State Transition Modeling of Automatic Periodic Surveillance Test Logic for Plant Protection System

Tae Young Jhee [a], Tae Ryoun Kim [a], Soonae Lee [b], Jonghyun Kim [a*]

[a] Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology, 291 Daehak-ro, Yuseong-gu, Daejeon 34141, Republic of Korea

[b] SMR Instrumentation and Electrical Design Group, Central Research Institute, Korea Hydro and Nuclear Power (KHNP), Daejeon, Republic of Korea

E-mail: docotrjhee@kaist.ac.kr, bkteddy00@kaist.ac.kr, soonaelee@khnp.co.kr, jonghyun.kim@kaist.ac.kr

Abstract: Periodic surveillance testing (PST) is performed to verify that nuclear power plant protection systems continue to perform their intended safety functions as designed. In many nuclear power plants, PST remains largely procedure-driven and manually executed. The multi-channel architecture of the Plant Protection System (PPS) requires repeated channel-level verification of protection logic functions while maintaining the required level of protection, thereby increasing the procedural and operational burden. Moreover, manual channel-by-channel testing can prolong the time during which individual channels are placed in a test or bypass condition, potentially reducing overall protection system availability. Several automatic testing approaches have been proposed; however, existing implementations are generally limited to externally triggered execution or operator-assisted execution, and the expected benefits of automation—including shorter test duration, reduced operator involvement, and improved protection system availability—have not been fully achieved. These limitations motivate further automation of PST procedures, which can be achieved by integrating the test sequence directly within the field-programmable gate array (FPGA)-based logic processor. This study proposes a built-in automatic PST framework for two types of variable-setpoint bistable logic that are currently tested manually: one with a rate-limited variable setpoint and the other with a manual-reset variable setpoint. A common test architecture is adopted in which the built-in test logic generates a test stimulus, applies it to the target protection logic, captures the test output, compares it with the expected result, and represents the test-status reporting process within the state-transition model. To ensure functional isolation, the test stimuli and resulting test outputs are tagged with a test identifier, so that the tagged output is confined within the processor test logic and is not propagated to the downstream protection or coincidence logic. For each test item, a dedicated state-transition model is defined and implemented in MATLAB/Simulink using Stateflow. State and transition coverage analyses confirmed that all defined states and transitions were exercised during simulation. The results demonstrate that the proposed automatic PST framework can be applied to selected manually performed PST items for RPS bistable logic with variable setpoints. This study is also expected to provide a structured basis for implementing scalable built-in automatic PST functions in FPGA-based

Keywords: Built-in Automatic Testing, State transition diagram, Plant Protection System (PPS)

1. INTRODUCTION

In nuclear power plants (NPPs), the Plant Protection System (PPS) provides safety-related protection functions by initiating reactor trips and other protective actions when monitored process variables exceed or fall below predefined trip setpoints or when specified trip logic conditions are satisfied. Periodic surveillance testing (PST) is performed to verify that the PPS and its associated subsystems remain capable of carrying out their design-basis functions when demanded [1, 2]. In conventional NPPs, operators or maintenance personnel manually execute these tests according to predefined procedures.

Manual PST is particularly burdensome in protection systems with multiple redundant channels because each channel must be tested. This requires successive channel-level testing of protection logic functions and increases the procedural and operational workload [3]. In addition, channel-by-channel testing can

reduce overall protection system availability [4]. Therefore, the development of automated PST methods is necessary.

Digital instrumentation and control (I&C) systems important to safety have advanced toward FPGA-based logic implementations. In FPGA-based protection systems, protection functions are implemented as deterministic hardware logic rather than as software executed on PLC-based platforms [5, 6]. This implementation approach reduces dependence on operating systems and complex runtime software. The practical applicability of FPGA-based safety I&C has also been demonstrated in safety-related functions, including isolation functions and shutdown system implementations [7, 8].

Automatic testing approaches have already been introduced in PLC-based digital protection systems. For example, the IDiPS-RPS developed under the KNICS program incorporated automatic test features using the ATIP to support periodic verification of protection logic during normal operation [9]. Jeon et al. also proposed an automatic testing system for PLC-based reactor protection systems using a dedicated signal processing unit for test stimulus injection and response acquisition [10]. However, these approaches still depend partly on procedure-based test execution. Test initiation remain operator-dependent, and test stimulus injection is not yet fully integrated into the protection logic processor. In addition, these approaches generally require protection channels to be placed in bypass during test execution, reducing system availability. Consequently, the expected benefits of automation, including reduced test duration, reduced operator workload, and improved protection system availability, have not been fully realized.

