Emulated Programmable Logic Controllers for the Protection and Safety Monitoring and Operation I&C Systems in a Representative PWR Plant for Cybersecurity Applications

Mohamed S. EL-Genk, Timothy Schriener, Andrew Hahn, Ragai Altamimi
Institute for Space and Nuclear Power Studies and Nuclear Engineering Department, University of New Mexico, Albuquerque, NM, USA
Raymond Fasano, Christopher Lamb
Sandia National Laboratories, Albuquerque, NM, USA

Technical Work Scope: *Nuclear Energy–Cybersecurity Research Topics and Metrics Analysis (NE-1).* DOE-NEUP Project 18-15055. DOE Contract No. Nu-18-NM-UNM-050101-01 to University of New Mexico (UNM)

Performance Period: 04-01-2019 to 09-30-2020

Report No. UNM-ISNPS-03-2020
Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA, http://isnps.unm.edu/reports/

October 2020

Executive Summary

The expanding use of digital instrumentation and control (I&C) systems in commercial nuclear power plants and energy infrastructure raises cybersecurity concerns and emphasizes the needs for high-fidelity measures. The Nuclear Instrumentation & Control Simulation (NICSim) platform, currently being developed at the University of New Mexico's Institute for Space and Nuclear Power Studies (UNM-ISNPS) in collaboration with Sandia National Laboratories (SNL) under a DOE NEUP award, attempts to address some of these needs. In addition to its emulytics capabilities this platform couples the emulation and simulation models of digital I&C system components to a dynamic, physics-based model of a representative PWR plant for conducting cybersecurity investigations of the I&C systems

The PLCs in the I&C system for a representative PWR plant serve different safety and autonomous control functions. They include: (a) a Core Protection Calculator (CPC) PLC for performing the reactor trip function, (b) an Engineered Safety Features Actuation System (ESFAS) PLC for autonomously actuating the plant's engineered safety features, and (c) a Coincidence Logic Processor (CLP) PLC, which compares voting signals from the four separate safety divisions and determines if there is a 2/4 voting coincidence to act.

The objective of this work is to develop and test emulated PLCs in a representative PWR plant for regulating the reactor power, adjusting the pressure and water level in the pressurizer, regulating the feedwater inflow to adjust the water level in the steam generator to remain within preset points, and regulating the shaft rotation speed of the reactor coolant pump. The reactor power is regulated by moving the control element assemblies within the core and monitoring any mismatch between the reported thermal power and that calculated by the physics-based model of the reactor primary loop. The pressurizer's pressure PLC regulates the pressure in the primary loop by controlling the operation of the water immersed proportional and backup heaters, the liquid spray nozzle, and if needed, the pressure relief nozzle or valve. The pressurizer's water level PLC adjusts the charging and letdown flow rates to regulate the coolant inventory in the primary loop. The feedwater control PLCs maintain the water level on the shell side of the steam generators within pre-programed setpoints by adjusting the feedwater injection rate in the shell side of the steam generator. The reactor coolant pump PLCs regulate the shaft rotational speed to match the coolant flow rate to that calculated using the physics-based model of the primary loop.

The CPC and ESFAS PLCs continuously receive and compare values of state variables to those calculated by the developed physics-based model of a representative PWR primary loop. They also calculate and compare the values of the safety parameters to preprogramed setpoints to determine whether to vote to trip the reactor or actuate one of the engineered safety features. The operation PLCs autonomously control various components in the developed physics-based integrated model of a representative PWR plant. They regulate the reactor power, the system pressure, the water levels in the pressurizer and steam generators, and the speed of the reactor coolant pumps.

The emulated PLCs developed in this work are tested in simulated operation transients to ensure the reliability and fidelity of the setpoint controllers and to determine responses. The

calculated responses of the PLCs with PI controllers are used to determine the values of the proportion and integral gain coefficients to achieve a smooth system response in simulated operational transients. The calculated PLC signal response delay of 50ms is acceptable for current and future industrial applications. The determined coefficients of the PI controller for the pressurizer's water level PLC to achieve a smooth response during simulated transients are P = -2800 and I = -50. The testing results of the steam generator feedwater control PLC in a simulated reactor startup sequence, show that P = 0.02 and I = 0.6 produce the smallest difference between the normalized feedwater flow rate and the normalized steam generator water level. Testing of the reactor coolant pump PLC show that P = 0.001 and I = 0.0415 produce smooth responses of the shaft rotation speed and coolant flow rate.

The research detailed in this report has developed and demonstrated important elements for future implementation of the emulated PLCs and the physics-based model of a representative PWR plant into NICSim platform. The emulated PLCs developed and tested in this work provide direct control of the developed integrated model of a representative PWR plant during simulated transients. The emulated I&C system architectures and PLCs will be integrated within the SCEPTRE framework at SNL to support future cybersecurity analyses and investigation of PWRs and other nuclear power plants.

List of Contents

List of Figures

List of Tables

Nomenclature and Abbreviations

A_{cs}: Cross sectional area (m^2)

C: ANL CHF correlation coefficient

C_{ax}: Axial correction factor in the core

C_{rad}: Radial hot channel correction factor for the core

C_p: Specific heat capacity (J/kg.K)

C*: Combined control parameter for PLCs

G: Mass Flux (kg/m^2)

H_{pzr} Height of pressurizer (m)

I: Integral gain constant for PI controller

L: Water level (m)

L_d: Desired water level (m)

L_{max}: Maximum water level (m)

m: CHF correlation exponent

$\dot{m}$: Mass flow rate (kg/s)

$\dot{m}_{ch}$: Charging rate into primary loop (kg/s)

$\dot{m}_{fw}$: Feedwater flow rate (kg/s)

$\dot{m}_{fw,max}$: Maximum feedwater flow rate (kg/s)

$\dot{m}_{ld}$: Letdown rate from primary loop (kg/s)

$\dot{m}_p$: Flow in steam generator U-tubes (kg/s)

$\dot{m}_{rv}$: Pressurizer relief valve flow rate (kg/s)

$\dot{m}_s$: Steam flow rate (kg/s)

n_{rods} Number of fuel rods in core

p: Pressure (Pa)

p_{sg}: Steam generator pressure

p_{sys}: System pressure (Pa)

P: Proportional gain constant for PI controller

P_{Rx}: Reactor thermal power (W)

P_{th}: Reactor thermal power calculated from energy balance (W)

P_s: Setpoint reactor thermal power (W)

q'': Heat flux (W/m^2)

q''_{av}: Fuel rod average surface heat flux (W/m^2)

t:	Time (s)
T_{ave}	Average temperature (K)
T_b:	Bulk coolant temperature (K)
T_{ex}:	Exit temperature (K)
T_{in}:	Inlet temperature (K)
X_{db}:	Deadband filter size

Greeks

α_{CEA}:	CEA movement rate (m/s)
β:	Delayed neutron fraction, non-dimensional torque
Δp_c:	Curvature pressure loss (Pa)
Δp_{cl}:	Cold leg pressure loss (Pa)
Δp_{con}:	Contraction pressure loss (Pa)
Δp_{exp}:	Expansion pressure loss (Pa)
$\Delta p_{exp-con}$:	Sum of expansion and contraction pressure losses (Pa)
Δp_f:	Friction pressure losses (Pa)
Δp_{hl}:	Hot leg pressure loss (Pa)
Δp_{loss}:	Pressure losses (Pa)
Δp_{pump}:	Pump pressure head (Pa)
Δt:	Timestep size (s)
φ:	PLC sampling rate (Hz)
ρ:	Reactivity, density (kg/m^3)
ρ_{ex}:	External reactivity ($)
ρ_{fb}:	Feedback reactivity ($)
ρ_{tot}:	Total reactivity($)

Abbreviations

BOL:	Beginning of Life
CEA:	Control Element Assembly
CHF:	Critical Heat Flux
CHFR:	Critical Heat Flux Ratio
CPC:	Core Protection Calculator
DHCP:	Dynamic Host Control Protocol
DOE:	Department of Energy

EOL:	End of Life
ESF:	Engineered Safety Features
ESFAS:	Engineered Safety Features Actuation System
HITL:	Hardware-In-The-Loop
IAPWS:	International Association for the Properties of Water and Steam
I&C:	Instrumentation and Control
ICS:	Industrial Control System
I/O:	Input-Output
IT:	Internet Technology
NEUP:	Nuclear Engineering University Program
NICSim:	Nuclear Instrumentation and Control Simulation
OT:	Operational Technology
PCAP:	Packet Capture
PI:	Proportional-Integral
PID:	Proportional-Integral-Differential
PLC:	Programmable Logic Controller
PWR:	Pressurized Water Reactor
QR:	Query-Response
RCP:	Reactor Coolant Pump
RQ:	Response-Query
SG:	Steam Generator
SP:	Setpoint
SNL:	Sandia National Laboratories
TCP:	Transmission Control Protocol
TCP/IP:	Transmission Control Protocol over Internet Protocol
UDP:	User Datagram Protocol
UNM-ISNPS:	University of New Mexico’s Institute for Space and Nuclear Power Studies
VM:	Virtual Machine

1. Introduction

In this and past decades targeted cyberattacks on critical energy infrastructure have occurred on Industrial Control System (ICS) networks, including prominent attack campaigns such as Stuxnet, Havex, BlackEnergy III, and CrashOverride (Dragos Inc., 2017a; Dragos Inc., 2017b; Falliere, Murchu, Chien, 2011; Karnouskos 2011). The reported outcomes indicate that the developed and deployed malicious cyberattacks can destabilize and disable specialized industrial digital control systems. These digital control systems rely on specialized computers, such as Programmable Logic Controllers (PLCs), for monitoring and autonomous control of key functions. Unlike enterprise Information Technology (IT) networks, ICS networks frequently do not have the same cybersecurity safeguards and defensive technologies to confront increasingly sophisticated cyberattacks. While these systems may be isolated within a facility network, prior attacks have shown that isolation, while an important measure, is insufficient protection.

The increasing use of digital instrumentation and control (I&C) systems in current and future commercial nuclear power plants is a cybersecurity major concern in the nuclear industry (National Research Council, 1997; Korsah, et al., 2008). A targeted cyberattack could potentially have severe consequences to the operation and safety of a nuclear power plant. The recent infiltration of Wolf Creek's business network in 2018, a Westinghouse Pressurized Water Reactor (PWR) in Kansas, demonstrates the ever present and evolving cyber threat to nuclear power plants in the United States (Perlroth, 2019). Furthermore, since future nuclear power plants would be designed with mostly or all-digital I&C infrastructures, there is an urgent need for high fidelity cybersecurity measures and detailed analyses of their I&C architectures.

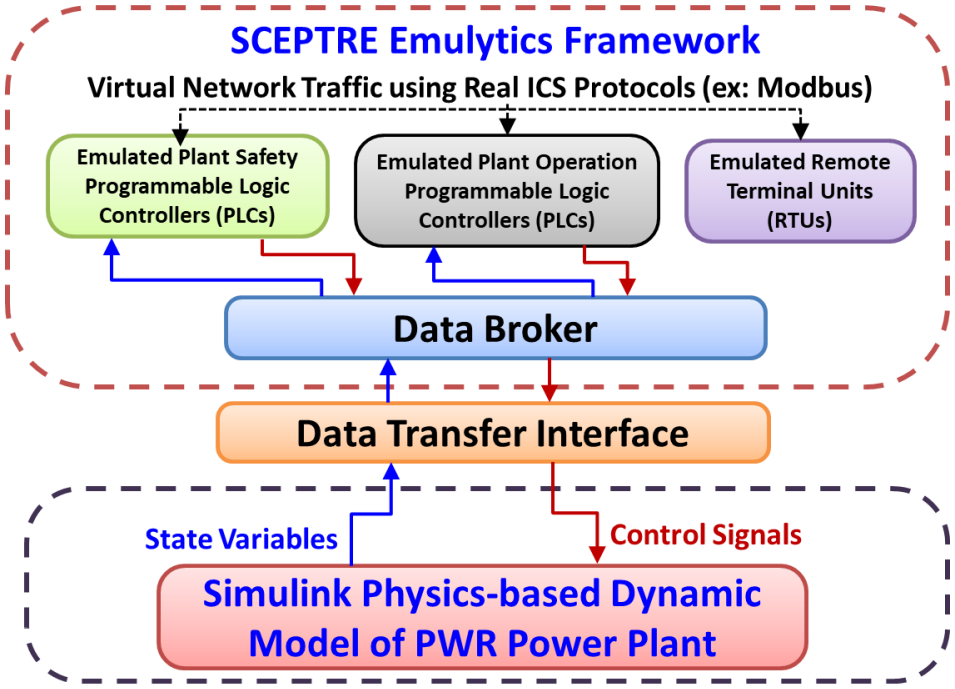

Fig. 1.1: Nuclear Instrumentation and Control Simulation (NICSim) Platform (El-Genk et al., 2020a, 2020b, 2020c)

To address some of these needs, the Nuclear Instrumentation & Control Simulation (NICSim) platform is currently being developed at the University of New Mexico's Institute for

Space and Nuclear Power Studies (UNM-ISNPS) in collaboration with Sandia National Laboratories (SNL), under a DOE NEUP award. In addition to its emulytics capabilities, this platform couples emulation and simulation models of digital I&C system components to a dynamic, physics based model of a representative PWR plant for conducting cybersecurity investigations of the I&C systems (Fig. 1.1). The NICSim platform will be implemented into the Department of Energy's (DOE's) SCEPTRE emulation framework (Camacho-Lopez, 2016; Sandia National Laboratories, 2016). This framework has been initially developed to deal with cyberattacks on energy grids and is acquiring capabilities to emulate and simulate I&C system architectures in nuclear power plants. The NICSim platform will include models of the digital PLCs, which control many of the control and safety system actuation processes in nuclear power plants. The emulated and simulated I&C components are linked to a physics-based dynamic model of a representative commercial PWR plant to demonstrate operation and investigate the response of PLCs in the I&C systems (Fig. 1.1).

