Transactions of the Korean Nuclear Society Spring Meeting
Chuncheon, Korea, May 25-26 2006

Development of Application Programming Tool for Safety Grade PLC (POSAFE-Q)

Kyungmo Koo, Byungyong You, Tae-Wook Kim, Sengjae Cho, and Jin S. Lee
Department of Electrical Engineering
Pohang University of Science and Technology, Pohang, 790-784, Korea
{pumpkins, ybyvic, tw0822, sjcho80, jsoo}@postech.ac.kr

1. Introduction

The *pSET* (POSAFE-Q Software Engineering Tool) is an application programming tool of the POSAFE-Q which is a safety graded programmable logic controller (PLC) developed for the reactor protect system of the nuclear power plant. The pSET provides an integrated development environment (IDE) which includes editors, compiler, simulator, downloader, debugger, and monitor. The pSET supports the IEC61131-3 standard software model and languages such as LD (ladder diagram) and FBD (function block diagram) [1] which are two of the most widely used PLC programming languages in industry fields. The pSET will also support SFC (sequential function chart) language.

The pSET is developed as a part of a Korea Nuclear Instrumentation & Control System (KNICS) project.

2. Development of pSET Software

The pSET software works on a PC with Windows 2000 or XP operating system and connected to POSAFE-Q via RS-232C as shown in Fig. 1. Using pSET software, user can develop application programs for POSAFE-Q through a series of editing, compiling, simulation, downloading, debugging and monitoring.

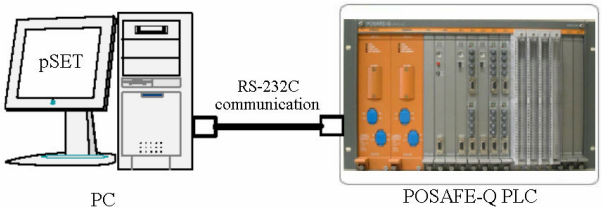

Figure 1. The pSET is an application programming tool of the POSAFE-Q.

2.1 Software feature

The pSET provides an IDE which includes several tools for development of application programs such as editors, compilers, simulator, code downloader, monitor, and simulator. The graphic user interface (GUI) of pSET is shown in Fig. 2. The pSET supports IEC61131-3 standard software model which has layered hierarchy of *configuration*, *resource*, and *programs* with *tasks* [1]. This hierarchy structure can be edited easily in the work space which is on the left side in the pSET IDE. *Access path* which provides facilities for communicating data and information with external of configuration [1] is also supported.

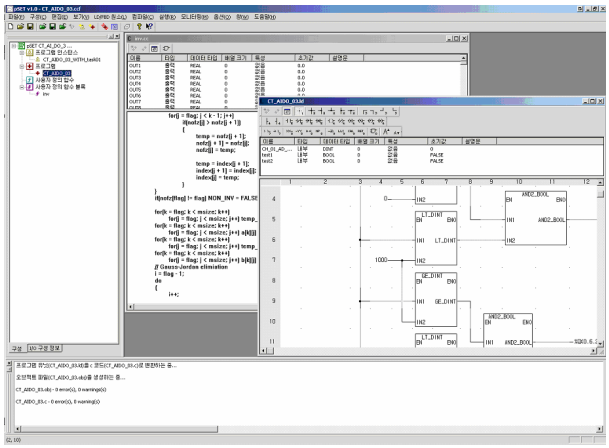

Figure 2. Graphic user interface of pSET software

User can edit not only application programs, but also I/O and task information. User also can compile, simulate, download, debug, and monitor application programs in a single IDE. In a debug mode, it is possible to insert/remove break points rung by rung at run-time and execute in a single-step. In a monitoring mode, states of variables are displayed not only in an output window (as a tabular form), but also in a program editor. Variables are possible to be forced to a specific state or reset. Several functionalities of variable editor such as *import/export variable list* or *clean up unused variables* are useful to handle huge number of variables in a large-size application program. Moreover, the *pSETSIM*, pSET simulator, can be used to simulate application programs on a PC before they are downloaded to POSAFE-Q.

Any user-specific programs can be packed in modules in a form of user-defined function block (UDF). It helps application programs to be more abstract, to be reusable, and to be more compact by eliminating repeated codes. Moreover, UDF can be written not only by LD/FBD languages but also by C language. It makes C programs which are already working in existing systems to be imported in new system of POSAFE-Q with easy. The number of I/O and the program size of UDF are not limited as far as the hardware resources are available.