These limitations motivate processor-integrated surveillance testing, in which the test sequence is implemented as dedicated test logic within the FPGA-based logic processor. In this approach, test outputs are captured within the processor-integrated test logic and prevented from propagating to downstream protection logic, thereby maintaining functional isolation from the protection signal path. A previous study proposed a built-in automatic testing method for FPGA-based digital protection logic and demonstrated its feasibility using fixed-setpoint bistable logic as an initial case study [11].

This study extends the built-in automatic PST methodology from fixed-setpoint bistable logic to variable-setpoint bistable logic. Two representative protection logic functions are considered: bistable logic with a rate-limited variable setpoint and bistable logic with a manual-reset variable setpoint. Unlike fixed-setpoint logic, these functions require test sequences that account for setpoint variations associated with rate-limit constraints or manual-reset operations. For each logic function, the built-in automatic test sequence is modeled as a state-transition diagram and implemented in MATLAB/Simulink using Stateflow. State and transition coverage analyses confirmed that all defined states and transitions were exercised during simulation. The proposed approach demonstrates its applicability to the development of built-in automatic PST functions for FPGA-based digital protection systems.

2. BACKGROUND

2.1 Plant Protection System and Periodic Surveillance Testing

In nuclear power plants, the Plant Protection System (PPS) monitors safety-related plant variables and generates protective actuation signals when predefined protection logic conditions are satisfied. In the APR1400 design, the PPS consists of the Reactor Protection System (RPS) and the Engineered Safety Features Actuation System (ESFAS), which generate reactor trip signals and engineered safety feature actuation signals, respectively. The PPS implements these functions using bistable logic, coincidence logic, initiation logic, and maintenance/test logic. Bistable logic compares measured process variables with trip setpoints and generates channel-level trip signals. Coincidence logic evaluates trip signals from redundant channels, such as through two-out-of-four voting in the APR1400 RPS, before issuing protective actuation signals. Periodic surveillance testing (PST) is performed to verify that these protection logic functions and associated signal paths remain capable of performing their intended safety functions.

2.2 State-Transition Diagram

A state-transition diagram is an abstract graphical model used to represent finite-state-machine (FSM) behavior for sequential logic. In FPGA-based logic design, this representation can be synthesized into hardware logic and implemented on the FPGA fabric. In its conventional form, a state-transition diagram is expressed as a state-transition graph composed of states, transition conditions, and state outputs. Each state is represented as a node, typically shown as a circle, while directed arcs indicate transitions between states.

Each transition arc is labeled with the transition condition that enables the corresponding change from one state to another. The output associated with each state is specified within the state node. If a transition arc is presented without an explicit condition, it is interpreted as an unconditional transition, meaning that the system proceeds to the next state regardless of the input values.

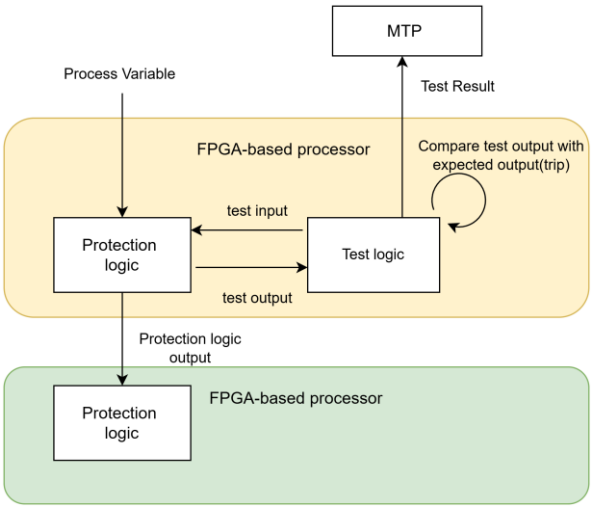
To ensure correct and deterministic operation, the transition conditions for each state must be complete and mutually exclusive. Completeness means that every possible combination of input conditions leads to a defined next state. Mutual exclusiveness means that the same input condition must not enable more than one outgoing transition from the same state, since this would result in ambiguous state-transition behavior.