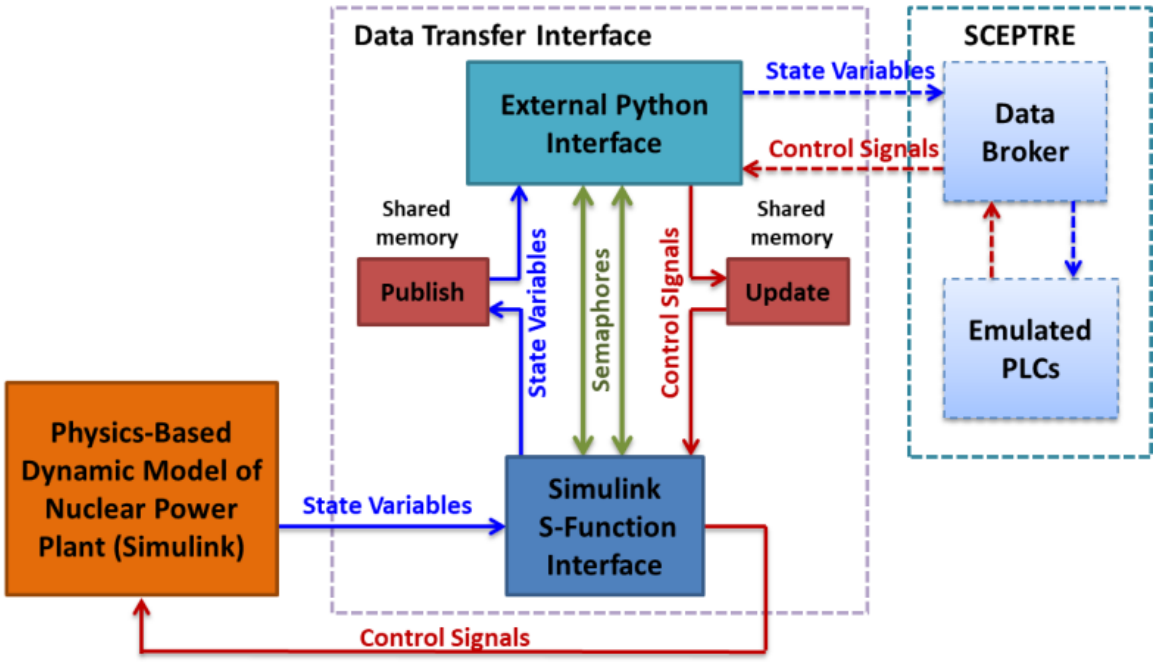

Fig. 1.2: A block diagram of NICSim's Data Transfer Interface (El-Genk et al., 2020a)

These emulated PLCs interface with the integrated PWR plant model running in Matlab Simulink (The Mathworks, 2019) using a fast running data transfer interface (El-Genk et al., 2019; Hahn, Schriener, El-Genk, 2020a, 2020b) (Fig. 1.2). A specialized Simulink S-function written in the C- programming language communicates the simulation state variables to an external interface program written in the python programming language (Fig. 1.2). The state variables calculated by the physics-based nuclear power plant model are communicated to the external python interface using shared memory inter-process communication (Hahn, Schriener, EL-Genk, 2020a, 2020b). The state variables are written and read from a shared memory location named 'Publish' (Fig. 1.2). The control signals from the I&C system's emulated PLCs pass back to the Simulink model through a second shared memory location named 'Update'.

Inter-process communication semaphores control access to the two shared memory locations and ensure that only one side of the interface can access a given shared memory location at a time. This arrangement avoids unreliable communication and prevents instabilities caused by

attempts to simultaneously access the memory location. The developed data transfer interface program also includes a time synchronization routine to ensure that the NICSim nuclear power plant model in Matlab Simulink runs in the same time scale as the emulated PLCs' control programming. For controllers using real-time clocks, this will ensure that the response time of the nuclear plant model is in tune with that expected by the controllers' software.

The NICSim platform employs a physics-based model of a representative, two-loop PWR plant (Fig. 1.3), detailed elsewhere (El-Genk et al., 2020a, 2020b, 2020c). The modeled PWR plant has two hot legs exiting the reactor and each has a U-tubes steam generator. The return water from each steam generator splits into two cold legs, each with a separate coolant pump (Fig. 1.3). The hot legs and the steam generators are labeled 1 and 2, the cold legs and reactor coolant pumps are labeled 1a, 1b, 2a, and 2b. The pressurizer is connected to hot leg 1.

The developed PWR plant model includes charging and letdown lines to inject and remove coolant from the primary loop to / from the reactor coolant management and chemistry control system. The charging and letdown lines are also used to adjust the water inventory in the primary loop and hence, the water level in the pressurizer. During nominal operation, coolant is continuously injected and removed from the primary loops to adjust the soluble boron concentration for reactor control and the water chemistry to limit corrosion. The charging line is connected to cold leg 2a, and the letdown line is connected to cold leg 2b. The steam generators 1 and 2 thermally couple the reactor primary loops to the secondary loops of the plants for electricity generation.

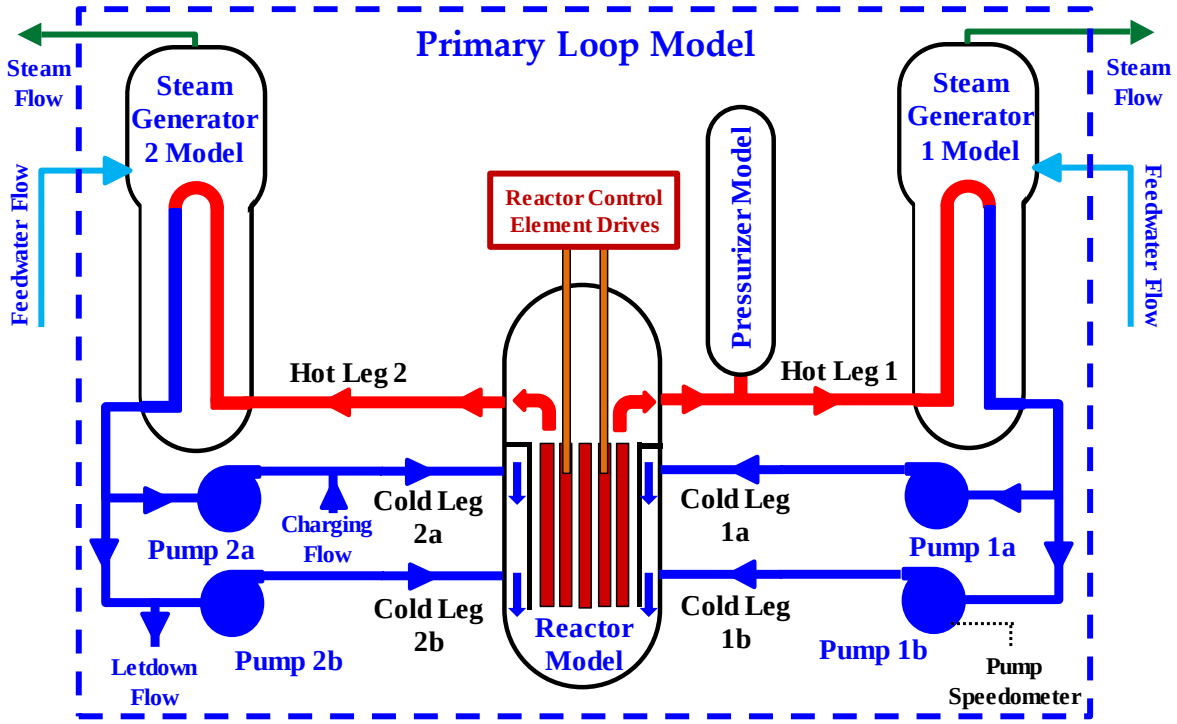

Fig. 1.3: Developed physics-based components model of a representative PWR plant (El-Genk et al., 2020b, 2020c).

The secondary loops are not modeled in detail, but the related operation parameters for the steam generator model are specified commensurate with the determined steam flow based on the reactor thermal power and primary loop coolant temperatures. The exit enthalpy and pressure of the steam existing the steam generator to the secondary loops are kept constant and equal to their

values for nominal plant operation. Thus, a change in the steam load demand will change the exit quality from the steam generator to the secondary loops and the enthalpy of the water exiting the steam generator to the cold legs of the primary loops (Fig. 1.3). The latter affects the reactor thermal power due to the negative temperature reactivity feedback in the reactor core.

The present PWR plant model is developed within the versatile Matlab Simulink platform (The Mathworks 2018), which is constructed using discrete-time blocks. It facilitates the compilation of the plant model into an executable by the Matlab Simulink Coder to support parallel processing. Simulink simultaneously solves the coupled governing equations in the physics-based models of the plant components and the primary loops using a fixed timestep discrete solver. The selected timestep size ensures numerical stability and convergence of results within a short running time. The selected modeling approach and efficient use of Matlab Simulink make it possible to run the developed PWR plant model synchronous with real time. Together the emulated PLCs, the developed plant and components models with be integrated into the NICSim platform at SNL. The developed PWR plant and components models could easily be configured for different designs, dimensions, materials properties, temperature reactivity feedback coefficients, primary coolant pump characteristics.

The focus of the work described in this report is to develop and demonstrate an emulated I&C system architecture for a representative PWR plant with several PLCs that serve various safety and autonomous control functions. The PLCs of the Plant's Protection and Safety Monitoring I&C System (PMS) provide an essential regulatory and safety functions. They autonomously trip the reactor and actuate the Engineered Safety Features (ESF) when the plant's operation exceeds programed safety setpoints. The plant operation I&C system assists the operators of the plant by automatically actuating control mechanisms to regulate the plant operation state variables, such as the primary loop system pressure and the reactor power, to within programed setpoints.

The next section describes the Protection and Safety Monitoring System I&C architecture in a representative PWR plant. This architecture will be incorporated into the NICSim platform. This section also describes the developed emulated Core Protection Calculator (CPC) PLC, Engineered Safety Features Actuation System (ESFAS) PLC, and coincidence logic processer PLC. Section 3 provides details of the autonomous controllers of the PWR plant. These are a reactor regulator PLC, pressurizer pressure and pressurizer water level PLCs, a steam generator feedwater control PLCs, and reactor coolant pump PLCs. Section 4 summarizes the results present and discussed in this report and the arrived at conclusions. Appendix-A provides an update to the Task-1 report on the Implementation and Validation of PLC Emulation and Data Transfer (El-Genk et al., 2019). It also provides an expanded description of the emulation methodology and presents and discusses testing and validation results.

2. Programmable Logic Controllers for Protection and Safety Monitoring in a Representative PWR Plant

The Plant Protection and Safety Monitoring I&C System (PMS) for a representative PWR plant incorporates several emulated PLCs that perform the reactor trip initiation and the Engineered Safety Features (ESF) actuation functions. This safety I&C system is isolated to a greater degree from other computing systems in the plant. The PMS comprises four separate divisions. Each division, with separate PLCs, is connected to independent sensor instruments to reduce the risk that an equipment failure in one division compromises the other three. The PLCs in each division vote independently on whether to trip the reactor or actuate one or more of the plants' safety systems. The votes are tallied together using coincidence logic processor PLCs. At least 2 of the 4 safety divisions votes are required to send the control signals to initiate a reactor trip or actuate the ESF.

A representative PWR plant's PMS comprises three types of PLCs. The reactor trip protection function is performed by the Core Protection Calculator (CPC) PLC. It is linked to a physics based PWR plant model and receives the state variables calculated by the developed integrated plant model and the models of the plant components. These state variables are analogous to the sensor instrument measurements received by the physical PLC's I/O modules in a real plant. The CPCs' programming takes these state variables and either use them directly or to calculate key safety parameters, which are compared to the programmed reactor trip setpoints. If the setpoints are exceeded, the CPC will send a voting signal for a trip. The Engineered Safety Features Actuation System (ESFAS) PLCs perform the ESF actuation function. They receive state variables from the PWR plant model and compare them to the safety setpoints for the different ESF systems' actuation functions. If the setpoints for one of the safety systems are exceeded, the ESFAS PLC will sent a voting signal calling for its actuation. The voting signals from the CPC and ESFAS PLCs are communicated to the coincidence logic processor PLCs which determine whether at least 2/4 safety divisions are voting to send the appropriate control signal to the PWR plant model.

Controllers for the PLCs in the plant's protection and safety monitoring I&C system are developed to support future cybersecurity investigations and analyses. These controllers emulate the PLCs operating system kernel and control software, and communicate using the same ICS communication protocols. Each PLC is emulated with a virtual machine running the open-source OpenPLC software and its control logic program (Alves, et al., 2014). The OpenPLC software runs control programs written in IEC 61131-3 standard PLC programming languages. The virtual machines use the VMWare virtualization platform (VMware, 2019), which runs an image of the Raspian operating system with OpenPLC installed. Individual emulated PLCs are created by changing the control programming within the OpenPLC runtime. The calculated values of the state variables by the developed PWR plant model are communicated via the data transfer interface to the control program within the OpenPLC runtime over the network using the Modbus ICS communication protocol over TCP/IP. The values of the state variable are stored as Modbus register values by OpenPLC. The control signals generated by the PLCs are communicated to the data interface program using Modbus over TCP/IP and transmitted back to the PWR plant model. Direct communication between the PLCs is also performed using Modbus signals over TCP/IP.

These PLCs are developed using an emulation methodology established by the NICSim project team to characterize the key physical and digital signatures of the PLC and validate these

signatures against those of an emulated PLC (Fasano, et al., 2020). Validation and testing of the PLC emulation methodology is conducted to determine the settings required to ascertain that the emulated PLCs replicate the performance and network traffic behavior of the physical devices. The description and validation testing for the PLC emulation methodology are the subject of a prior report (El-Genk, et al., 2019), and are updated in Appendix A of this report.

2.1 Core Protection Calculator PLCs

The Core Protection Calculator (CPC) PLCs perform the reactor trip voting function (Fig. 2.1) (Hahn, EL-Genk, Schriener, 2020a). The four independent CPC PLCs, one for each of the four safety divisions, receive values of the state variables from the physics based PWR plant model. These are the reactor thermal power, the positions of the control element assembly in the reactor core, the water temperatures in the hot and cold legs, and both the pressure and water level in the pressurizer. Each PLC uses two separate state variables to determine the coolant flow rate through the reactor primary loops from the calculated pressure losses across a segment of the hot leg and the from the pump supply curves and shaft speed of the primary coolant pumps. Each of the four CPCs receives the same set of state variable values (Fig. 2.1).

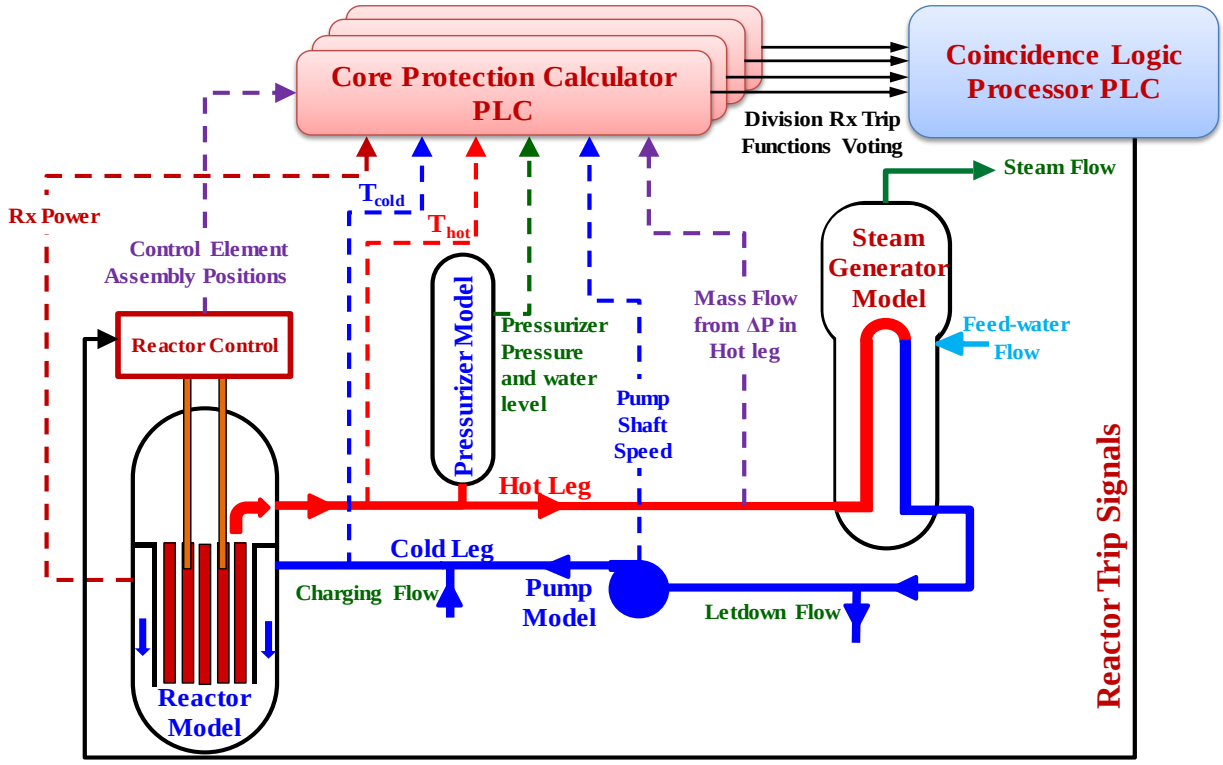

Fig. 2.1: A block diagram of reactor digital safety I&C system for the trip function (El-Genk et al., 2020b; Hahn, El-Genk, Schriener, 2020a).

