# Automated PLC software testing for Reactor Core Protection System Interface and Test Processor using the execution control method for test sequences

Hyeongseok Eun [a,b*], Lingjun Liu [b], Eunkyoung Jee [b], Doo-Hwan Bae [b], Changjae Lee [a], Yoonhee Lee [a]
[a]*KEPCO E&C Company Inc., 989-111 Daedeokdaero, Yuseong-gu, Daejeon, 34057, Republic of Korea*
[b]*School of Computing, Korea Advanced Institute of Science and Technology (KAIST)*
[*]*Author: ehs@kepco-enc.com*

## 1. Introduction

Programmable logic controllers (PLCs) are used to implement safety-critical systems in nuclear power plants (NPPs). The importance of PLC testing is increasing, and accordingly, several studies on PLC testing have been conducted [1-9]. The main characteristic of a PLC is that the program is executed periodically and continuously. This characteristic makes the precise testing of PLC programs difficult. To overcome this difficulty and precisely test the PLC program per cycle, it is necessary to inject the input and confirm the output of the task sequence per cycle. In previous studies [1,2], it was possible to inject the input and confirm the output of the task sequence per cycle; however, this was not applied to real PLC software. Therefore, in this study, real PLC software testing was conducted using the reactor core protection system (RCOPS) PLC software.

RCOPS generates a trip signal when the calculated departure from the nucleate boiling ratio (DNBR) or local power density (LPD) exceeds trip setpoints. The RCOPS is composed of a core protection processor (COPP) that contains the main functions of the RCOPS, control element assembly processor (CEAP), channel communication processors (CCPs), and interface and test processor (ITP). RCOPS ITP transmits the outputs to the Qualified Indication and Alarm System – Non-Safety (QIAS-N).

In this study, we show that the previously proposed method [1,2] can be applied to the RCOPS ITP software to reduce the testing time.

## 2. Methods and Results

### *2.1 RCOPS ITP PLC existing testing method*

The RCOPS ITP receives process variables from the COPP and sends them to the QIAS-N. Some units of the RCOPS variables are different from those of QIAS-N; hence, unit conversion logic is implemented in the RCOPS ITP. It also generates signal quality information and transmits it to the QIAS-N. RCOPS ITP PLC testing is performed to check the unit conversion logic and the generation logic of the signal quality information, as shown in Figure 1.

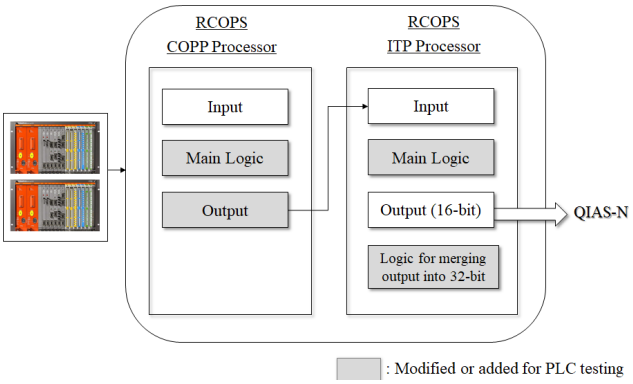


Figure 1. RCOPS ITP PLC existing testing method

The existing RCOPS ITP PLC testing method is described as following steps:

1) The ITP software is modified to bypass the logic for communication verification, which checks the COPP heartbeat. Merging logic is added to merge two 16-bit split integer values for QIAS-N into one 32-bit value.
2) The COPP software is modified to block the main logic to bypass continuous output changes per cycle.
3) Each COPP output for the ITP is manually changed by the tester according to the test case. The changed outputs are transmitted to the ITP.
4) After the main logic runs, the tester confirms that the final 32-bit outputs match the expected outputs in the ITP software through the PLC monitoring program.
5) To change the test inputs, repeat steps 3 and 4.

The communication verification logic that compares the current heartbeat with the previous heartbeat to verify the communication cannot be passed because the heartbeat is fixed in the COPP. The result of the verification is added to the signal quality information; therefore, the verification function is blocked to simulate a normal status.

In addition, the variables transmitted to QIAS-N are 16-bit split integer values owing to the communication protocol. As such, the final outputs cannot be intuitively understood through PLC monitoring. Therefore, the logic to merge two 16-bit split integer values for QIAS-N into one 32-bit process value was added for confirmation.

The existing PLC software testing has the following disadvantages: considerable logic in the COPP and ITP must be modified, and each COPP output for the ITP must be manually changed to inject input. The main reason of this difficulty is that I/O simulator is expensive and hard to modify the high reliability-safety data network (HR-SDN) communication between the COPP and ITP. Second, it is not possible to systematically control the user function block (UFB) and standard function block (SFB) to inject test sequences per cycle in the PLC program with the existing testing method.