2.3 Built-in Automatic Testing Method

The built-in automatic testing method is a periodic surveillance testing approach in which the processor-integrated test logic automatically generates test stimuli at predetermined test intervals. Depending on the target protection function, the generated test stimulus may represent either an abnormal trip condition or a setpoint exceedance condition derived from the initial setpoint and the current process variable value. These test stimuli are applied to the FPGA-based logic processor to periodically evaluate whether the protection logic performs its intended function.

As illustrated in Figure 1, the FPGA-based processor includes built-in test logic that generates the test stimulus, applies it to the protection logic, and captures the corresponding output for comparison with the expected response. To maintain functional independence from the safety function, both the test stimulus and the resulting output are assigned a test identifier. The output carrying this identifier is collected only by the processor-integrated test logic for pass/fail determination and is not transmitted to the downstream protection or actuation logic responsible for executing the safety function. Through this architecture, the built-in test function is functionally separated from the trip execution path, thereby preventing inadvertent actuation of safety functions during automatic testing.

Fig. 1 Concept of Built-in Automatic Testing



3. STATE CHART MODELING OF BUILT-IN AUTOMATIC PST LOGIC

3.1 State Chart Modeling of Rate Limited Variable Setpoint Bistable Logic

3.1.1 Automatic Test logic for Rate Limited Variable Setpoint Bistable Logic

A rate-limited variable setpoint is designed to track changes in the process variable while maintaining a predefined trip margin under normal operating conditions. When the process variable changes within the allowable tracking rate, the variable setpoint follows the process variable without causing a trip. However, if the process variable changes faster than the maximum allowable rate of change of the variable setpoint, the separation between the process variable and the trip setpoint decreases. As a result, the process variable may reach the trip setpoint, causing the bistable logic to generate a trip signal.

The built-in automatic test for bistable logic with a rate-limited variable setpoint is intended to verify the functional integrity of bistable logic applied to process variables governed by rate-limited variable setpoints, such as Variable Overpower Trip (VOPT) functions in the APR1400 PPS [12]. For this test, the built-in test logic generates a test stimulus that exceeds the maximum allowable setpoint change calculated using the current setpoint and the current process variable value, thereby representing an abnormal trip condition. The generated stimulus is intended to simulate an abnormal trip condition. The test response is then evaluated by verifying the generation of the expected bistable output and confirming the consistency between the observed output and the expected trip decision for the applied stimulus. The proposed built-in automatic periodic testing method is performed according to the following procedure:

(1) The built-in test logic generates a test stimulus representing an abnormal trip condition based on the current setpoint, the current process variable value, and the maximum allowable setpoint change. The test input is generated as follows:

$$\boldsymbol{X}_{\text{test}} = \boldsymbol{X}_{\text{PV}} + \boldsymbol{M}_{\text{trip}} \times \boldsymbol{K}_{\text{exc}} \qquad (1)$$

where $\boldsymbol{X}_{\text{test}}$ is the generated test input, $\boldsymbol{X}_{\text{PV}}$ is the current process variable value, $\boldsymbol{M}_{\text{trip}}$ is the trip margin between the current process variable and the trip setpoint, and $\boldsymbol{K}_{\text{exc}}$ is the exceedance factor.

(2) Apply the generated stimulus to the bistable logic.
(3) Produce the corresponding bistable output (trip/non-trip) in response to the applied stimulus.
(4) The processor test logic acquires the bistable output from the bistable logic.
(5) Determine the test result (pass/fail) by comparing the acquired output with the expected trip decision for the applied stimulus.
(6) Transfer the resulting test status to the MTP (Maintenance and Test Panel). The tagged test output is confined to the test logic and is not propagated to the coincidence logic.

3.1.2 State-Transition Diagram of the Built-in Automatic Testing Method for Bistable Logic with a Rate-Limited Variable Setpoint

Figure 2 shows the state-transition diagram of the built-in automatic testing method for bistable logic with a rate-limited variable setpoint. As shown in Table 1, The proposed logic consists of seven states: WAIT, LATCH_DATA, APPLY_TEST, CAPTURE_OUT, PASS, FAIL, and SEND. These states are categorized into two main functional groups: test stimulus generation, which includes LATCH_DATA and APPLY_TEST, and output verification and result transmission, which includes CAPTURE_OUT, PASS, FAIL, and SEND.