The logic programming of the CPCs uses the provided plant state variables to calculate safety parameters and compare them to minimum setpoints (Fig. 2.2). These are: (a) the Critical Heat Flux Ratio (CHFR), (b) the reactor coolant flow rate based on sensor readings, and (c) the margin of the coolant core exit temperature from that for saturation at system pressure. The primary trip function of the CPC is to calculate the CHFR and compare it to the minimum allowable setpoint (Fig. 2.2). This setpoint is determined considering the response time of the

PLC to trip the reactor before the CHFR drops below 1.0 and boiling ensues in the hot channel. The minimum CHFR setpoint provides sufficient safety margin and enough time for the CPC to respond before reactor conditions become unsafe. The CHFR is calculated for the identified hot channel in the reactor core from the pre-cycle core design analysis. The CPC also calculates the axial distribution of the surface heat flux at 10 discrete nodes of the fuel rod in the hot channel and compares the lowest CHFR to the specified minimum setpoint.

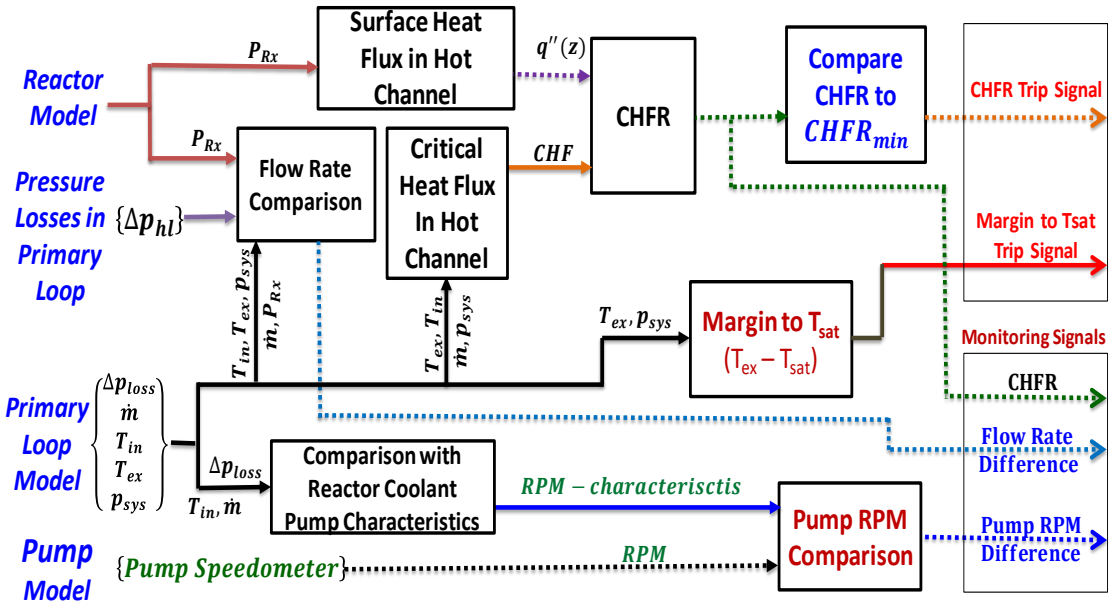

Fig. 2.2: A block diagram and procedures for determining the CHFR and temperature margin trip functions (El-Genk et al., 2020b; Hahn, El-Genk, Schriener, 2020a).

The fuel rod surface heat flux at an axial node j in the hot channel is calculated as:

$$q"(j) = q"_{av}\, C_{ax}(j)\, C_{rad} \tag{2.1}$$

In this expression $q"_{av}$ is the average surface heat flux for the fuel rods in the core based on the reported reactor thermal power, P_{Rx}, C_{rad} is the radial hot channel correction factor for the core, and $C_{ax}(j)$ is the axial correction factor in the core at the axial node j. The values of C_{rad} and $C_{ax}(j)$ are specific to a reactor core design and fuel loading and in practice are determined from neutronics analysis of the PWR plant. The Critical Heat Flux (CHF) is calculated using the ANL correlation (Jens and Lottes, 1951), as:

$$CHF(j) = C\left(\frac{G}{10^6}\right)^m \left(T_{sat} - T_b(j)\right)^{0.22} \tag{2.2}$$

In this correlation, the values of the coefficient C and the exponent m are tabulated as functions of the system pressure, G is the coolant mass flux, T_{sat} is the saturation temperature at the system pressure, and $T_b(j)$ is the bulk coolant temperature at axial node, j. The coolant mass flux for the hot channel is determined as:

$$G = \frac{(\dot{m}/n_{rods})}{A_{cs}} \tag{2.3}$$

The calculate values of CHF(j) are divided by those of the local heat flux q"(j) to determine the axial distribution of the CHFR (j) for the fuel rod in the hot channel, as:

$$\mathrm{CHFR(j)} = \left(\mathrm{CHF(j)}/\mathrm{q}^{''}\,\mathrm{(j)}\right) \tag{2.4}$$

The PLC's program takes the lowest CHFR(j) value as the minimum CHFR. When the calculated minimum CHFR reaches the lowest setpoint, the CPC sends a trip voting signal to a coincidence logic processer PLC.

The CPC also calculates the margin of the reactor coolant exit temperature from the saturation temperature, T_{sat}, and compares the difference to a programed setpoint (Fig. 2.2). If the margin drops below the setpoint, the CPC's program will send a trip voting signal to the logic coincidence counter PLC. The value of T_{sat} is calculated at the determined system pressure, p_{sys}, by the plant pressurizer. All the above calculations use programed thermophysical property correlations based on the values determined using the International Association for the Properties of Water and Steam (IAPWS) Industrial Formulation 1997 standard (International Association for the Properties of Water and Steam, 2007). The flexible CPC program could support different PWR core designs. While the tested reactor trip functions program focused on the minimum CHFR and temperature margin to T_{sat}, additional trip functions can be easily added to CPC's programming.

In addition to the trip protection functions, the CPC also monitors different plant operation state variables and sends warning signals to the operators if they exceed the programmed limits. Fig. 2.2 shows two of these functions. The first calculates the mass flow rate from the energy balance across the reactor using the state variables of T_{in}, T_{ex}, p_{sys}, and P_{Rx} as:

$$\dot{m} = \frac{P_{Rx}}{C_p(T_{ex} - T_{in})} \tag{2.5}$$

In this expression, the coolant heat capacity, C_p, is calculated as a function of the core average coolant temperature $(T_{ex} + T_{in})/2$ and the determined system pressure, p_{sys} by the pressurizer model. The determined mass flow rate from eq. (2.5) is compared to that determined from the measured pressure losses across a segment of the hot leg, Δp_{hl}, as:

$$\dot{m}_{hl} = \left[\frac{\Delta p_{hl}}{0.184 * \left(\frac{L}{D^{1.8} A c^{2.2}}\right)\left(\frac{\mu^{0.2}}{\rho}\right)}\right]^{1/1.8} \tag{2.6}$$

In this expression, L is the length of the hot leg segment across it the pressure drop is measured, D and A_c are is the inner diameter and the cross sectional flow area of hot leg pipe, and μ and ρ are the dynamic viscosity and density of the water coolant at the reactor core exit temperature and system pressure. During steady state operation, a difference between these two values of the coolant mass flow rate could indicate a malfunction of the measurements sensors or in the PLC performing the calculations. If the difference is larger than the programed tolerance, the CPC sends a warning signal to the operators in the control room.

The second monitoring function shown in Fig. 2.2 compares the determined rotation speed of the reactor coolant pumps by the developed pump model to the determined RPM from the pump characteristics based on the calculated total pressure losses, Δp_{loss}, the total coolant mass flow rate in the primary loops and the inlet temperature to the reactor core, T_{in}. The pump characteristics are presented in Fig. 2.3 as the pumping head versus the coolant mass flow rate, at different rotation speeds fo the pump shaft.

The pump rotation speed in RPM is determined from the determined from the intersections of the pump supply curves with demand curve for the reactor primary loop. The determined RPM is compared to that determined by the pump model. If this RPM differs from that predicted by the

pump characteristics in excess of the allowed tolerance, it may indicate a malfunction in one of the coolant pumps and the CPC sends a warning signal to the operator.

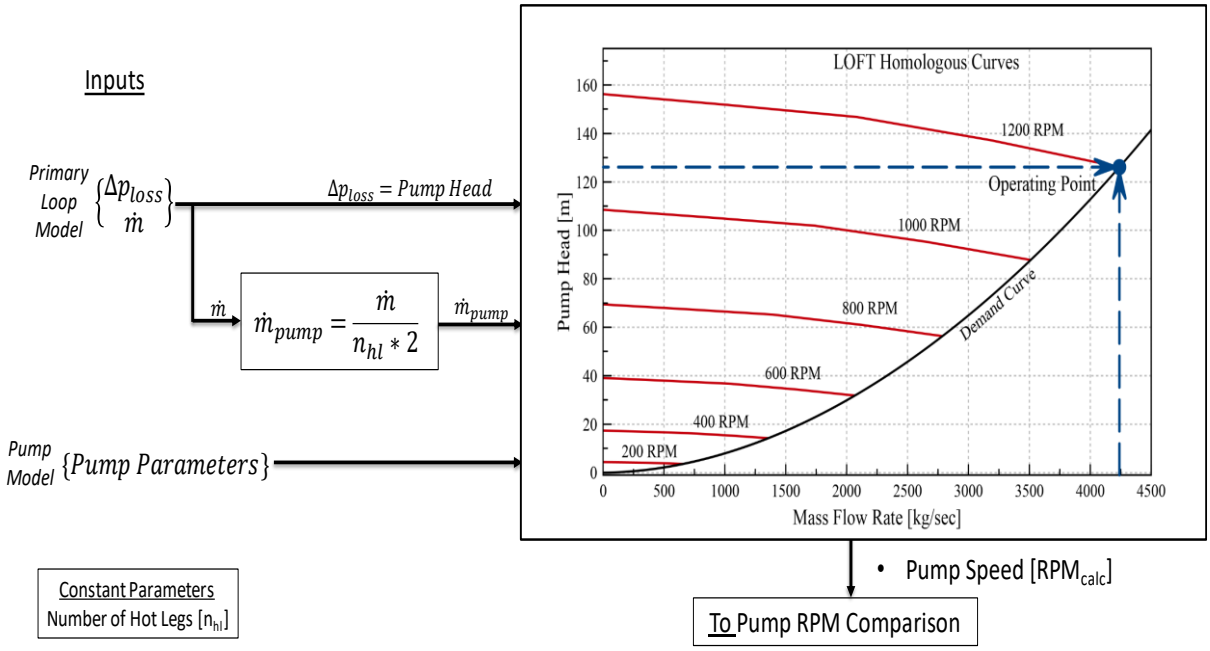

Fig. 2.3: Functional block diagram for calculating of the shaft RPM from the characteristics of the primary coolant pump.

The control program for the CPC PLC is written in structured text PLC programming language and uploaded into the OpenPLC runtime on the emulated PLC. The state variables values communicated to the PLCs program and the reactor trip voting and monitoring signals are stored as Modbus register values within the compiled OpenPLC control program. The data broker program writes the corresponding state variables values to the Modbus registers using a Modbus over TCP connection. The Modbus registers corresponding to the reactor trip voting signals are read by the coincidence logic processer PLC by a direct Modbus over TCP connection between the two PLCs.

2.1.1 Response Time Testing Results

The developed CPC is tested to determine its response characteristics and the response time of the reactor trip functions. The tested CPC PLC is integrated with the PWR reactor model using the developed Data Transfer Interface (Fig. 2.4), which is configured to communicate directly with the CPC PLC using Modbus over TCP outside of the SCEPTRE framework. The Simulink PWR model and the Data Transfer Interface program ran on a multiprocessor Linux server while the emulated PLC ran within a Raspian VM with OpenPLC operating on a networked Windows PC.

To evaluate the performance and response time of the CPC, the obtained results with the emulated CPC are compared to those of a mirror CPC implemented within the PWR Simulink model. The developed reactor model using this mirror CPC gives the expected response of the core protection PLC, but with no delay. This is because this ideal controller does not have to communicate the state variables to the external emulated PLC and the resulting control signal back to the PWR plant model.

Fig. 2.5 shows an example of the trip response delay for the CFHR trip function. The

calculated value of the CHFR by the PLC lags that calculated using the same equations built into the 'ideal' Simulink controller. The trip vote signal is generated when the calculated CHFR in the PLC decreases below the setpoint limit. This occurs slightly after the CHFR values for the ideal controller reaches the trip setpoint (Fig. 2.5). The difference between when the ideal CHFR and the PLC CHFR reach the setpoint is defined as the trip response delay.

