Computer Science and Engineering, Konkuk University
1 Hwayang-dong, Gwangjin-gu, Seoul, 143-701, Korea
**Corresponding author: ldalove@konkuk.ac.kr*

1. Introduction

Controllers in safety critical systems such as nuclear power plants often use Function Block Diagrams (FBDs) to design embedded software. The design is implemented using programming languages such as C to compile it into particular target hardware. The implementation must have the same behavior with the design and the behavior should be verified explicitly. For example, the pSET (POSAFE-Q Software Engineering Tool) [1] is a loader software to program POSAFE-Q PLC (Programmable Logic Controller) and is developed as a part of the KNICS (Korea Nuclear Instrumentation & Control System R&D Center) [2] project. It uses FBDs to design software of PLC, and generates ANSI-C code to compile it into specific machine code. To verify the equivalence between the FBDs and ANSI-C code, mathematical proof of code generator or a verification tools such as RETRANS [3] can help guarantee the equivalence. Mathematical proof, however, has a weakness that requires high expenditure and repetitive fulfillment whenever the translator is modified. On the other hand, RETRANS reconstructs the generated source code without consideration of the generator. It has also a weakness that the reconstruction of generated code needs additional analysis

This paper introduces verification process of behavioral consistency between design and its implementation of the pSET using the HW-CBMC [4]. The HW-CBMC is a formal verification tool, verifying equivalence between hardware and software description. It requires two inputs for checking equivalence, Verilog for hardware and ANSI-C for software. In this approach, FBDs are translated into semantically equivalent Verilog program [5], and the HW-CBMC verifies equivalence between the Verilog program and the ANSI-C program which is generated from the FBDs.

2. Background

2.1 Translation from FBDs into Verilog

Our approach of the verification using HW-CBMC first translates FBDs into Verilog. The translation rules were proposed in [5]. The rule consists of three parts, corresponding to unit, component and system FBDs respectively. First part describes how a unit of FBD is translated into a function in Verilog language. It first determines Verilog function type, and each input and its type is declared. And then behavioral description is followed, such as arithmetic, logic or selection operations. Second part explains translation rules for component FBDs. The component FBD is a logical block of independent function blocks which a number of function blocks are interconnected with to generate meaningful outputs. The rules of the second part declare component FBD's name, ports, wire type variables and reg type variables. Every function block translated as a Verilog function and included in the definition of module for the component FBD, and they called if there is a function according to its execution order to generate outputs of the component FBD. The last part describes how a system FBD is translated into a Verilog program. A system FBD contains a number of component FBDs which is translated in Verilog Modules, and their sequential interconnections. Verilog modules are instantiated and called according to their execution order with outputs communicated.

2.2 HW-CBMC

The HW-CBMC is a formal verification tool for verifying behavioral consistency between two implementations of the same design: one written in ANSI-C, and one written in register transfer level HDL like VHDL or Verilog. Motivation of the HW-CBMC is to reduce additional time for debugging and testing of the HDL implementation by providing automated way of establishing the consistency of HDL implementation using the ANSI-C implementation as a reference, because debugging and testing cost of the ANSI-C implementation is usually lower.

In this paper, the HW-CBMC is used for verification of behavioral consistency between Verilog program translated from FBDs and ANSI-C program generated from the FBDs. The data in the Verilog modules is available to the ANSI-C program by means of global variables, and we can check change of two programs' data using the function *assert(condition)* in C program.

3. Verification Process

Fig.1 shows verification process of behavioral consistency between design written in FBD and its implementation written in ANSI-C program. The pSET translates FBDs program into ANSI-C program using code generator. To verify equivalence between the FBDs and ANSI-C programs, we translated the FBDs into seman-

tically equivalent Verilog program using automatic translator named *FBDtoVerilog* [5]. The ANSI-C program and translated Verilog program are two inputs of the HW-CBMC to check equivalence between both. Automatically translated Verilog program, however, is not exactly suitable for the HW-CBMC. The following subsection describes how to make the Verilog program suitable for the HW-CBMC.