Table 1. State Descriptions of the Rate Limited Variable Setpoint Test Logic

State	Description
WAIT	Waits until the predefined periodic test interval is reached.
LATCH_DATA	Retrieves the rate limited variable setpoint data required for the test input generation
APPLY_TEST	Generates and applies a test stimulus representing an abnormal trip condition
CAPTURE_OUT	Captures and validates the bistable output corresponding to the applied test stimulus
PASS	Indicates that the captured test output matches the expected trip decision.
FAIL	Indicates that the captured test output does not match the expected trip decision.
SEND	Transmits the test result to the MTP after the required hold duration.

In the WAIT state, the logic remains in a standby mode until the predefined periodic test interval is reached, expressed as Time ≥ Test Period. Once this condition is satisfied, a new automatic test sequence is initiated.

In the LATCH_DATA state, the rate-limited variable-setpoint data required for the current test cycle are retrieved. After the required data, including the current trip setpoint and current process variable value for the selected trip function such as Variable Overpower Trip (VOPT), are loaded, the logic proceeds to the APPLY_TEST state for test input generation and application.

In the APPLY_TEST state, the built-in test logic generates a test stimulus representing an abnormal trip condition based on the current process variable value, the trip margin, and the exceedance factor. For the high-trip function considered in this case study, the test input is calculated as follows:

$$\boldsymbol{X}_{\text{test}} = \boldsymbol{X}_{\text{PV}} + \boldsymbol{M}_{\text{trip}} \times \boldsymbol{K}_{\text{exc}} \qquad (1)$$

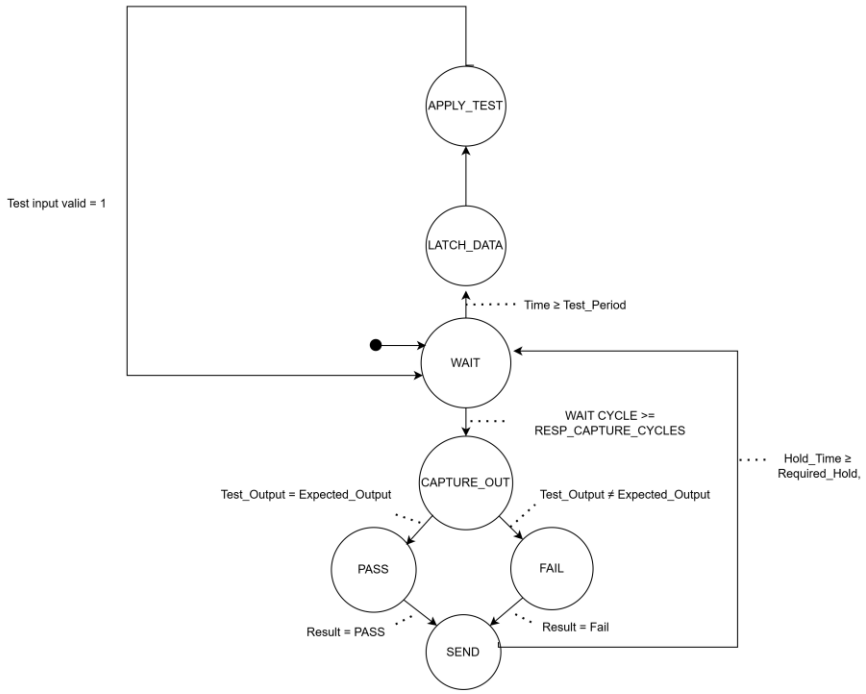
where $\boldsymbol{X}_{\text{test}}$ is the generated test input, $\boldsymbol{X}_{\text{PV}}$ is the current process variable value, $\boldsymbol{M}_{\text{trip}}$ is the trip margin between the current process variable and the trip setpoint, and $\boldsymbol{K}_{\text{exc}}$ is the exceedance factor.