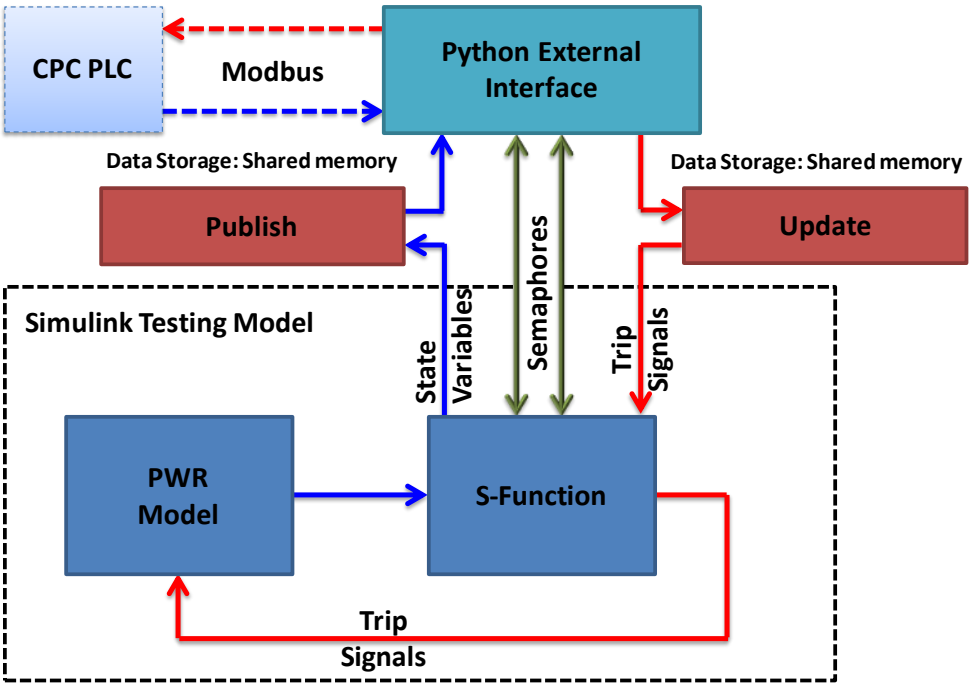

Fig. 2.4: Block diagram of the testing setup of the CPC response characteristics (Hahn, EL-Genk, Schriener, 2020a, 2020b)

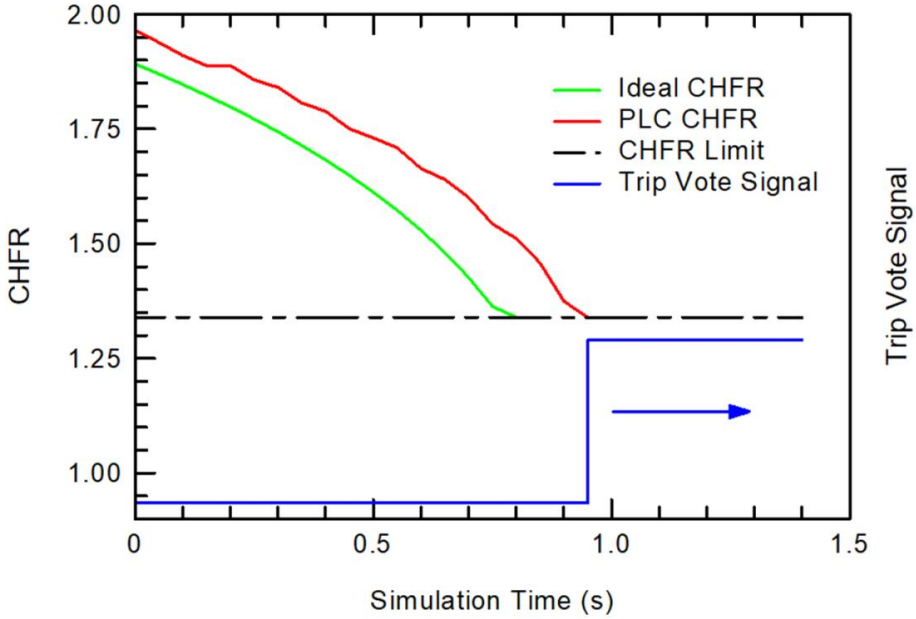

Fig. 2.5: Calculated CPC response for ideal controller and emulated PLC for CHFR versus simulation time

Figures 2.6a-c present the results of a test scenario designed to initiate a CHFR trip by the emulated PLC that is linked to the dynamic physics-based model of the reactor in a representative PWR plant (Figs. 2.2 and 2.2). For this testing setup, the PLC is configured with a sampling rate of 50 Hz, and simulation timestep of 50 ms. The test scenario begins with the reactor operating at steady state at 50% of its nominal thermal power. Then positive external reactivity is inserted at a rate of 0.01 cents/sec until the PLC sends a trip signal. (Fig. 2.6a). Fig. 2.6b and 2.6c present the data of the CHFR trip test and compare the response of the emulated PLC and to that of the ideal simulation. Fig. 2.6a comperes the values of the inserted external reactivity, the feedback reactivity, and the total reactivity.

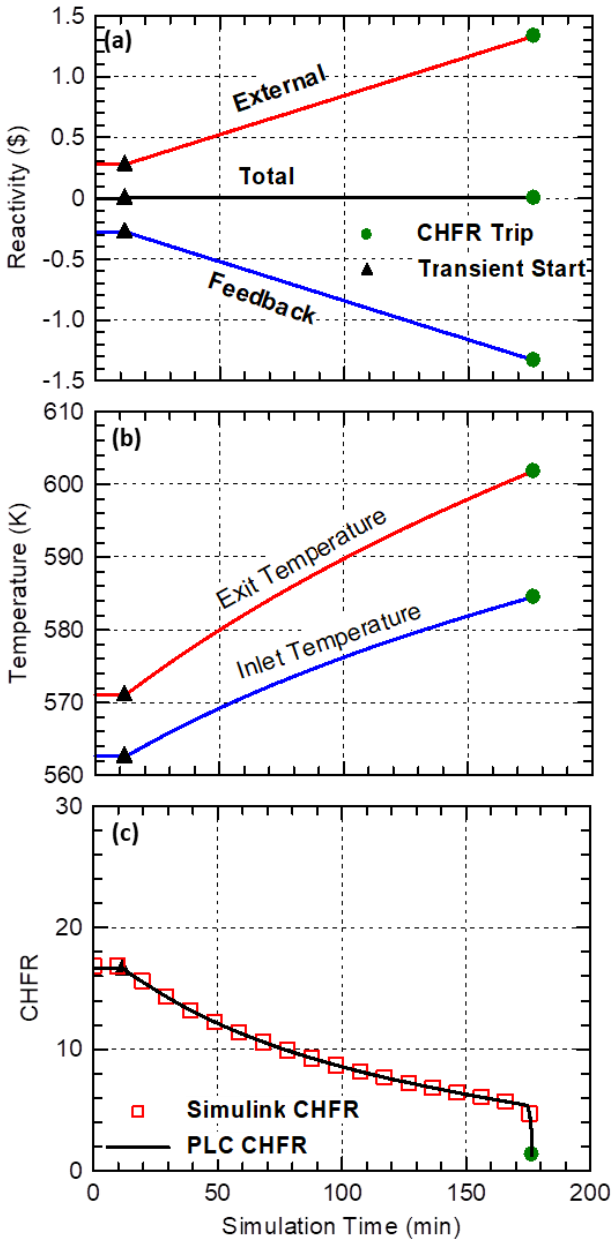

Fig. 2.6: Results of simulated transients of a reactor startup and an initiated trip by the CHFR CPC (Hahn, EL-Genk, Schriener, 2020a).

The changes in the reactor thermal power and the inlet and exit coolant temperatures are displayed in Fig. 2.6b. In Fig. 2.6c the calculated CHFR by the PLC is compared to that of the ideal internal CPC in the Simulink model. The inserted external reactivity (Fig. 2.6a) increases the reactor thermal power and subsequently the reactor core inlet and exit coolant temperatures (Fig. 2.6b). The negative temperature reactivity feedback in the reactor core maintains the total reactivity slightly above \$0. The calculated CHFR by the CPC PLC in the identified hot channel in the core decreases following the start of the reactivity insertion until the PLC signals for a trip, 176 minutes later (Fig. 2.6c). The trip response of the PLC is very close to that of the ideal controller. The CHFR trip response delay response is ~ 150 ms relative to the ideal internal CPC. The trip signal delay response of 150 ms is within the industry accepted range for CHFR calculators (Hung, 2010).

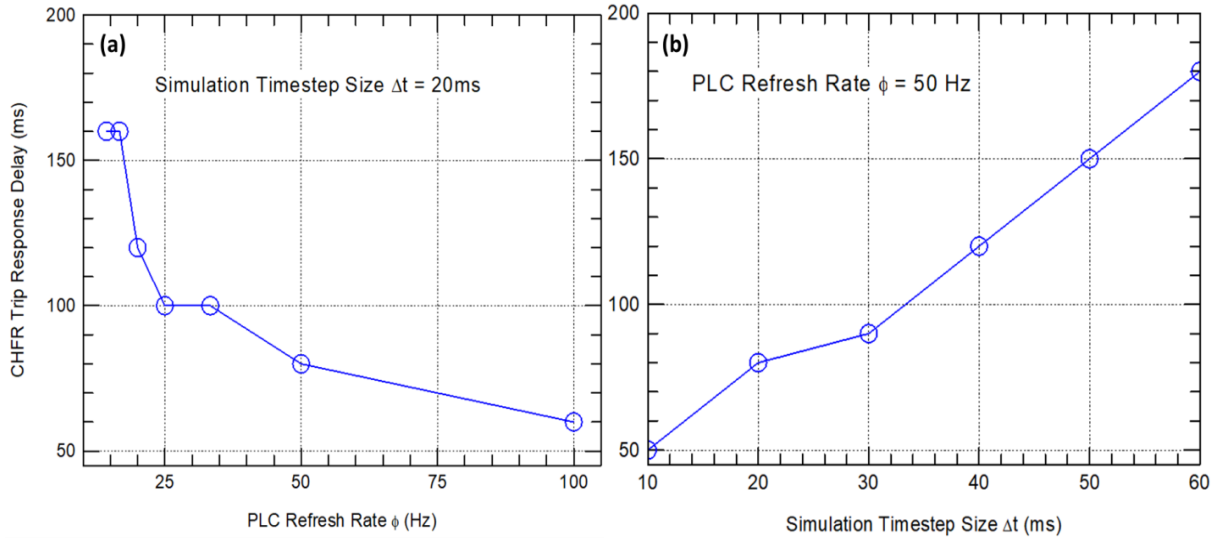

Fig. 2.7: Effects of PLC refresh rate and timestep size in simulations on the response delay of the CHFR trip function.

A parametric analysis is performed to determine the effects of the PLC refresh rate and the timestep size in the Simulink simulation model on the CPC PLC's trip signal delay response. Fig. 2.7a shows the results of the CHFR trip signal delay with different PLC refresh rates for a fixed timestep size of 20 ms. Fig. 2.7b shows the results of the trip response delay with different simulation timesteps for a fixed sampling rate of 50 Hz. A large decrease of the response delay time occurs when the sampling rate increases to 25 Hz, with a smaller reduction with further increase in the sampling rate (Fig. 2.7a). The CHFR trip response delay increases almost linearly with increased simulation timestep size (Fig. 2.7b). Based on the results of this analysis a PLC sampling rate of 50 Hz is selected for the emulated CPC PLC. The developed emulated CPC PLC has been successfully characterized and tested to ensure that the PLC trip and monitoring functions would initiate voting or warning signals when the setpoints are exceeded. This emulated PLC is ready to be integrated into the NICSim platform alongside other safety and operation PLCs in the I&C system to investigate cybersecurity risks in I&C systems.

2.2. Engineered Safety Features Actuation System PLC

The Engineered Safety Features Actuation System (ESFAS) PLC performs the automatic actuation function for the plant's ESF. The four independent ESFAS PLCs receive values of the state variables from the components' models in the PWR plant model. These state variables are

the hot and cold leg coolant temperatures, the system pressure, the water levels in the pressurizer, and the water level and internal pressure in the steam generators (Fig. 2.8). The received values are compared to setpoints programed within the PLCs for the different ESF systems of the plant. If the PLC programming determines that any of the plant state variables exceeds its safety setpoint for an ESF system, the PLC generates a voting signal to actuate the system to the coincidence logic processor PLC. The coincidence logic processor PLC receives the voting signals from the four ESFAS and sends an actuation signal to the systems components (Fig. 2.8).The representative ESFAS PLC that is developed for the NICSim platform contains the following actuation functions to: (a) the safety injection system, (b) the containment isolation system, (c) the main steam isolation system, (d) the auxiliary feedwater, and (e) the auxiliary spray and letdown isolation (Table 2.1).

Table 2.1: Engineered safety functions for representative ESFAS PLC.

Engineered Safety Feature	Action	Initiating Events
Safety Injection System	Actuate high pressure injection of borated water into the primary loops and initiate emergency core cooling system	System pressure decreases below its preprogramed setpoint, SP_1
Containment Isolation System	Initiate isolation of process lines penetrating through the containment	System pressure decreases below its preprogramed setpoint, SP_1
Main Steam Isolation System	Isolate main steam, main feedwater, and blowdown lines for both steam generators	Pressure in steam generator decreases below its setpoint, SP_2, the average coolant temperature in primary loops decreases below its setpoint, SP_3, or water level in SG decreases below its setpoint, SP_4
Auxiliary Feedwater Actuation	Start auxiliary feedwater pumps and open auxiliary feedwater valves to one steam generator when signal is generated for other steam generator	Steam generator water level decreases below its setpoint, SP_5, or the pressure difference between the two SGs in the two hot legs of the plant is higher than setpoint, SP_6
Auxiliary Spray and Letdown Isolation	Close letdown valve and switch liquid spray in the pressurizer to auxiliary water system	Pressurizer water level decreases below setpoint, SP_7

The ESFAS PLC has seven programed setpoints within its internal control logic. The SP_1 is a low pressure setpoint for the pressurizer, SP_2 is the low pressure setpoint in the steam generators, and SP_3 is the low setpoint for the reactor average temperature, calculated as the average of the hot leg and cold leg temperatures. The SP_4 and SP_5 are the setpoints for the low water level in the two steam generators of modeled PWR plant (Fig. 1.3). The setpoint SP_6 is that for the pressure difference between the two steam generators, calculated as $|p_{sg1} - p_{sg2}|$. This setpoint is exceeded when the magnitude of the pressure difference increases above the programed value. The setpoint SP_7 is for a low water level in the pressurizer.

The program for the ESFAF PL is written in the structured text PLC programming language and uploaded into the OpenPLC runtime on the Raspian VM. As with the CPC PLC above, the data broker program writes the state variables to the Modbus registers, while the ESF actuation-voting signals are read by the coincidence logic processer PLC using a direct Modbus over TCP connection.

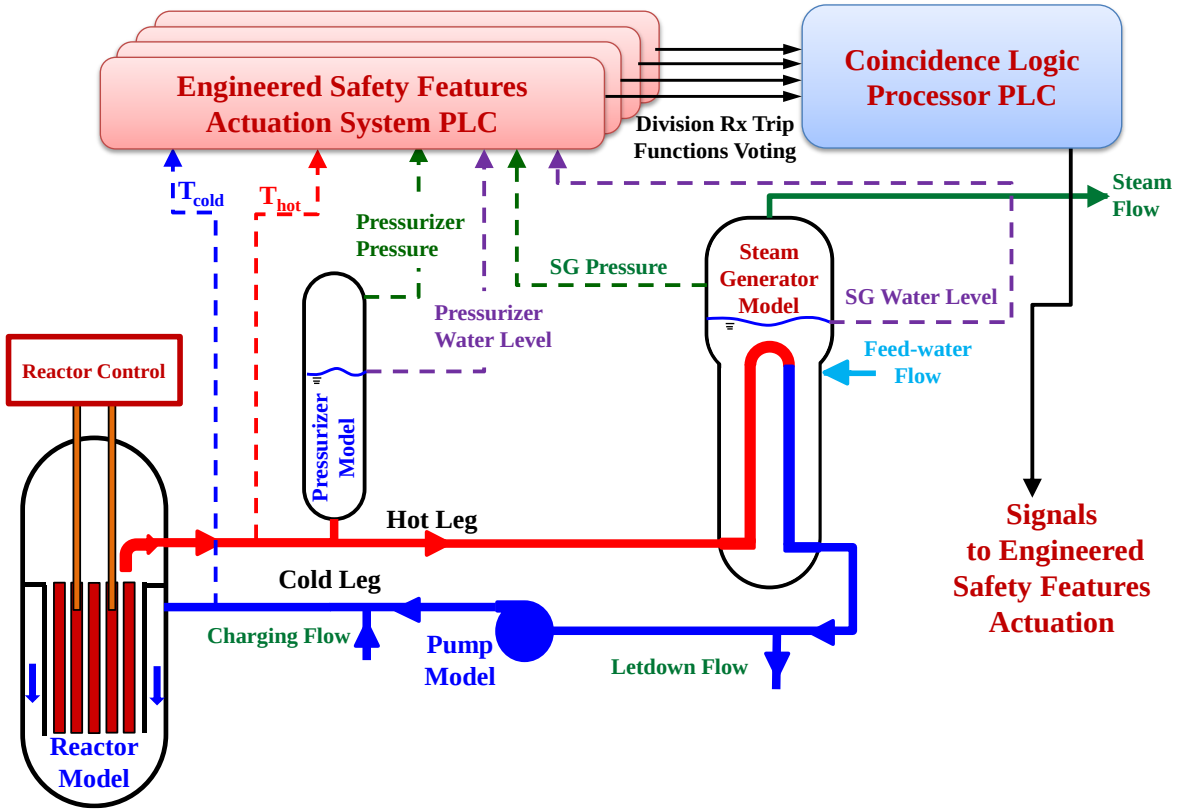

Fig. 2.8: A block diagram of reactor digital safety I&C system (El-Genk et al., 2020a, 2020b).