### *2.2 RCOPS ITP PLC testing with the EC method*

In this study, we used a test driver and test stub with the execution control (EC) method [1,2] that can conduct the automated PLC testing without changing the main logic and without additional equipment such as COPP or I/O simulator. RCOPS ITP PLC testing using the EC method is performed as shown in Figures 2 and 3. All inputs for the test sequences can be injected by adding a test driver and test stub in front of the main logic.

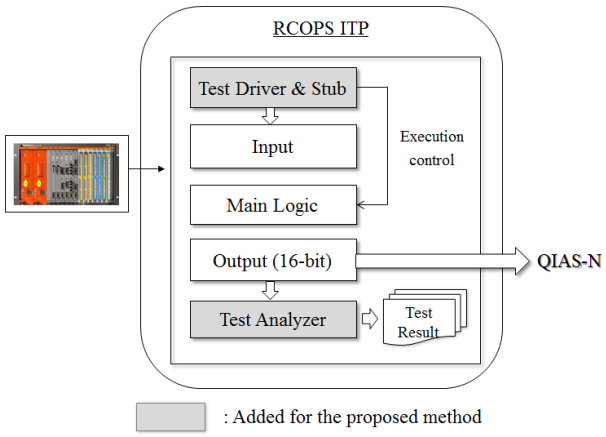


Figure 2. RCOPS ITP PLC testing with the EC method

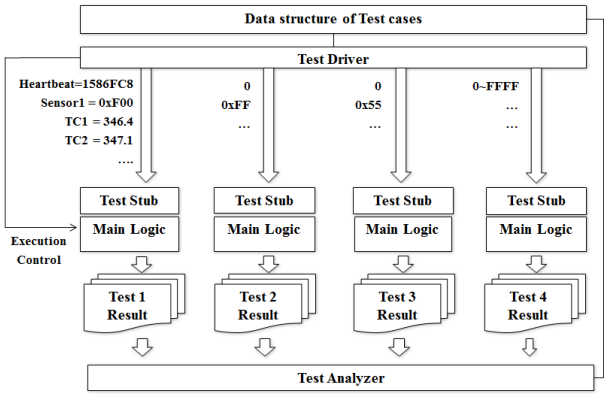


Figure 3. Test driver, test stub, and test analyzer for the EC method

The EC method is described as following steps:

1) The test driver and test stub are added in front of the main logic, and a test analyzer is added at the back of the main logic.
2) Test sequences are automatically loaded onto the test driver. If necessary, some function blocks are controlled by the test driver.
3) After the main logic runs, the tester confirms the match flag and all results of the test analyzer through the PLC monitoring program.
4) If required, test cases can be selected for the test driver by the tester.

The test inputs and expected outputs for the test case are precompiled as the data structure. The test driver selects the test inputs from the data structure, and the test stub injects these inputs in front of the main logic.

The test driver has an automated testing option so that a tester can select manual or automated testing. When automated testing is selected by the tester, the test cases are injected and executed per cycle, and they are called “test sequences” instead of “test cases”.

The EC method is used to prevent communication and block communication verification. The test driver prevents the execution of the communication function block and manipulates the communication verification results. The test driver can also automatically increase the heartbeat value.

When the main logic is executed with the injected inputs, the results are displayed and compared with the expected outputs by the test analyzer. The test analyzer checks whether all the outputs match the expected results and generates a match flag that indicates whether all the output values are matched. It also merges all 16-bit split outputs into 32-bit outputs to display them. Therefore, the tester can evaluate the result using the automatically generated match flag or manually check all results.

Compared to the existing RCOPS ITP testing, which requires many modifications of the main logic of the ITP and COPP software, the EC method has an advantage in that it does not require the main logic to be modified. It is possible to perform all test cases of the existing testing by adding test functions at the front and at the back of the main logic. If the PLC software is revised, testing can be performed quickly because the main logic does not require modification. Furthermore, the COPP is not used anymore in the EC method because it does not need to manually change each COPP output to inject the test input.

### *2.3 RCOPS ITP PLC testing results with the EC method*

The three original test cases for the Shin-Kori NPP units 5 and 6 did not include the dynamic heartbeat change.

Using EC method, the test case for the heartbeat can be added to verify the logic for the communication

verification, which checks the COPP heartbeat. This test case and the three original test cases for Shin-Kori NPP units 5 and 6 were performed, as shown in Figure 4 and Table I.