The generated stimulus is then applied to the bistable logic by setting the test input valid signal to 1.
In the CAPTURE_OUT state, the bistable output corresponding to the applied test stimulus is captured and validated. The transition condition, Test_Output_ID = Selected_PV and Test_Output_ID ≠ Previous_Test_Output_ID, ensures that the acquired output corresponds to the currently selected process variable and represents a newly generated response rather than a previously latched output.

The logic transitions to the PASS state when the captured output matches the expected trip decision, expressed as Test_Output = Expected_Output. Conversely, if the captured output does not match the expected output, expressed as Test_Output ≠ Expected_Output, the logic transitions to the FAIL state.

In the SEND state, the final test result, either PASS or FAIL, is transmitted to the Maintenance and Test Panel (MTP) after being maintained for the required hold duration, expressed as Hold_Time ≥ Required_Hold. After the result transmission is completed, the logic returns to the WAIT state and remains ready for the next periodic test cycle.

Fig. 2 State Transition Diagram of the Built-in Automatic Testing Method for Bistable Logic with a Rate-Limited Variable Setpoint



3.2 State Chart Modeling of Manual Reset Variable Setpoint Bistable Logic

3.2.1 Manual Reset Variable Setpoint Bistable Logic

A manual-reset variable setpoint is a type of variable setpoint in which the trip setpoint is reduced by a predefined step in response to each operator reset action using the manual reset switch. By repeatedly resetting the setpoint when a pretrip condition is reached, the operator can perform a controlled plant cooldown without unnecessarily initiating a reactor trip or other protective actions. When the input signal increases after the most recent reset, the variable setpoint increases accordingly to maintain a predefined margin relative to the input signal.

The built-in automatic test for bistable logic with a manual-reset variable setpoint is intended to verify the functional integrity of bistable logic applied to process variables governed by manual-reset variable setpoints, such as the Low Pressurizer Pressure Trip (LPPT) and Low Steam Generator Pressure Trip functions in the APR1400 PPS [11]. For this test, the built-in test logic generates a test stimulus that crosses the current manual-reset variable trip setpoint in the trip direction, thereby representing an abnormal trip condition. The test stimulus is generated based on the current trip setpoint, the current process variable value, and the predefined trip margin associated with the manual-reset variable setpoint. The test response is then evaluated by verifying the generation of the expected bistable output and confirming the consistency between the observed output and the expected trip decision for the applied stimulus. The proposed built-in automatic periodic testing method is performed according to the following procedure:

(1) The built-in test logic generates a test stimulus representing an abnormal trip condition based on the current process variable value, the current setpoint, and the exceedance factor. The test input is generated as follows:

$$X_{\text{test}} = X_{\text{PV}} - (X_{\text{PV}} - SP_0) \times K_{\text{exc}} \qquad (2)$$

where X_{test} is the generated test input, X_{PV} is the current process variable value, SP_0 is the current setpoint, and K_{exc} is the exceedance factor.

(2) The generated test stimulus is applied to the bistable logic.
(3) The bistable logic produces the corresponding output, either trip or non-trip, in response to the applied stimulus.
(4) The processor test logic acquires the bistable output from the bistable logic.
(5) The test result is determined as pass or fail by comparing the acquired output with the expected trip decision for the applied stimulus.
(6) The resulting test status is transferred to the Maintenance and Test Panel (MTP). The tagged test output is confined within the test logic and is not propagated to the downstream coincidence logic.

3.2.2 State-Transition Diagram of the Built-in Automatic Testing Method for Bistable Logic with Manual Reset Variable Setpoint Bistable Logic

Figure 3 shows the state-transition diagram of the built-in automatic testing method for bistable logic with a manual-reset variable setpoint. As shown in Table 2, The proposed logic consists of eight states: WAIT, LATCH_DATA, SELECT_PV, APPLY_TEST, CAPTURE_OUT, PASS, FAIL, and SEND. These states are categorized into two main functional groups: test stimulus generation, which includes LATCH_DATA, SELECT_PV, and APPLY_TEST, and output verification and result transmission, which includes CAPTURE_OUT, PASS, FAIL, and SEND.