2.3. Coincidence Logic Processor PLC

The PLC of the coincidence logic processer compares the voting signals of the four separate safety divisions' CPC PLCs (Fig. 2.1). The coincidence logic processer PLC generates a reactor trip signal when the required 2/4 voting coincidence is satisfied. This signal is then communicated back to the nuclear plant model to trip the reactor or actuate the plants engineered safety features (Figs. 2.1 and 2.8). The emulated PLC runs two separate software programs. The first is a communication program written in the python programming language that handles the Modbus over TCP commination between the voting PLCs and the coincidence logic processer. The communication program serves as a Modbus over TCP client, which connects with the OpenPLC runtime on the emulated voting PLCs. The program queries the values of the Modbus registers related to the reactor trip or ESF actuation voting signals and writes the values to the input Modbus registers on the local OpenPLC runtime. The local OpenPLC program runs the voting coincidence logic program written in the structured text PLC programming language, and which compares the votes vote values from the CPC and ESFAS PLCs (Figs. 2.8 and 2.9).

The coincidence logic processer PLC's programming ensures that the PLCs give a steady vote to trip or actuate the ESF. To be counted as a positive vote, the signal needs to remain steady for at least three consecutive cycles. Requiring the voting signals to hold true for a specified period matches common practice in voting PLCs in PWR safety I&C systems (Schindhelm and Single, 2010). This ensures that an errant blip is insufficient to cause an accidental trip or ESF actuation event. The combinatorial logic checks for each combination of PLC voting by the four safety divisions (A, B, C, and D). These are positive voting by divisions AB, AC, AD, BC, BD, and CD. If any of the six valid combinations test true, the PLC generates a control signal for the

appropriate reactor trip or ESF actuation function and writes the value to the output Modbus registers. The collective values are communicated to the data broker in SCEPTRE, which transfers them to the developed data transfer interface program, which communicates them back to the developed physics-based model of a representations PWR plant (Fig. 2.1).

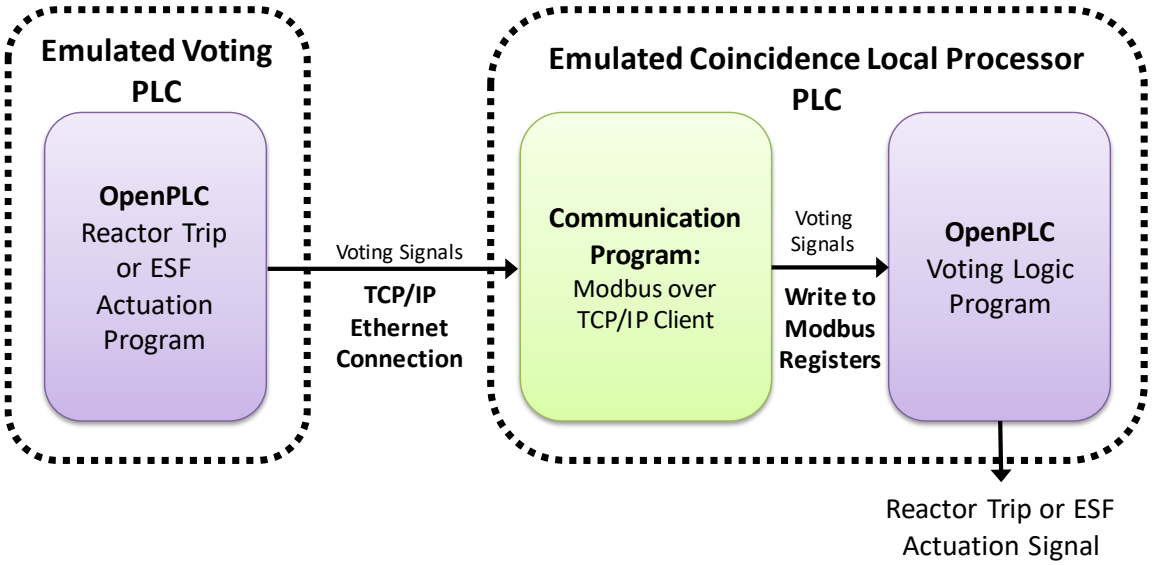

Fig. 2.9: A block diagram of communicating signals between the voting PLCs and the emulated coincidence logic processer PLC.

2.4 Summary

A representative PWR plant design and I&C system architecture include PLCs in the protection and safety monitoring I&C system. This work developed designs of a CPC PLC for performing the reactor trip function, the ESFAS PLC for autonomously actuating the plant's engineered safety features including the coincidence logic processor PLC, which compares the voting signals from the four separate safety divisions and determines whether there is 2/4 vote to act. The CPC and ESFAS PLCs continuously receive values of the state variables from the PWR primary loop model and compare them and the calculated safety parameters to programed setpoints. Such comparisons determine whether the PLC should vote to trip the reactor or actuate one of the plant's ESF systems.

The developed emulated CPC and ESFAS PLCs are tested to investigate their signal delay response characteristics and verify that their programming functions correctly. The determined response times are within the range acceptable for commercial safety systems (Hung, 2010). The next section presents the developed component PLCs for a representative PWR plant operation I&C system, which provides autonomous control of the developed PWR plant model.

3. Programmable Logic Controllers in a Representative PWR Plant

The operation I&C system of a representative PWR plant for implementation into the NICSim platform comprises several emulated PLCs for automatic control of the developed dynamic physics based PWR plant model. The PLCs in the plant operation I&C system regulate the reactor power, system pressure, pressurizer's water level, and the feedwater flow to the steam generators (Fig. 3.1). The PLCs of the plant operation I&C system architecture serve an integral role in the operation of the plant and thus represent potential targets for cyberattacks. The operation I&C system in nuclear power plants is less isolated than the plant protection and safety monitoring I&C system, with more network connections to other components. These may make the PLCs in this system more vulnerable to cyberattacks.

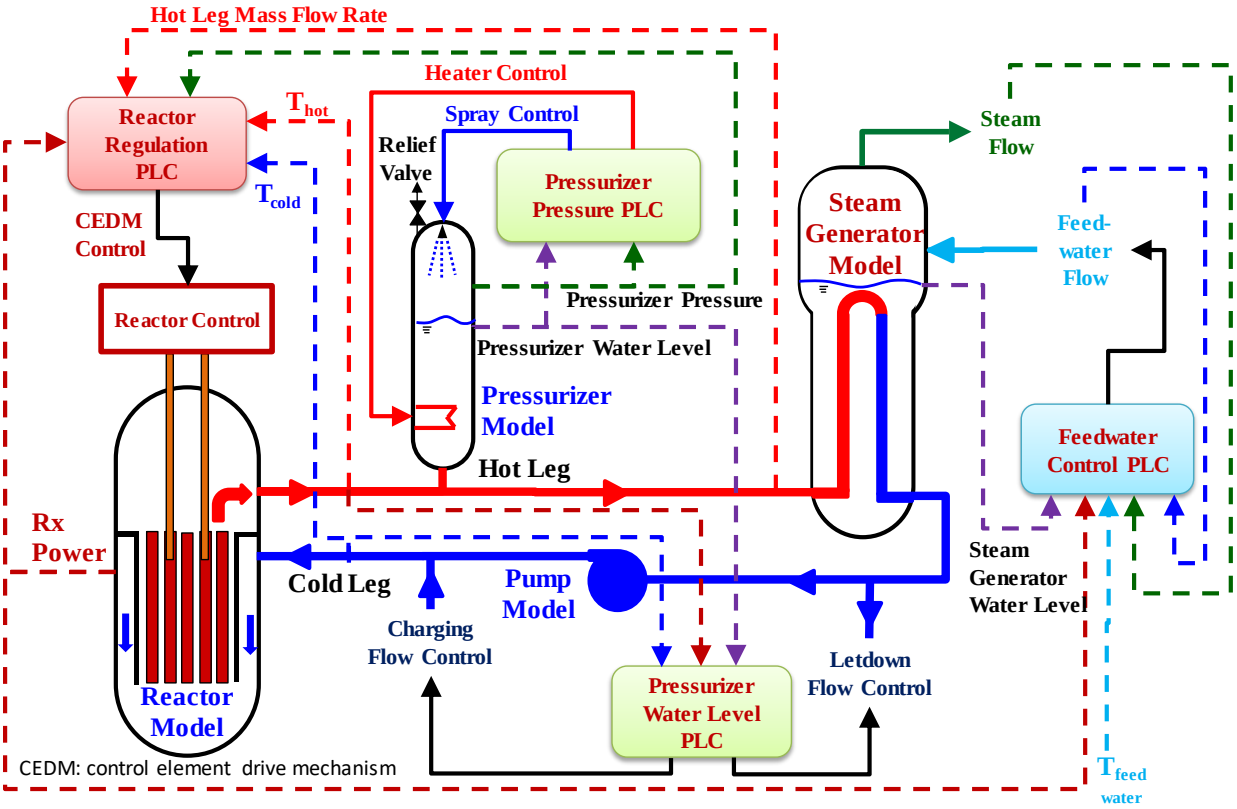

Fig. 3.1: Block diagram of the programmable logic controllers in the primary loop I&C system of a representative PWR plant (El-Genk et al., 2020a, 2020b, 2020c).

The representative PWR plant operation I&C system has five PLCs, namely: a reactor regulation PLC, a pressurizer pressure control PLC, pressurizer water level control PLC, steam generator feedwater control PLC, and pump control PLC. These PLCs receive state variables from the physics-based models and send signals back to the plant model for direct control feedback. Some of these PLCs function mostly independent of the reactor operators using pre-programed setpoint values, but others allow the operators in the control room to change setpoints to adjust the state of the plant during operation. The state variables calculated by the physics-based models of the plant components would be communicated to the PLCs in the emulated plant operation I&C system within the SCEPTRE framework. The data broker determines which state variables should be sent to the different PLCs. These state variables are analogous to the sensor instrument measurements received by the physical PLC's I/O modules in a real plant. The

PLCs respond to the communicated values of the state variables according to their programming and generate control signals to be transmitted back to the plant model to adjust operation.

The emulated PLCs are developed using OpenPLC software (Alves, et al., 2014) on Raspian VMs, like those used in the emulated PMS described in Section 2. The PLCs use the same basic setup but are configured for their different roles by changing the control program within the OpenPLC runtime. Each developed PLC in the plant operation I&C system is independently tested while being connected first to the associated component model, and subsequently to the integrated PWR plant model. Several of the plant operation I&C system PLCs use Proportional-Integral-Differential (PID) controllers whose functions depend on the system simulation time and require that the PLC time matches that of the simulation. To fulfill such a requirement, the developed data transfer interface program synchronizes the Simulink PWR plant model with a real clock time to ensure that the emulated PLCs and plant model run on the same time scales.

3.1 Reactor Regulation PLC

The reactor regulation PLC autonomously monitors the reactor thermal power, which is measured by multiple means, and compares it to a specified value by the operators. When the difference exceeds an allowed tolerance, the PLC indicates that corrective action may be required and sends a warning signal to the operators. Fig. 3.2 shows a functional block diagram of the programming PLC of the reactor regulation. The emulated PLC receives values of the reactor inlet and exit temperatures, T_{in} and T_{ex}, respectively, from the primary loop model, the system pressure, from the pressurizer model, p_{sys}, the total primary loop mass flow rate, $\dot{m}$, from the PWR primary loop model, and the reactor thermal power, P_{Rx}, calculated by the developed reactor model that couples reactor kinetics and thermal-hydraulics. The latter is representative of the reactor power determined by the calibrated power range nuclear instrumentation for the reactor (Palo Verde Nuclear Generating Station 2017).

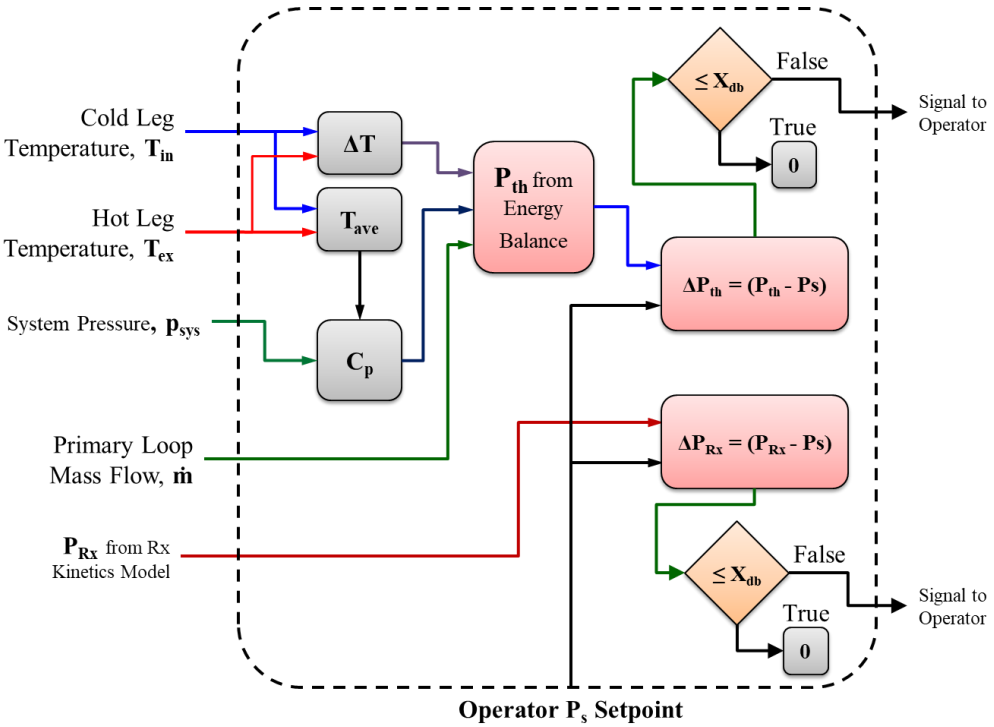

Fig. 3.2: A block diagram of control program for the reactor regulation PLC.

The PLC compares the reported value of P_{Rx} from the reactor model to the operator specified setpoint value, P_s, and passes the difference $\Delta P_{Rx} = (P_{Rx} - P_s)$ to a deadband filter. It sets ΔP_{Rx} to zero when it falls within the programmed upper and lower bands. When ΔP_{Rx} is outside the bounds of the deadband filter, the PLC sends a signal to the operators that warrant an active intervention to bring the reactor power back to within the desired operating condition. The PLC accommodates minor changes in ΔP_{Rx} due operation transients with active interferences.