Other detailed features can be found in pSET Software Requirement Specification (SRS) [2]. Every features described in the SRS are considered and implemented in the current version of pSET software.

2.2 Language Conversion

Since POSAFE-Q executes compiled machine code of application programs directly, it is much faster than interpreter-type PLC. Because a single application code can make the whole PLC system crash, however, the binary code must be generated by a reliable compiler. Hence, it is important to develop a highly-reliable compiler. And, the compiler should support a way to validate that the generated code is functionally safe and logically equivalent to the original application program written in IEC61131-3 languages.

To cope with this problem, we construct LD/FBD and SFC compilers to convert application programs to ANSI-C code, not binary machine code. If application programs are converted to C code correctly, we can generate a reliable machine code through commercial C compiler such as *gcc* or other CPU vender-supplied C compiler. Moreover, since the generated C code can be verified by engineers or commercial C code-verifying tools, it can be used in verification and validation (V&V) procedure for the development of safety-critical application programs.

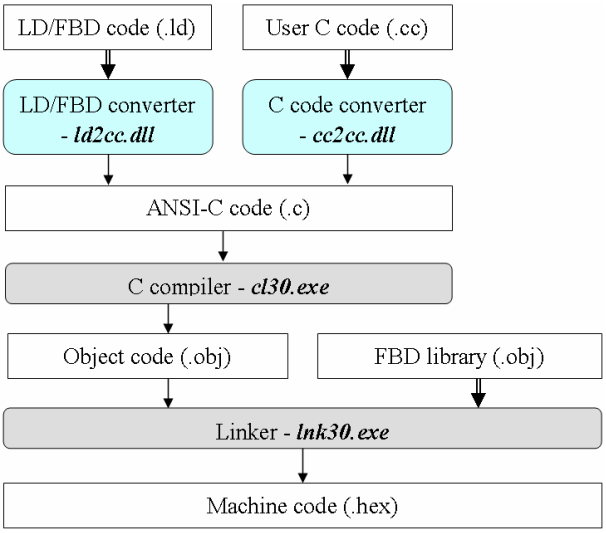

Figure 3. Compile procedure of user application programs

The procedure to make machine code from user application program is described in Fig. 3. Each user program or UDF written in LD/FBD language is converted to ANSI-C format by the *ld2cc* (LD/FBD to C converter). And, each UDF written in C-code is transformed to ANSI-C format by *cc2cc* (C-code to C converter). Converted C files are compiled to object code by *cl30.exe* which is a C compiler for C32 DSP of Texas Instruments. Every standard function blocks defined in IEC61131-3 are implemented and provided as vendor-provided FBD library. When application programs include vendor-provided FBD, corresponding FBD libraries are linked to the compiled object codes. Then, finally, the machine code is generated.

2.3 Safety and Security Problems

Most of GUI-related parts of the pSET such as editor, simulator, and monitor are not safe-critical, but compiler and code loader are safety-related parts which are designed and developed with consideration of NUREG-CR6463 guidelines [3]. As described earlier, the pSET compilers convert application programs to ANSI-C codes, and extensive tests have been conducted in not only the machine code level but also the converted C code level.

Security is also an important issue in safety-critical programs. Each operation of PLC must be allowed only to a person with valid authentication. The pSET specifies the following levels of authentication.

- Create/edit resource and programs (highest)
- Compile programs
- Load machine codes
- Run/stop/monitor machine codes (lowest)

Every operation is recorded in the log file with information of operating time, operator, and operator's comments.

3. Conclusion

We developed pSET, an application programming tool of the POSAFE-Q, as a part of KNICS project. The pSET is an IDE including editors, compiler, simulator, downloader, debugger, and monitor. The pSET supports IEC61131-3 standard software model and programming languages such as LD/FBD and SFC. Compilers of the pSET convert application programs to ANSI-C codes, which allow to be used in V&V for safety-critical applications. Moreover, it is also possible to generate reliable machine code with well-established commercial C compiler.

REFERENCES

[1] IEC Standard 61131-3: Programmable controllers-Part 3, IEC61131, 1993.
[2] KNICS-PLC-SRS: pSET Software Requirement Specification, KERI/POSCON, 2005.
[3] NUREG/CR-6463: Review guidelines on Software Languages for Use in Nuclear Power Plant Safety Systems.