Table 2. State Descriptions of the Manual Reset Variable Setpoint Test Logic

State	Description
WAIT	Waits until the predefined periodic test interval is reached.
LATCH_DATA	Retrieves the Manual Reset variable setpoint data required for the test
SELECT_PV	Selects the process variable to be tested and latches the corresponding data for test-input generation.
APPLY_TEST	Generates and applies a test stimulus representing an abnormal trip condition
CAPTURE_OUT	Captures and validates the bistable output corresponding to the applied test stimulus
PASS	Indicates that the captured test output matches the expected trip decision.
FAIL	Indicates that the captured test output does not match the expected trip decision.
SEND	Transmits the test result to the MTP after the required hold duration.

In the WAIT state, the logic remains in a standby mode until the predefined periodic test interval is reached, expressed as Time ≥ Test Period. Once this condition is satisfied, a new automatic test sequence is initiated.

In the LATCH_DATA state, all manual-reset variable-setpoint data required for the current test cycle are retrieved, including the trip setpoints and process variable values for the applicable trip functions. After the required data are loaded, the logic proceeds to the SELECT_PV state.

In the SELECT_PV state, the process variable (PV) to be tested is selected, and the corresponding data is latched. The logic then proceeds to the APPLY_TEST state for test input generation and application.

In the APPLY_TEST state, the built-in test logic generates a test stimulus representing an abnormal trip condition based on the current process variable value, the selected trip setpoint, and the exceedance factor. For the low-trip functions considered in this case study, the test input is calculated as follows:

$$X_{\text{test}} = X_{\text{PV}} - (X_{\text{PV}} - SP_0) \times K_{\text{exc}} \qquad (2)$$

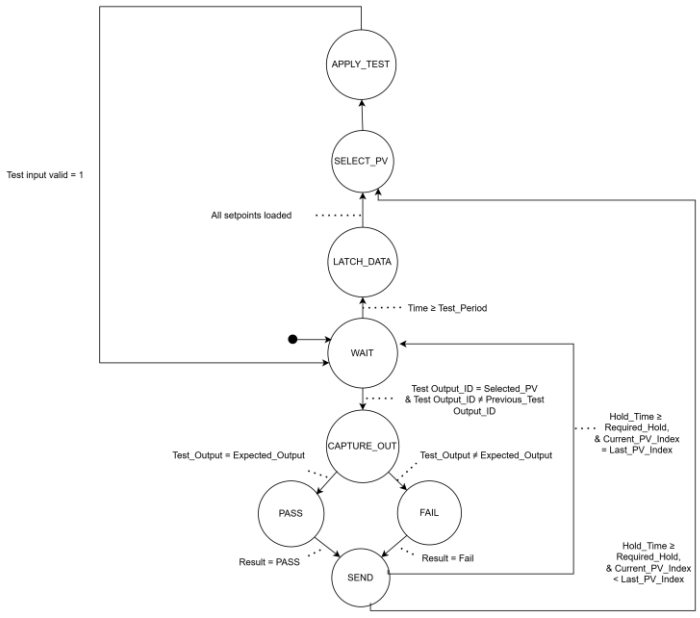
where X_{test} is the generated test input, X_{PV} is the current process variable value, SP_0 is the current setpoint, and K_{exc} is the exceedance factor.

The generated stimulus is then applied to the bistable logic by setting the test input valid signal to 1. In the CAPTURE_OUT state, the bistable output corresponding to the applied test stimulus is captured and validated. The transition condition, Test_Output_ID = Selected_PV and Test_Output_ID ≠ Previous_Test_Output_ID, ensures that the acquired output corresponds to the currently selected process variable and represents a newly generated response rather than a previously latched output.

The logic transitions to the PASS state when the captured output matches the expected trip decision, expressed as Test_Output = Expected_Output. Conversely, if the captured output does not match the expected output, expressed as Test_Output ≠ Expected_Output, the logic transitions to the FAIL state.

In the SEND state, the final test result, either PASS or FAIL, is transmitted to the Maintenance and Test Panel (MTP) after being maintained for the required hold duration, expressed as Hold_Time ≥ Required_Hold. The PV index comparison, Current_PV_Index < Last_PV_Index or Current_PV_Index = Last_PV_Index, determines whether the logic proceeds to the next PV for continued testing or returns to the WAIT state for the subsequent test cycle.