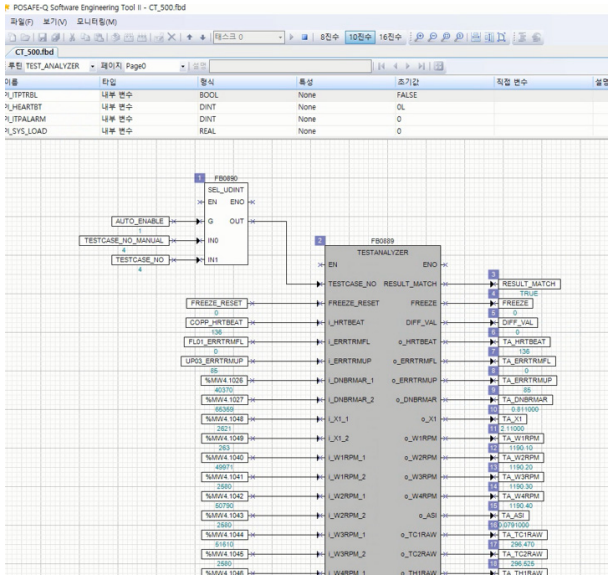


Figure 4. Test analyzer for the EC method with PLC monitoring

Table I: Test results with the EC method

| Test case*1 | #1 | #2 | #3 | #4 |
|---|---|---|---|---|
| **Modified Inputs** | 16 | 3 | 3 | 1 |
| **Expected Outputs** | 16 | 1 | 1 | 1 |
| **Matched Outputs** | 16 | 1 | 1 | 1 |
| **Match Rate (%)*2** | 100% | 100% | 100% | 100% |

*1: Test case #1: For a unit conversion
Test case #2: For a normal signal quality
Test case #3: For an abnormal signal quality
Test case #4: For a dynamic heartbeat change

*2: $\frac{Matched\ outputs}{Expected\ outputs} \times 100\%$

All test cases were automatically executed as test sequences, and the testing execution time was calculated as follows:

- *Testing execution time = PLC scan time (500 ms) × Number of test sequences (4) = 2 s*

Assuming that it takes at least one minute for one test case replacement using the existing method, and it takes approximately 20 s for PLC monitoring confirmation for both methods, the testing time ratio of the existing method to the EC method is as follows:

- $\frac{Time\ of\ the\ EC\ method}{Time\ of\ the\ existing\ method} = \frac{2+20\ s}{4*60+20\ s} \times 100\% = 8.5\%$

Compared to the existing testing method, 91.5% of the testing time was reduced by the proposed method. Therefore, it is shown that the manual effort and testing time can be significantly reduced using the EC method.

## 3. Conclusions

We demonstrated that all test cases can be executed using the EC method without any modification of the main logic of the COPP and ITP; 91.5% of the testing time was reduced. We expect that there will be a dramatic reduction if the EC method is applied to other PLC software.

The test cases cannot be changed during real-time execution because the test inputs and expected outputs for the test case are precompiled. Therefore, methods to efficiently inject test cases and confirm test results in real time should be considered in future work.

## ACKNOWLEDGMENTS

This work was supported by the Nuclear Safety Research Program through the Korea Foundation Of Nuclear Safety (KoFONS) using the financial resource granted by the Nuclear Safety and Security Commission (NSSC) of the Republic of Korea. (No. 2105030)

## REFERENCES

[1] H. Eun, L. Liu, E. Jee, and D. H. Bae, “A systematic execution and output method of test sequences for PLC program,” Proceedings of the Korean Information Science Society Conference, pp. 245-247, 2021. (in Korean)
[2] H. Eun, L. Liu, E. Jee, and D. H. Bae, “An execution control method of test sequences per cycle for multi-task PLC programs,” Proceedings of the 24th Korea Conference on Software Engineering, pp. 137-140, 2021. (in Korean)
[3] L. Liu, E. Jee, and D. H. Bae, "MuFBDTester: A mutation-based test sequence generator for FBD programs implementing nuclear power plant software," Software Testing, Verification and Reliability, 2022. (to appear)
[4] J. Song, E. Jee, and D. H. Bae, "FBDTester 2.0: Automated test sequence generation for FBD programs with internal memory states," Science of Computer Programming, Vol. 163, pp. 115-137, Oct. 2018.
[5] E. Jee, D. Shin, S. Cha, J. S. Lee, and D. H. Bae, "Automated test case generation for FBD programs implementing reactor protection system software," Software Testing, Verification and Reliability, Vol. 24, No. 8, pp. 608-628, Sep. 2014.
[6] E. Jee, J. Song, L. Liu, and D. H. Bae, "Evolution case analysis of testing techniques for FBD programs implementing safety-critical systems," Communications of the Korean Institute of Information Scientists and Engineers, Vol. 39, No. 10, pp. 30-40, Oct. 2021. (in Korean)
[7] S. Jang, B. Choi, and A. Sung, “A Test Method for PLC Input and Output Devices Using Function Block Diagrams," Proceedings of Korea computer congress, Vol. 35, No. 1, Jun. 2008. (in Korean)
[8] J. Sim, C, Kwon, and K Han, “Model based PLC instruction Verification Method,” In Proceedings of 2011 Conference on Information and Control Systems (CICS 2011), 2011. (in Korean)