Reliability Engineering and System Safety 120 (2013) 139–149

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

A systematic verification of behavioral consistency between FBD design and ANSI-C implementation using HW-CBMC

Dong-Ah Lee [a], Junbeom Yoo [a,*], Jang-Soo Lee [b]

[a] Division of Computer Science and Engineering, Konkuk University, Republic of Korea
[b] Man-Machine Interface System Team, Korea Atomic Energy Research Institute, Republic of Korea

ARTICLE INFO

Available online 2 July 2013

Keywords:
Verification process
Behavioral consistency
Model checking
FBD
ANSI-C
HW-CBMC

ABSTRACT

Controllers in safety critical systems such as nuclear power plants often use the Function Block Diagram (FBD) to design software embedded in the PLC (Programmable Logic Controller). Software engineers develop FBD programs manually, while engineering tools provided by PLC vendors translate them into ANSI-C programs mechanically. Every new PLC and its software engineering tool should demonstrate the so-called *FBD-to-C* translator's correctness thoroughly. This paper proposes a verification process which can efficiently verify the translator's correctness using the model checking technique. The HW-CBMC model checker verifies the behavioral consistency between FBD and ANSI-C programs formally according to the process and templates which this paper proposes. We also developed a CASE tool '*CWrapper*' and performed a case study with simplified examples of the APR-1400 (Advanced Power Reactor-1400) nuclear reactor protection system in Korea.

1. Introduction

Safety is an important property for real-time embedded systems [1] such as nuclear power plants to obtain permissions for operation and export from government authorities. As the nuclear reactor protection system (RPS) makes decisions for emergent reactor shutdown, RPS software should be verified throughout its entire development life-cycle. RPS software is typically modeled with IEC-61131 FBD (Function Block Diagram) [2] in the design phase, and then in the implementation phase, translated into ANSI-C programs and compiled into an executable machine code for RPS hardware—PLC (programmable Logic Controller). Compiler expert companies typically provide C compilers with a thorough demonstration of functional correctness. On the other hand, PLC vendors usually develop translators which perform FBDs into C programs by themselves. They should demonstrate the translator's correctness and functional safety [3] sufficiently.

In the PLC industry for RPS, vendors such as *AREVA* [4], *invensys* [5] and *POSCO ICT* [6] have provided safety-level PLCs and their own software engineering tool-sets. '*SPACE*' [7] is a software engineering tool-set for *AREVA*'s PLC '*TELEPERM XS*' [8]. It stores FBD programs into a database '*INGRES*' and generates ANSI-C programs to perform code-based testing and simulation ('*TXS SIVAT*' [9]). *ISTec GmbH* [10] also has developed a reverse engineering tool '*RETRANS*' [11] for checking consistency between FBD programs and generated C programs. The mechanical translator in '*SPACE*' has been validated in such ways, and the software engineering tool-sets have been used successfully for more than a decade. PLCs of *invensys* also have been widely used. '*TriStation 1131*' [12] is its software engineering tool-set. It provides enhanced emulation-based testing and real-time simulation of FBDs, but does not include a C translator yet.

KNICS (Korea Nuclear Instrumentation and Control System R&D Center) [13] and *POSCO ICT* in Korea have recently developed a safety-level PLC '*POSAFE-Q*' and its software engineering tool-set '*pSET*' [14]. The tool-set provides a graphical editor for FBD and LD (Ladder Diagram) programming languages [2], and also generates ANSI-C programs automatically. However, sufficient demonstration of correctness and functional safety of the so-called '*FBD-to-C*' translator is still in progress. Thus, it must be one of the most critical obstacles needed to pass inspection in order to obtain permissions for the export of the new Korean nuclear power plant [15] as a whole, *i.e.*, including control software—I&C (Instrumentation & Control).

This paper proposes a systematic way to demonstrate functional correctness of the '*FBD-to-C*' translator using the model checking techniques [16]. We use the '*HW-CBMC*' [17] model checker which can verify the behavioral equivalence between FBD and ANSI-C programs. We first translate a FBD program into a behaviorally equivalent Verilog program based on translation rules in [18]. We modify the rules to translate it into a suitable Verilog program for HW-CBMC, because the Verilog program as an input of HW-CBMC is different from that of the VIS verification

[*] Corresponding author. Tel.: +82 2 450 3258.
E-mail address: jbyoo@konkuk.ac.kr (J. Yoo).

0951-8320/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ress.2013.06.006

system [19]. The next step is to prepare an ANSI-C program which is the other input of HW-CBMC. We provide a '*CWrapper*' program which wraps the ANSI-C program with template-based statements to help users perform the HW-CBMC verification mechanically. The HW-CBMC model checker then verifies the behavioral consistency between the Verilog program translated from FBDs and the wrapped ANSI-C program. This paper uses a part of the FBD programs for ARP-1400 (Advanced Power Reactor-1400) RPS BP (Bistable Processor) to demonstrate feasibility and efficiency of the proposed verification technique.