Fig. 3 State Transition Diagram of the Built-in Automatic Testing Method for Bistable Logic with a Manual-Reset Variable Setpoint



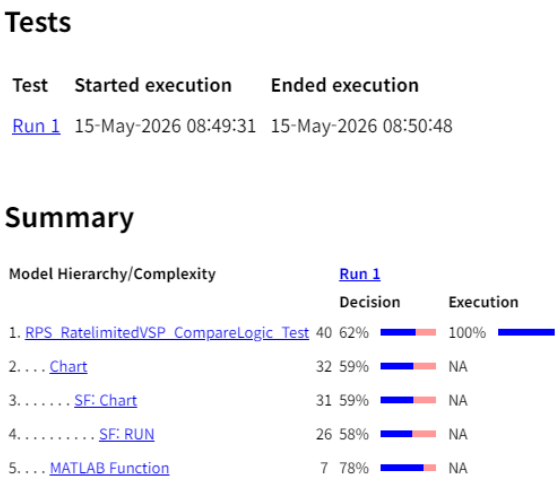
3.3 Implementation and Coverage Analysis of the Proposed State-Transition Model

A coverage analysis was conducted using the MATLAB Simulink Coverage tool to evaluate whether the implemented state-transition model followed the intended operating sequence. In this simulation, the target protection logic was not directly connected; instead, a simplified response-generation function was used to emulate the output behavior of the protection logic. Therefore, the coverage analysis

focused on verifying the operability of the proposed test sequence, including state transitions, test stimulus generation, response acquisition, and the PASS/FAIL evaluation flow.

As shown in Figures 4 and 5, all defined states achieved 100% block execution coverage, confirming that the implemented model executed all intended states. The lower decision coverage resulted from the use of the emulated protection-logic response rather than direct integration with the target protection logic. These results demonstrate that the implemented model operates in accordance with the designed state-transition model.

Fig. 4. MATLAB Coverage Test Results for the Rate Limited Variable-Setpoint State Transition Model



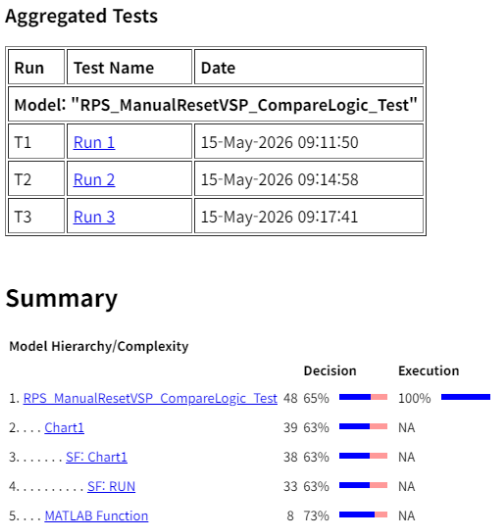
Tests

Test	Started execution	Ended execution
Run 1	15-May-2026 08:49:31	15-May-2026 08:50:48

Summary

Model Hierarchy/Complexity	Run 1			
		Decision		Execution
1. RPS_RatelimitedVSP_CompareLogic_Test	40	62%		100%
2. . . . Chart	32	59%		NA
3. . . . . SF: Chart	31	59%		NA
4. . . . . SF: RUN	26	58%		NA
5. . . . MATLAB Function	7	78%		NA

Fig. 5. MATLAB Coverage Test Results for the Manual Reset Variable-Setpoint State Transition Model



Aggregated Tests

Run	Test Name	Date
Model: "RPS_ManualResetVSP_CompareLogic_Test"		
T1	Run 1	15-May-2026 09:11:50
T2	Run 2	15-May-2026 09:14:58
T3	Run 3	15-May-2026 09:17:41

Summary

Model Hierarchy/Complexity				
		Decision		Execution
1. RPS_ManualResetVSP_CompareLogic_Test	48	65%		100%
2. . . . Chart1	39	63%		NA
3. . . . . SF: Chart1	38	63%		NA
4. . . . . SF: RUN	33	63%		NA
5. . . . MATLAB Function	8	73%		NA

4. CONCLUSIONS

This study extended the built-in automatic periodic surveillance testing (PST) methodology for FPGA-based reactor protection logic to variable setpoint bistable logic. Whereas the previous logic-model demonstration focused on fixed-setpoint bistable logic, this study applied the proposed method to more complex variable setpoint comparison logic, including bistable logic with a rate limited variable setpoint and bistable logic with a manual reset variable setpoint. The results demonstrate that the built-in automatic testing sequence can be systematically modeled and applied to different types of variable setpoint protection logic using a state-transition model.