The PLC additionally compares the value of the specified reactor power by the operator, Ps, to that calculated by the integrated primary loops model, P_{th}, based on the determined temperature rise across the reactor core $\Delta T = (T_{ex} - T_{in})$, and the coolant average specific heat, C_p, and the system pressure, p_{sys}. The difference in the reactor thermal power values, $\Delta P_{th} = (P_{th} - P_s)$ is passed through a deadband filter which inhibits action when this difference is within an allowed tolerance. On the other hand, when ΔP_{th} exceeds the range of the deadband filter a warning signal is transmitted to the operator.

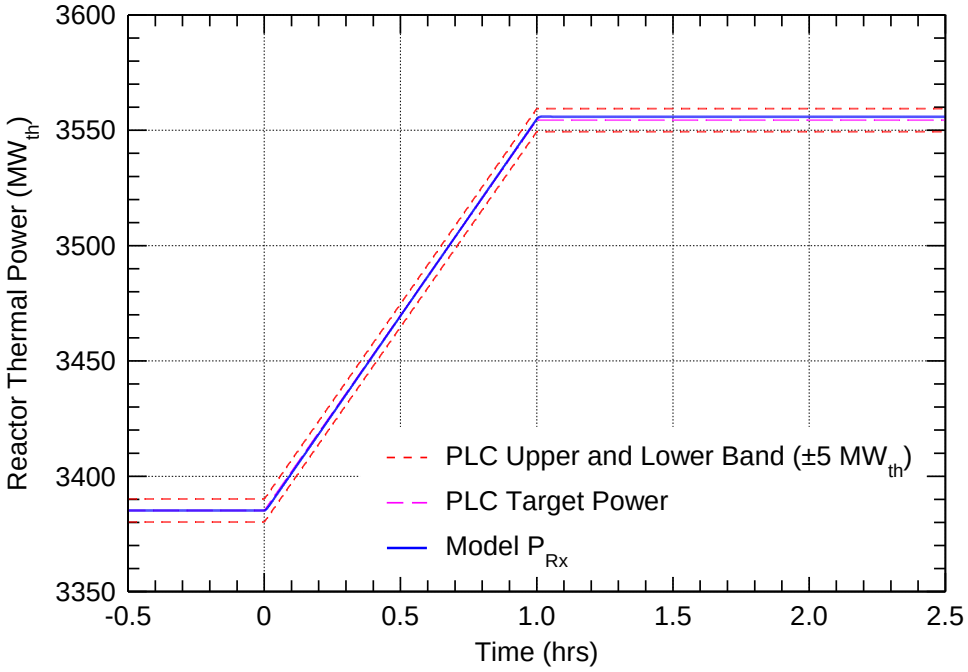

Fig. 3.3: Reactor power monitoring function of reactor regulation PLC following a 5% increase in load demand on the turbine in the secondary loop.

Figure 3.3 presents an example of the monitoring output from the PLC during a simulated transient of a 5% increase in load demand on the turbine in the secondary loop. In this example the deadband for the monitoring function of the reactor thermal power is set at ± 5 MW_{th}, or ~0.15% of the reactor full power. At time t = 0, the plant operates at steady state nominal conditions. The operation transient is initiated by increasing the steam demand 5% over a period of 1 hr. The PLC is set by the user to expect the increase in the reactor power. The reactor passively responds to the increase in the load demand by increasing the reactor thermal power due to its inherent positive temperature reactivity feedback until it eventually matches the increase in the load demand (Fig. 3.3). The reactor thermal power levels off at a 1.5 MW_{th} higher than the expected programed value. As this power difference is within the programed deadband tolerance of ± 5 MW_{th}, the PLC determines that no active reactor control intervention is

warranted and no warning signal is sent to the operator.

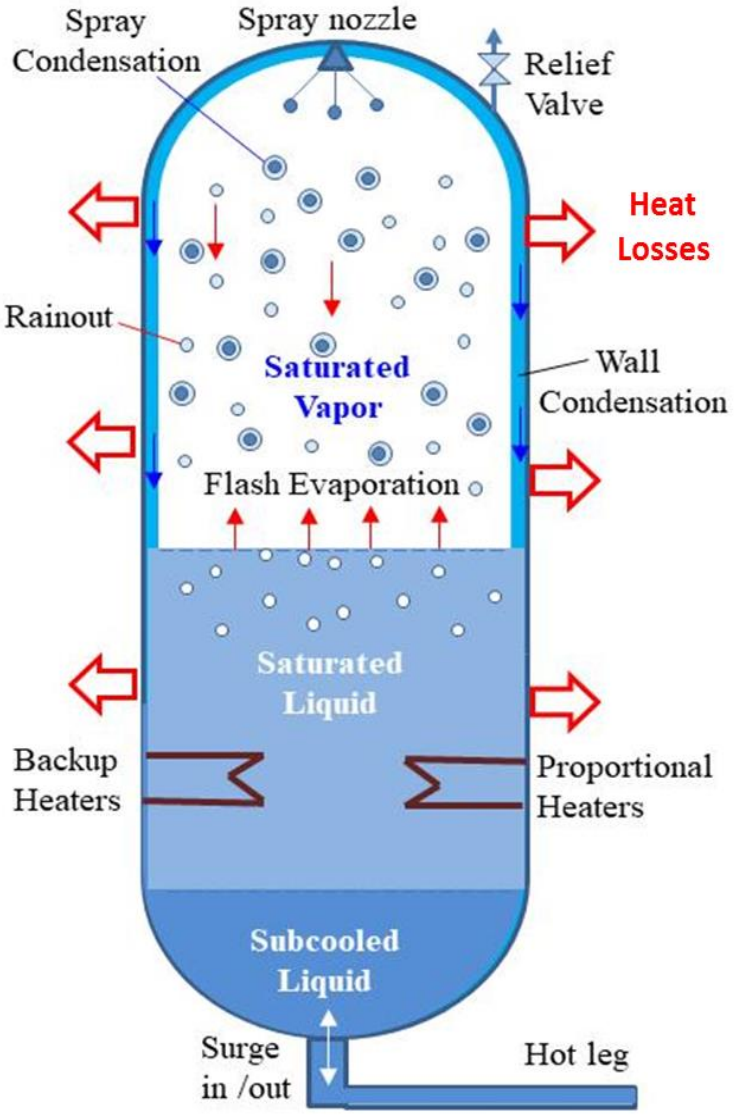

Fig. 3.4: An illustration of a PWR pressurizer with various regions and physical processes incorporated in the developed physics-based model of the pressurizer (El-Genk, Altamimi, Schreiner, 2020).

3.2 Pressurizer Pressure PLC

The pressure PLC regulates the pressure within the primary loop to stay within programed setpoints, by controlling the power supplied to the immersed proportional and backup heaters in the pressurizer and by opening or closing liquid spray nozzle (Fig. 3.4). This figure presents a sketch of pressurizer with illustrations of the different physical process associated with its functionality during operation transients causing a water surge in or surge out from and to the hot leg to the pressurizer, as a result of an over and under pressurization, respectively. The developed model divides the pressurizer volume intro three regions: a saturated vapor region at the top, a saturated liquid region in the middle, and a subcooled liquid region at the bottom (Fig. 3.4). The immersed proportional electrical heaters compensate for the heat losses through the pressurizer wall and partially with the backup heaters increase the system pressure by increasing the rate of

flash evaporation into the saturated liquid region of the pressurizer (Fig. 3.4). The liquid spray reduces the system pressure by condensing the vapor within the pressurizer's top region. The injected subcooled water from the cold leg of the primary loop through the spray nozzle breaks up into tiny droplets in the top region of the pressurize (Fig. 3.4). The saturated vapor in this region condenses onto the surface of the subcooled water droplets, reducing the vapor pressure and hence that within the primary loop (Fig. 3.4).

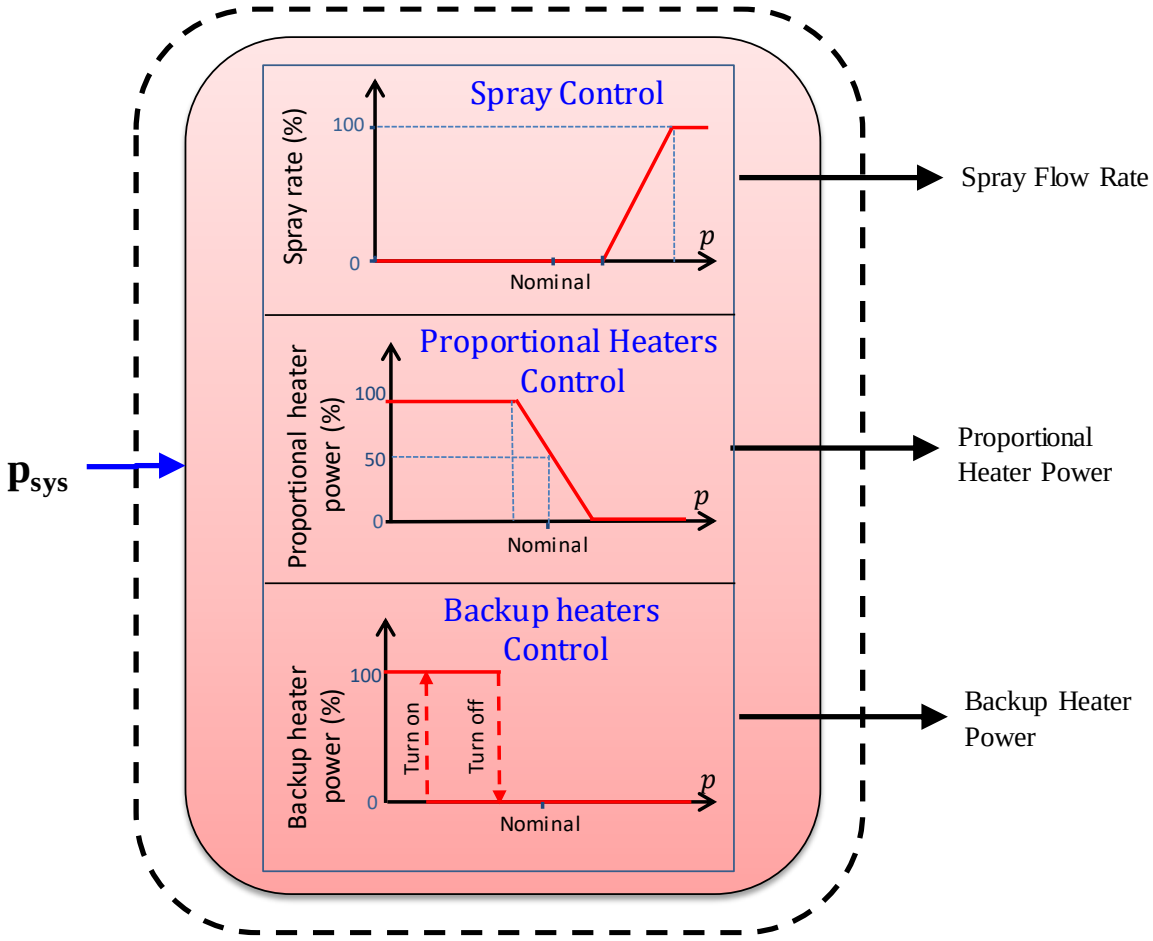

Fig. 3.5: Block Diagram of the Pressure PLC Control Program (Altamimi, El-Genk and Schriener, 2020; El-Genk et al., 2020b; El-Genk, Altamimi, Schreiner, 2020).

Figure 3.5 shows a functional block diagram of the control program for the pressurizer pressure PLC (El-Genk, Altamimi, Schreiner, 2020). The system pressure, p_{sys}, calculated by the developed pressurizer model is compared to the preprogramed setpoints for the three pressure control mechanisms, namely: the liquid spray, the immersed proportional electrical heaters, and immersed backup electrical heaters. It also compares the pressure to the high pressure setpoint for opening the pressure relief valve (Fig. 3.4) at the top of the pressurizer for venting excess vapor to help decrease the system pressure. The developed control program of this PLC has seven preprogramed pressure setpoints, for turning the proportional and the backup heaters ON and OFF and for opening and closing the liquid spray nozzle and pressure relief valve. These pressure setpoints may be configured differently for different PWR plant designs based on safety analysis results.

When the system pressure, p_{sys}, increases past the upper setpoint for the proportional heaters, the lower setpoint for the control of the liquid spray control will be exceeded and the control routine adjusts the opening of the spray nozzle (Fig. 3.5). The injected subcooled liquid droplets provide a large surface area for condensing the saturated vapor, which decreases the vapor volume in the upper region of the pressurizer and subsequently the system pressure.

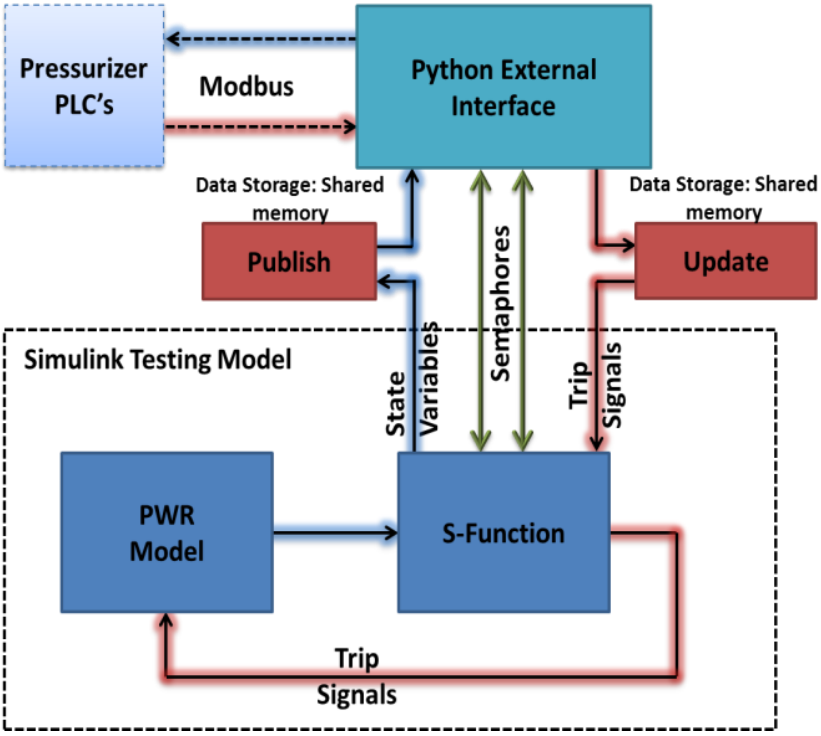

Fig. 3.6: Testing setup of linking and communicating the physics-based Simulink model of the pressurizer to the pressurizer's emulated PLCs.

The subcooled liquid spray nozzle control increases the flow rate of the water spray proportional to the increase in the pressure, from 0% with the nozzle closed at its lower pressure set point, to 100% with the valve is fully open at the upper pressure setpoint. When p_{sys} decreases below the lower setpoint for the proportional heaters controller, the backup heaters switch ON to increase flash evaporation into the top saturated vapor region of the pressurizer (Fig. 3.4) and raise the system pressure. Unlike the proportional heaters or the liquid spray nozzle, the backup heaters have only two settings of fully ON and full OFF (Fig. 3.5). The ON setting corresponds to a lower pressure setpoint than the OFF setting. When backup heaters turn ON, they stay ON until the upper pressure setpoint is reached, turning them OFF.