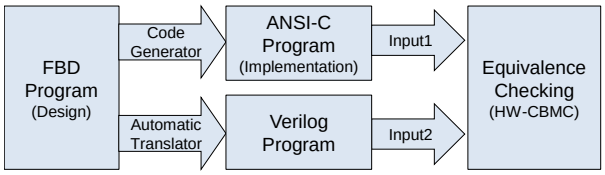

Fig. 1. Verification Process of Behavioral Consistency between FBDs and ANSI-C Programs of the pSET using the HW-CBMC

3.1 Modification of Verilog

One feature of the automatically translated Verilog program is modules have an output which has same name with the name of its module. The HW-CBMC, however, cannot access the output variable to check its change. We should modify the name of module's output described in Fig. 2. Another feature is that the translator translates a function block into a function. We also should modify functions into modules, because the HW-CBMC does not allow function calls. Fig. 2 also shows difference between a function and a module named *Equal*. The arguments of a module depend on its function.

Translated Verilog program	Modified Verilog program
module module1 (clk, IN, module1); … output main_module; … function Equal; … endfunction … endmodule	module module1 (clk, IN, module1_out); … output main_module_out; … endmodule module Equal(in1,in2,out); … endmodule

Fig. 2. Modification of automatically generated Verilog program

3.2 Verification using HW-CBMC

The ANSI-C program should have statements provided by the HW-CBMC to verify equivalence with Verilog program. For example, a function *set_input()* is inserted to synchronize the inputs of ANSI-C and Verilog program, and a function *next_timeframe()* makes transition of Verilog program once. A function *assert(conditions)* checks conditions which is normally used for checking consistency about variables of Verilog and ANSI-C program.

If the result of verification between modified Verilog program and ANSI-C program with additional statements is successful, then the result means that we verified equivalence between FBDs and ANSI-C programs indirectly. On the other side, if the result is fail, then it means FBDs and ANSI-C program operate differently and we can trace the fail condition through analysis of the counterexample which is generated by the HW-CBMC.

4. Conclusion

In this paper, we have introduced verification process of behavioral consistency between design written in FBDs and its implementation written in ANSI-C, which the pSET uses both to develop PLC software. We are planning to modify the translation rules from FBDs to Verilog in order to be appropriate to the HW-CBMC. We are also planning to verify equivalence between Verilog program which is automatically translated under the modified rules from FBDs and ANSI-C program which is automatically generated from the FBDs.

ACKNOWLEDGEMENT

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency (NIPA-2011-(C1090-1131-0008) and (NIPA-2010-C1090-1031-0003). This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2010-0002566). This work was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS(Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"]

REFERENCES

[1] Sengjae Cho, Kyungmo Koo, Byungyong You, Tae-Wook Kim, Taeyoon Shim, Jin S. Lee, Development of the loader software for PLC programming. In: Proceedings of Conference of the Institute of Electronics Engineerers of Korea, Vol. 30, No.1, pp. 595-960, 2007

[2] Korea Nuclear Instrumentation & Control System R&D Center, http://www.knics.re.kr/

[3] RETRANS, Institue for Safety Technology (ISTec), http://www.istec.grs.de/en/produkte/leittechnik/retrans.html?pe_id=54

[4] Edmund Clarke, Daniel Kroening: Hardware verification using ANSI-C programs as a reference. In: Proceedings of the 2003 Asia and South Pacific Design Automation Conference, pp. 308-311, 2003

[5] Eunkyoung Jee, Seungjae Jeon, Sungdeok Cha, Kwangyong Koh, Junbeom Yoo, Geeyong Park and Poonghyun Seong, FBDVerifier: Interactive and Visual Analysis of Counterexample in Formal Verification of Function Block Diagram. In: Journal of Research and Practice in Information Technology, Vol.42, No.3, pp.255-272, 2010