The proposed test sequences were modeled using state-transition diagrams and implemented in MATLAB/Simulink with Stateflow to evaluate their functional feasibility at the logic-model level. The models describe the automatic test sequence at the state-transition level, including data loading, test stimulus generation, output capture, PASS/FAIL decision logic, and the SEND state representing result reporting to the Maintenance and Test Panel (MTP). Through this modeling approach, the study shows that the proposed built-in automatic testing method can be extended from fixed-setpoint logic to variable-setpoint bistable logic.

This study was limited to logic-model-level implementation and simulation. HDL synthesis, FPGA-board implementation, and hardware interface verification with external test equipment were not addressed. Therefore, the hardware-level executability, timing behavior, and physical interface characteristics of the proposed test logic remain to be verified.

Future work will describe the proposed state-transition models in a hardware description language (HDL), synthesize the models, and implement them on an FPGA development board. Additional verification and validation (V&V) activities will be conducted to evaluate functional isolation between the test function and the protection function, fault-detection coverage, response-time characteristics, and applicability to safety-related digital protection systems.

Acknowledgements

This work was supported by the Innovative Small Modular Reactor Development Agency grant funded by the Korea Government (MCEE) (No. RS-2024-00408005).

References

[1] Institute of Electrical and Electronics Engineers (IEEE). “IEEE Standard Criteria for the Periodic Surveillance Testing of Nuclear Power Generating Station Safety Systems”, IEEE Std 338-2012, (2012).
[2] U.S. Nuclear Regulatory Commission (U.S. NRC). “Periodic Testing of Electric Power and Protection Systems”, Regulatory Guide 1.118, Rev. 3, (2007).
[3] V. Agarwal et al. “Technical Specification Surveillance Interval Extension of Digital Equipment in Nuclear Power Plants: Review and Research”, INL/EXT-19-54251, Idaho National Laboratory, (2019).
[4] Y. Lee et al. “The Fault-Tolerant Evaluation Model Due to the Periodic Automatic Fault Detection Function of the Safety-Critical I&C Systems in Nuclear Power Plants”, Nuclear Engineering and Technology, Vol. 53, No. 1, pp. 229-239, (2021).
[5] Institute of Electrical and Electronics Engineers (IEEE). “IEEE Standard for System, Software, and Hardware Verification and Validation”, IEEE Std 1012-2024, (2024).
[6] J.-J. Lu, T.-C. Hsu, and H.-P. Chou. “System assessment of an FPGA-based RPS for ABWR nuclear power plant”, Progress in Nuclear Energy, Vol. 85, pp. 44-55, (2015).
[7] U.S. Nuclear Regulatory Commission (U.S. NRC). “Wolf Creek Generating Station, Issuance of Amendment No. 181: Modification of the Main Steam and Feedwater Isolation System Controls”, ADAMS Accession No. ML090610317, (2009).
[8] J. She and J. Jiang. “On the speed of response of an FPGA-based shutdown system in CANDU nuclear power plants”, Nuclear Engineering and Design, Vol. 241, No. 6, pp. 2280-2287, (2011).
[9] J. Lee et al. “The Automatic Test Features of the IDiPS Reactor Protection System”, Nuclear Engineering and Technology, Vol. 39, No. 3, pp. 299-308, (2007).
[10] H. Jeon et al. “Development of Automatic Testing System for PLC-based Reactor Protection Systems”, Nuclear Engineering and Technology, Vol. 43, No. 2, pp. 113-122, (2011).
[11] T. Y. Jhee, T. R. Kim, and J. Kim. “Development of a Built-in Automatic Testing Method for Digital Protection Logic”, Transactions of the Korean Nuclear Society Spring Meeting, Jeju, Korea, May 7-8, (2026).
[12] Korea Hydro & Nuclear Power Co., Ltd. (KHNP). “APR1400 Design Control Document Tier 2, Chapter 7: Instrumentation and Controls”, Rev. 3, (2018).