3.2.1 Response Time Testing Results for Pressurizer Pressure PLC

The control functions of the developed emulated pressure PLC to adjust and regulate the system pressure are tested and the response time delay of the PLC linked to the physics-based pressurizer model is investigated. To determine the response delay time, the response of the emulated PLC is compared to that of an 'ideal' controller in the pressurizer's Simulink model during a simulated operation transient (El-Genk, Altamimi, Schreiner, 2020). The 'ideal' controller implemented within the Simulink physics-based pressurizer model uses the exact same control logic as the emulated PLC. The ideal Simulink controller represents a PLC with instantaneous response or zero-response delay. Therefore, it does not require signal

communication to and from the emulated PLC. The delay in the response time is determined from the difference in response of the emulated PLC and ideal controller within Simulink.

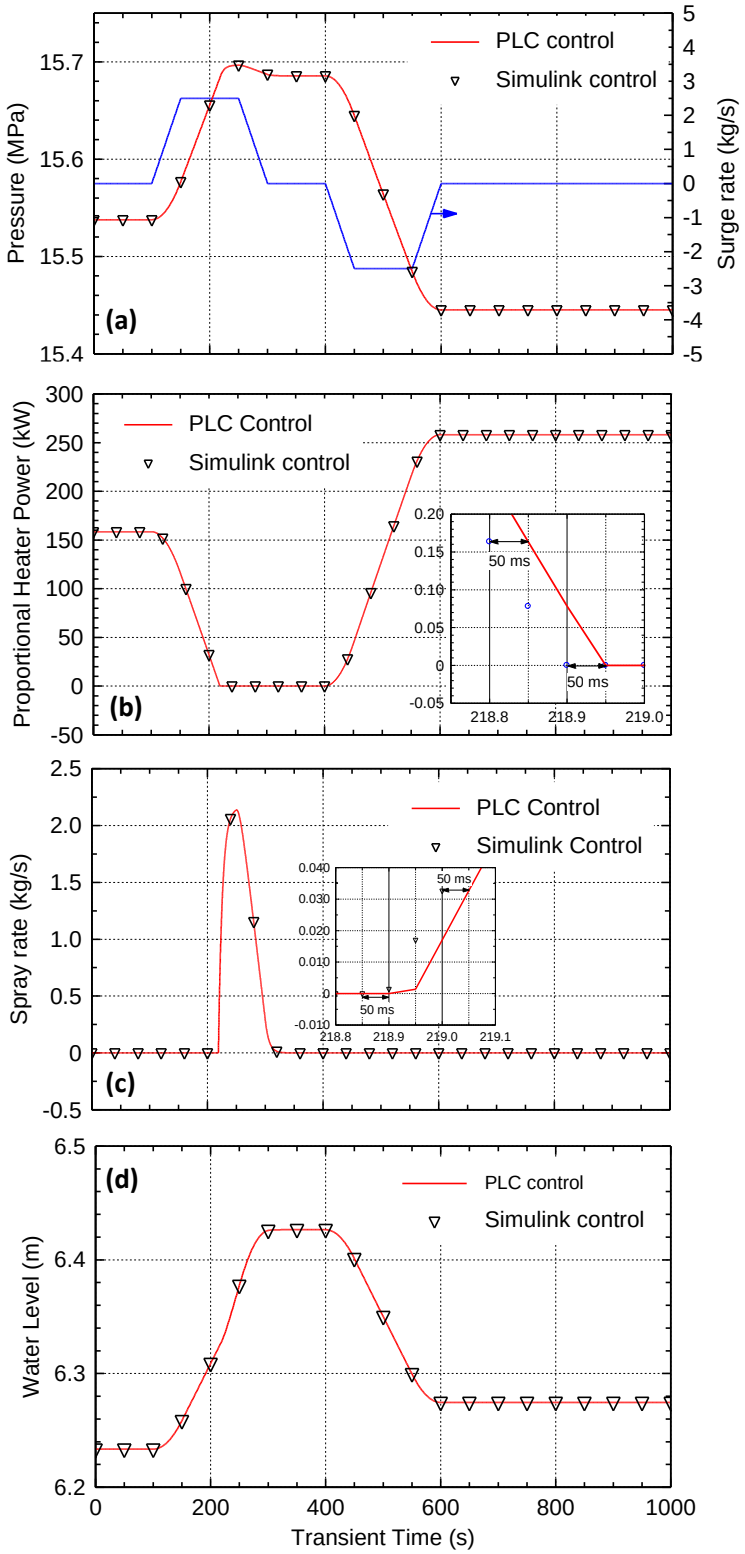

Fig. 3.7: Simulation results of pressurizer surge-in and surge-out transients.

Figure 3.6 shows the present testing setup used to investigate the responses of the emulated pressure PLC. It runs within a VM on a Windows PC networked to the Linux server running the Simulink pressurizer model using a developed data transfer interface program. The data transfer interface communicates directly with the OpenPLC runtime on the emulated PLC using Modbus over TCP. To test the PLC response time delay, the pressurizer model runs separate from the remainder of the PWR primary loop model. Instead, values of the coolant temperatures hot and cold legs and the pressurizer's surge rate are specified as functions of time during the simulated transient. The emulated pressure control PLC in this test operates at a sampling frequency, $\varphi = 20$ Hz (or 20 samples per second) (El-Genk, Altamimi, Schreiner, 2020).

The dynamic pressurizer model simulates transients involving an initial surge in of water from the hot leg to the pressurizer, followed by a subsequent surge out of water back to the primary loop (El-Genk, Altamimi, Schreiner, 2020). Fig. 3.7a-d shows the simulation results of the linked pressurizer model to the emulated PLC and compares the performance results to those of the ideal controller in the Simulink model. The transient begins at t = 100s with a surge in of water at a constant rate from the hot leg into the pressurizer over a period of 200 s (Fig. 3.7a). The surge in raises the internal water level in the pressurizer (Fig. 3.7d), compressing the vapor region and increasing system pressure (Fig. 3.7a).

When the pressure reaches the upper pressure setpoint for the proportional heaters (Fig. 3.7b), the controller turns these heaters OFF to stop further water evaporation from the saturated liquid region into the saturated vapor region of the pressurizer. However, as the water surge in continues, the pressure continues to increase until reaching the lower pressure setpoint for opening the valve for the liquid spray system. It injects water droplets into the saturated vapor region of the pressurizer to condense the vapor. The subcooled water flow to the liquid spray system comes from the cold leg of the reactor primary loop. The spray rate increases proportionally to the increased pressure (Fig. 3.5). The water spray stimulates condensation, which limits the increase in pressure caused by the surge in from the hot leg. When the surge in phase of the transient ends, the pressure reaches a new equilibrium value that remains steady until the subsequent surge out phase starts (Fig. 3.7b). The spray valve closes when the pressure reaches or drops below low pressure setpoint of the value (Fig. 3.7c).

At t = 400 s into the transient, water surges out from the pressurizer back into the surge line of the hot leg, at a constant rate that equals that of the early surge in (Fig. 3.7a). The surge out continues for 200 s, during which both the pressure and water level in the pressurizer decrease (Figs. 3.7a, d). The pressurizer PLC turns ON the immersed proportional electrical heaters when the pressure decreases below their actuation setpoint. The thermal power dissipated by the proportional heaters increases inversely proportional to the decrease in the system pressure (Fig. 3.7b). When the surge out phase ends, the system pressure and the water level in the pressurizer reach steady state, but their values are higher and lower, respectively, at the start of the simulated transients (Figs. 3.7a,d).

The simulation results using the emulated PLC and those using the 'ideal' Simulink controller show excellent agreement (Fig. 3.7). The inserts in Figs. 3.7b and 3.7c compares the response delay time of the emulated PLC to the internal Simulink control logic. For both the proportional heaters and water spray, the emulated PLC responds with a 50 ms delay, compared to Simulink model internal controller, responds with a 50 ms delay. This delay has only a minor effect on the simulated transient results as indicated by the close agreement of the calculated values of the pressure (Fig. 3.7a) and the water (Fig. 3.7d) in the pressurizer throughout the simulated transient. There also close agreement of the results for controlling the immersed

proportional electrical heaters and the rate of the water spray by the emulated PLC and internal controller in the Simulink model of the pressurizer (Figs. 3.7b,c).

3.3 Pressurizer Water Level PLC

The second PLC connected to the pressurizer model controls the water level in the pressurizer by regulating the water total volume in the primary loop (El-Genk, Altamimi, Schreiner, 2020). The water level PLC acts independently of the pressure PLC. Working independently, both PLCs help control the system pressure in the primary loop. The control program in the water level PLC accommodates changes in the water volume in the primary loop due to thermal expansion and contraction during a heat up or a cooldown of the primary loop and helps maintain the primary coolant inventory in the event of a leak. The PLC adjusts the water level in the pressurizer by controlling the rates of the coolant inflow to the primary loop from the charging pumps and the coolant outflow through the letdown valves.

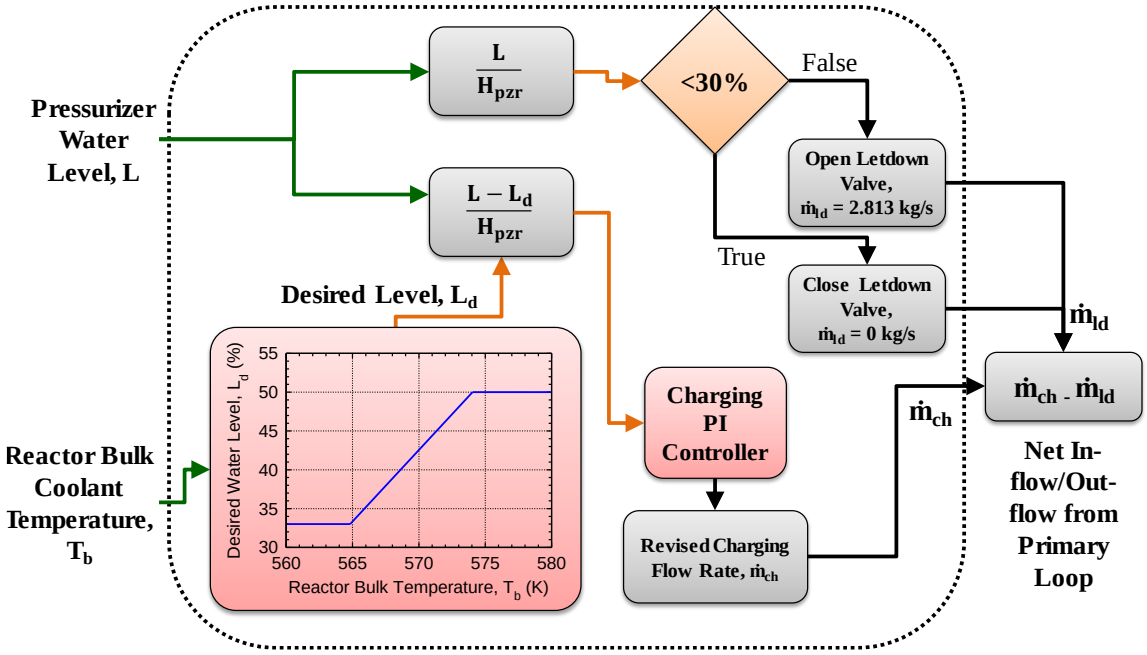

Fig. 3.8: A block diagram of the control program of the pressurizer's water level PLC (El-Genk et al., 2020b; El-Genk, Altamimi, Schreiner, 2020).

Figure 3.8 presents a block diagram of the control program for the pressurizer's water level PLC. It compares the determined water level, L, by the pressurizer model to the desired water level, L_d, calculated by the PLC. The control program determines the value of L_d as a function of the bulk coolant temperature in the reactor to help accommodate the thermal expansion of the primary coolant during startup. During this period, a heat up of the primary coolant results in thermal expansion, which increases the pressurizer water level. Increasing the value of L_d reduces the required adjustments to the charging rate (Fig. 3.8).

The difference between the actual and the desired water level, normalized to the height of the pressurizer, H_{pzr}, $(L-L_d)/H_{pzr}$, is passed to a Proportional-Integral (PI) controller which adjust the charging flow rate by the charging pumps into the primary loop (Figs. 3.1 and 3.8). The developed PI controller uses the following formulation:

$$\dot{m}_{ch}(t) = P \times \left(\frac{L-L_d}{H_{pzr}}\right) + I \int_0^t \left(\frac{L-L_d}{H_{pzr}}\right) dt \qquad (3.1)$$

In this expression, $\dot{m}_{ch}$ is the flow rate by the charging pump, and P and I are the proportional and integral gain constants for the controller. The control of the letdown flow rate, $\dot{m}_{ld}$, uses the water level normalized to the height of the pressurizer, L/H_{pzr}, as its control parameter. This value is compared to a low water level setpoint of 30%, and if equal or above the setpoint the letdown valve is kept open. This valve closes if the value of L/H_{pzr} decreases below the setpoint.

The difference between the charging and the letdown flow rate represents the net inflow or outflow of coolant in the primary loop, assuming no leakage. If the water in the pressurizer drops below the desired level, the water level PLC program adjusts the charging rate so the net inflow into the primary is positive, which increases the primary loop coolant inventory. When the pressurizer water level is above the desired level, the charging rate decreases below the letdown rate, resulting in a net outflow of the coolant from the primary loop.

3.3.1 Tuning of PI Controller Constants for Pressurizer Water Level PLC

This section presents the test results to determine the values of the proportional and integral constants, P and I, for the PI controller of the pressurizer water level PLC for smooth response by avoiding sharp changes in the water level during an operation transient. Results are presented for the PLC during a simulated reactor startup sequence using the developed physics-based model of the integrated PWR plant. The representative PWR plant design and startup sequence used in the simulation are detailed elsewhere (El-Genk et al., 2020a). The charging flow rate varied from 0-4.5 kg/s, with a constant letdown rate of 2.813 kg/s.

Figures 3.9a-b show the calculated results during the simulated reactor startup. Figs. 3.10 and 3.11 show the changes in the associated state variables for the pressurizer and primary loop. During the simulated startup scenario, the reactor power increased in stages up to the full nominal value by inserting external reactivity. This is accomplished by withdrawing the control rod assemblies in the reactor core and reducing the concentration of soluble boron in the coolant (Fig. 3.9a,b). Initially the boron concentration is kept constant, while withdrawing the control rod assemblies to increase the reactor thermal power to 5%, 15%, and subsequently 20% of full power. Subsequent external reactivity insertion is accomplished by reducing the boron concentration in the water coolant in the primary loops from 1,321 ppm to 1,160 ppm in stages (Fig. 3.9a), to further increase the reactor thermal power to 50%, 90%, and finally 100% of its nominal value (Fig. 3.9b).

As the reactor power increases, so does the temperature of the water coolant that is exiting the reactor into the hot legs (Fig. 3.9b). This also increases the coolant temperature flowing through in the U-tube bundles in the steam generators, which in turns increases the steam generation rate, and the rate of heat removed from the primary loop coolant (Fig. 3.9c), which decreases the temperature of the returning water in the cold legs to the reactor (Fig. 3.9b). The increase in the reactor thermal power also increases the bulk coolant temperature (Fig. 3.9b) and volume in the primary loop, which initiates a surge-in of water to the pressurizer (Fig. 3.10a).