The remainder of the paper is organized as follows: Section 2 briefly explains the basic elements of the proposed verification techniques, such as FBD, Verilog and HW-CBMC. It also details the typical software development life-cycle of PLC-based systems. The whole verification process is introduced in Section 3. In Section 4, we apply the proposed verification technique to a part of PLC software for APR-1400 RPS BP in Korea. Section 5 overviews related work and we conclude the paper in Section 6.

2. Background

2.1. PLC-based software development process

RPS is a real-time embedded system, implemented on the hardware—PLC. The RPS software is designed in FBD/LD languages and then translated into C programs which will be compiled and loaded on PLCs. Fig. 1 explains a typical software development process for RPS as a waterfall model [20].

SRS (Software Requirements Specification) is written in natural languages or formal specification languages [21–23]. Experts on PLC programming languages then translate the requirements specification into design models programmed in FBD or LD manually. PLC vendors provide their own automatic translators from the FBD/LD programs into ANSI-C programs, while typically using COTS (Commercial Off-the-Shelf) software such as '*TMS320C55x*' of *Texas Instruments* [24] for the C compilers. The COTS compilers were well verified and certified, and sufficient to be used for implementing the RPS software without additional efforts.

The lower part of the figure shows V&V (Verification and Validation) techniques which have been used to demonstrate correctness and functional safety of the '*Automatic Translator*.' '*TXS SIVAT*' [9] from *AREVA*'s *TELEPERM XS* [25] is an example of the C code-based simulation technique, while the '*RETRANS*' [11] is that of the bi-simulation technique. Structural testing techniques with coverage criteria [26] are also applied into the automatically translated C programs. The KNICS project in Korea used a testing tool '*IBM Rational Rhapsody*' [27] for C program testing. The equivalence checking is a verification technique which this paper proposes [28]. It uses a model checker *HW-CBMC* [17], which reads Verilog and ANSI-C programs and checks their behavioral equivalence [29]. It first translates FBD programs into behaviorally equivalent Verilog programs [18]. These various techniques spanning from simulation and testing to formal verification have all been used to guarantee the correct functioning of the PLC vendor-specific '*Automatic Translator*,' *i.e.*, the *FBD-to-C* translator.

2.2. Function Block Diagram

FBD (Function Block Diagram) is one of five standard PLC programming languages defined in the IEC 61131-3 standard [2]. It consists of an arbitrary number of function blocks connected together with wires similar to that of a circuit diagram. FBD has been widely used for developing software controllers of plants and machines because of its graphical notations and usefulness in implementing data flow based applications. For example, the FBD in Fig. 2 consists of 4 function blocks, and the first executed function block is `GE_DINT` while the last one is `SEL_DINT`. `GE_DINT` is the function block calculating logical '≥' with two decimal integer inputs. The whole FBD program is a set of FBDs interconnected with each other according to their predefined sequential execution order.

2.3. Verilog

Verilog is one of the most common HDLs (Hardware Description Languages) used by IC (Integrated Circuit) designers. Designs modeled in Verilog are technology independent, easy to develop and debug, and considered more readable than schematics. For this reason, Verilog is being increasingly used to specify software

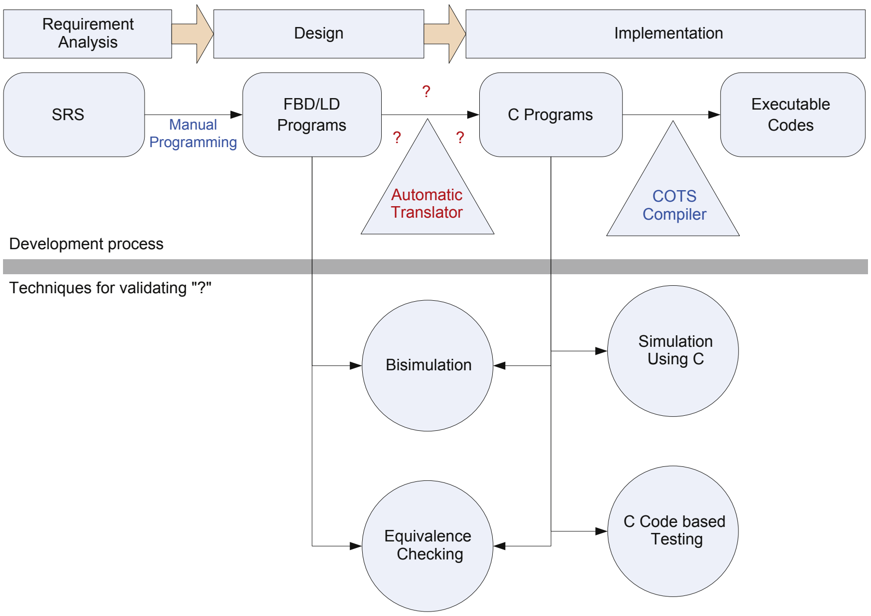

Fig. 1. RPS software development process using PLCs.