The surge-in increases the water level in the pressurizer as well as the system pressure (Figs. 3.10a and c). The water level PLC adjusts the charging rate of the primary loop in response to maintain the controller value $(L-L_d)/H_{pzr}$ at or near zero (Figs. 3.10b-d). These results are for two different negative constants for two PI controllers of the water level PLC; one controller has P = -280 and I = -5, and the other has P = -2800 and I = -50. Increasing the negative magnitude of the P and I constants speeds up the response of the PI controller and the water charging rate into the primary loop. Such increases, however, cause the PI controller to experience an oscillating response as it attempts to minimize the parameter $(L-L_d)/H_{pzr}$.

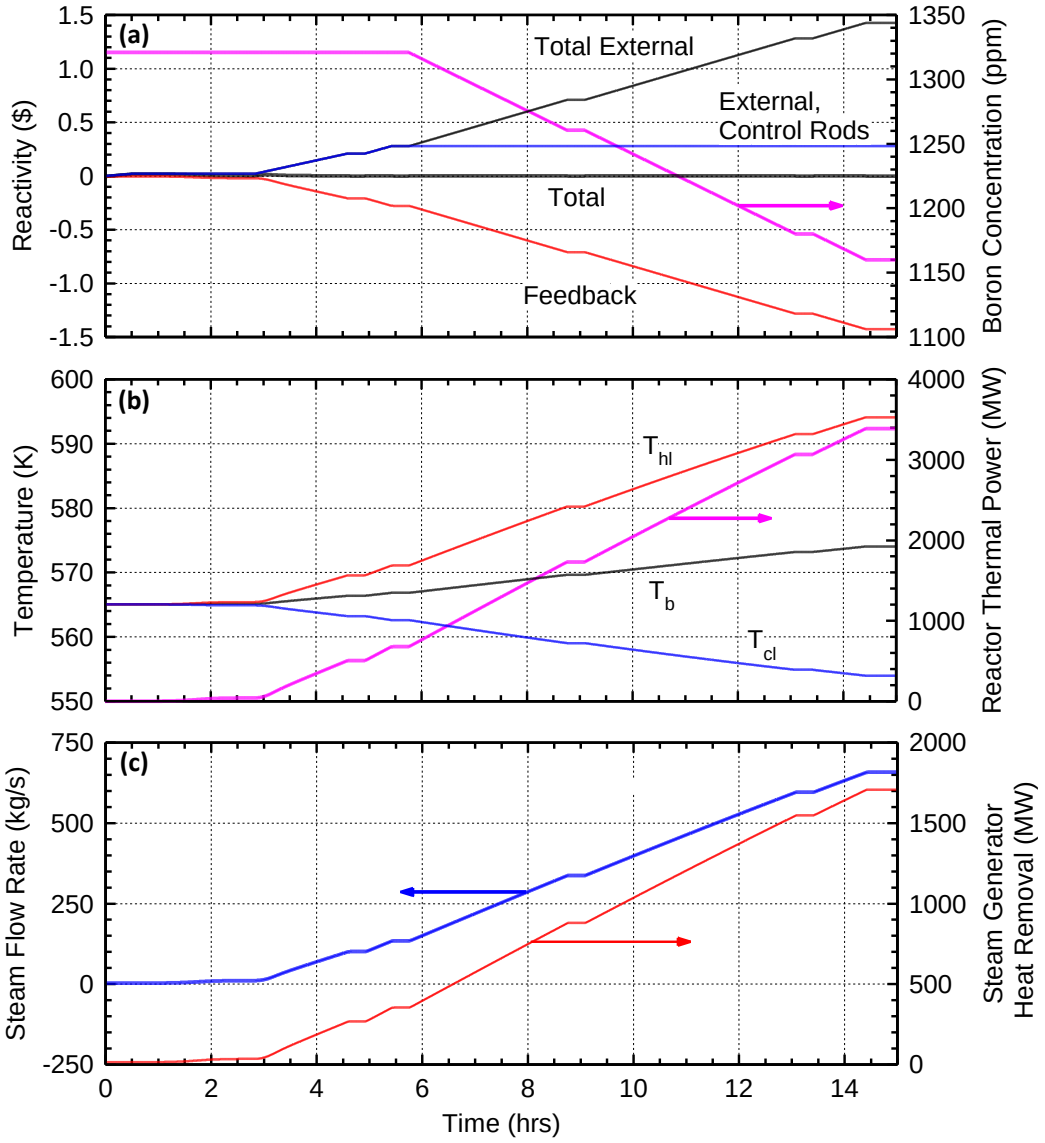

Fig. 3.9: Calculated changes in total and feedback reactivity and reactor's thermal power, coolant temperatures, and steam generation rate in a simulated startup transient to test the emulated water level PLC for pressurizer in a representative PWR plant (El-Genk, Altamimi, Schreiner, 2020).

During the simulated startup transient, the expansion of the heating up coolant in the primary loops causes a surge-in of water from the hot leg into the pressurizer (Fig. 11a), which increases the system pressure (Fig. 3.10a). The water level PLC adjusts the charging rate of water into the primary loop (Fig. 3.10c) to decrease the dimensional parameter $(L\text{-}L_d)/H_{pzr}$ (Fig. 3.10b). During the surge-in, the pressure PLC activates the subcooled water spray to reduce the increase in the system (Fig. 3.11b). This PLC turns ON the submerged proportional heaters in the pressurizer when the pressure drops below the upper setpoint for turning on these heaters (Fig. 3.10b). The PI controller constants for the water level PLC insignificantly affected the values of the system pressure and the surge-in rate during the simulated startup transient (Figs. 3.10a and 3.11a). The results also show that the PI controller constants P = -2800 and I = -50 produce a smaller deviation in the control parameter $(L\text{-}L_d)/H_{pzr}$ during the simulated transient (Fig. 3.10b). Therefore, they are selected for the PI controller of the emulated water level PLC.

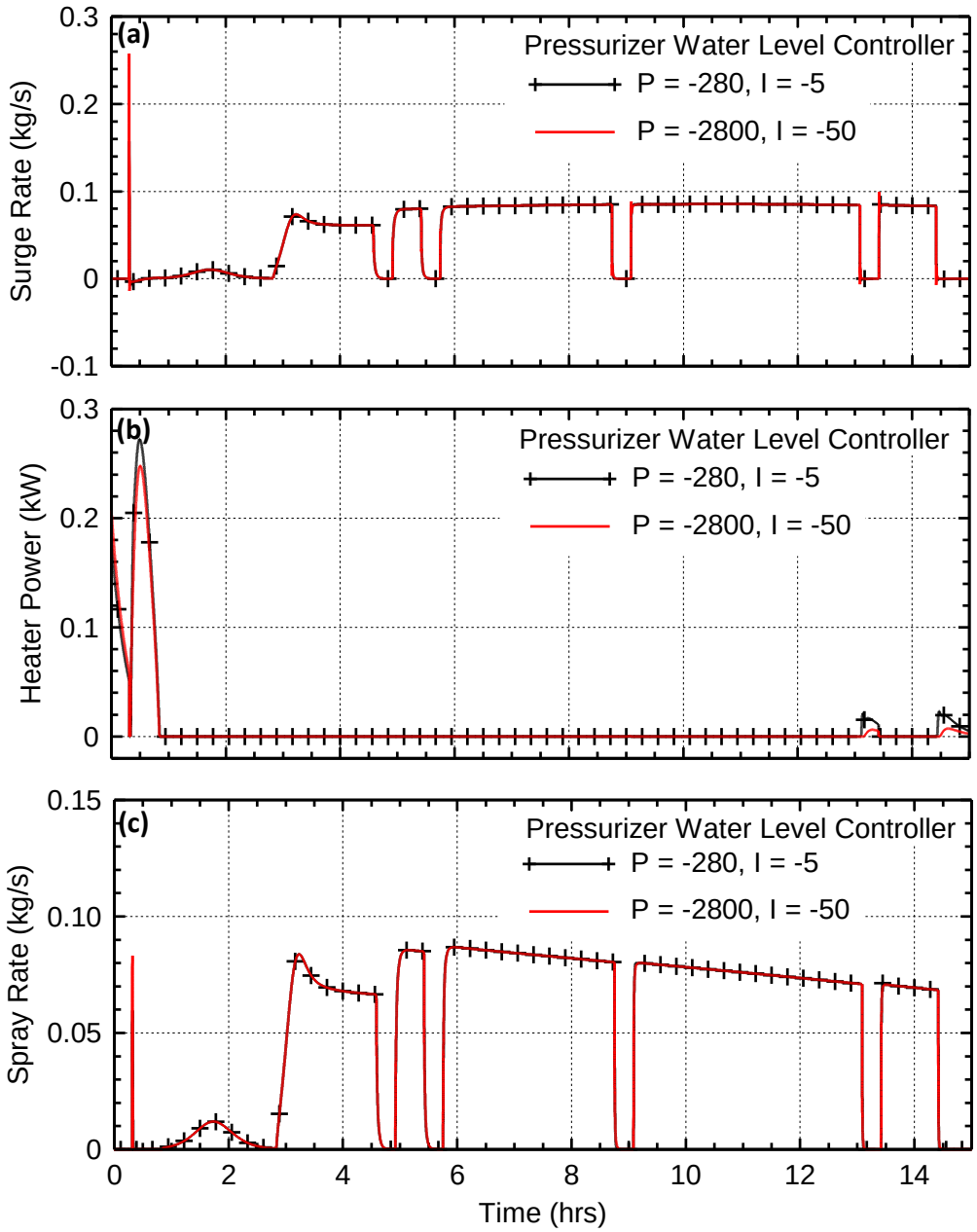

Fig. 3.10: Effect of PI controller constants for the pressurizer water level PLC on state variables of the pressurizer surge rate, heaters powers, and spray rate during a simulated startup sequence of the reactor in a representative PWR plant (El-Genk, Altamimi, Schreiner, 2020).

3.4 Steam Generator Feedwater Control PLC

The feedwater control PLC adjust and regulate the water inventory in the plant's steam generators to ensure that the U-tube bundles are adequately covered (Schriener and EL-Genk, 2020). It monitors the measured water level in the downcomer of the steam generator and adjusts the feedwater rate injected into the secondary, or shell, side of the steam generator. The feedwater rate in a representative PWR is adjusted using the throttle valve between the feedwater pumps and the feedwater injection ring in the steam generator. The feedwater control PLC adjusts the opening of the throttle valve to change the hydraulic resistance for increasing or decreasing the feedwater flow rate returning to the steam generator.

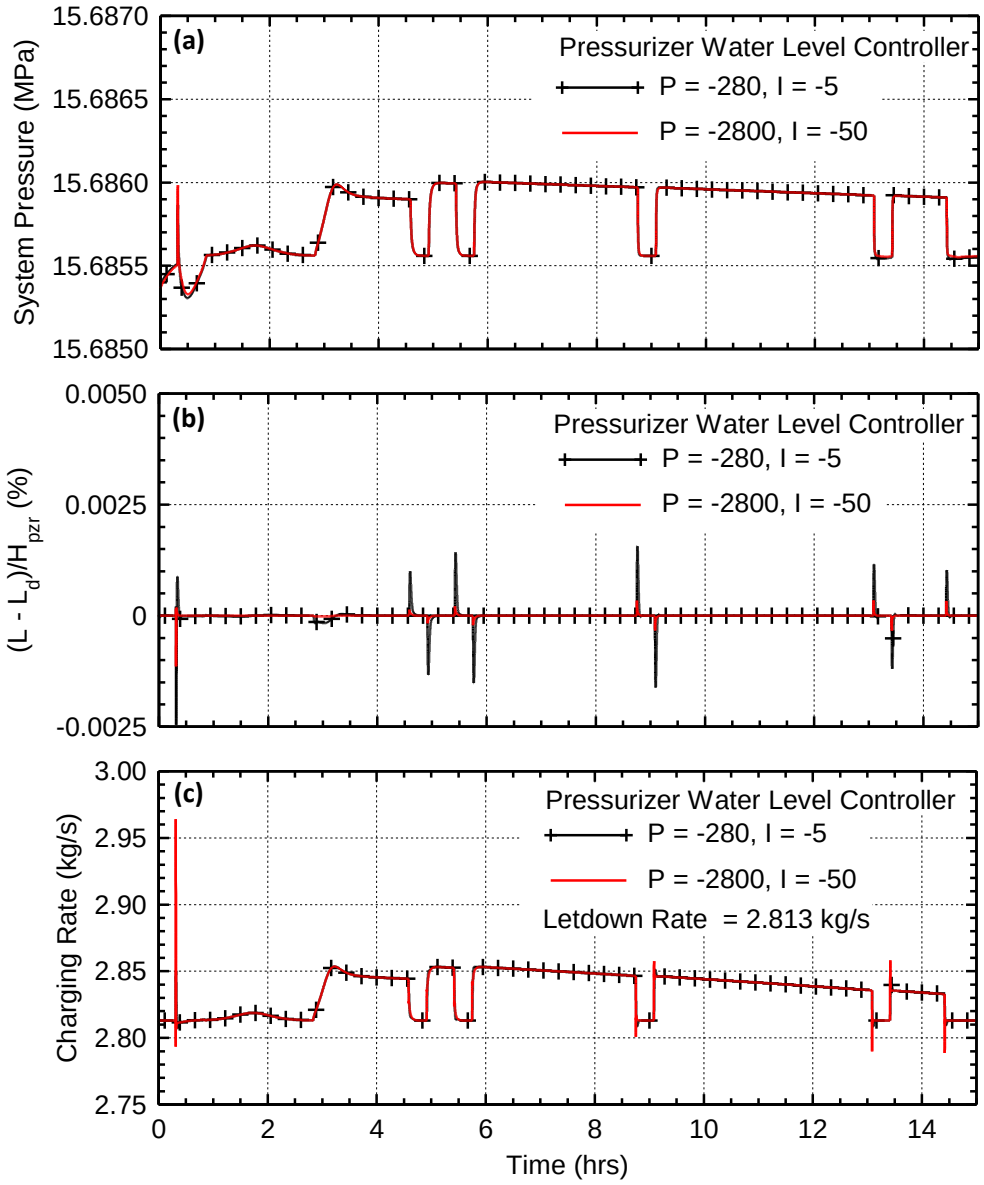

Fig. 3.11: Effect of PI controller constants for the pressurizer's water level PLC on the values of system pressure, PI control variable, and charging rate during a simulated startup transient of a representative PWR plant (El-Genk, Altamimi, Schreiner, 2020).

The throttle valve position is based on a control parameter, C*, which combines two comparisons of the state variables (Fig. 3.11). The first is of the difference between the water level in the annular downcomer of the steam generator L, and the desired water level, L_d, calculated by the PLC as a function of the reactor thermal power, P_{Rx}. The second comparison is of the difference between the flow rate of the steam exiting the steam generator, $\dot{m}_s$, and the feed water flow rate, $\dot{m}_{fw}$ to the steam generator. The difference between the water level in the downcomer and the desired level in the steam generator is normalized to the maximum water level in the steam generator as: $(L_d - L)/L_{max}$. Similarly, the difference between the steam and feedwater flow rates for the steam generator is normalized to the maximum feedwater flow rate as: $(\dot{m}_s - \dot{m}_{fw})/\dot{m}_{fw,max}$. These two normalized quantities are incorporated into the combined control parameter C* (Fig. 3.11). It ensures that the feedwater controller PLC simultaneously

considers both the differences in the water levels and the flow rates when adjusting the feedwater flow rate to the steam generator. Such consideration minimizes or dampens the changes in the water level and feedwater rate during operational transients.

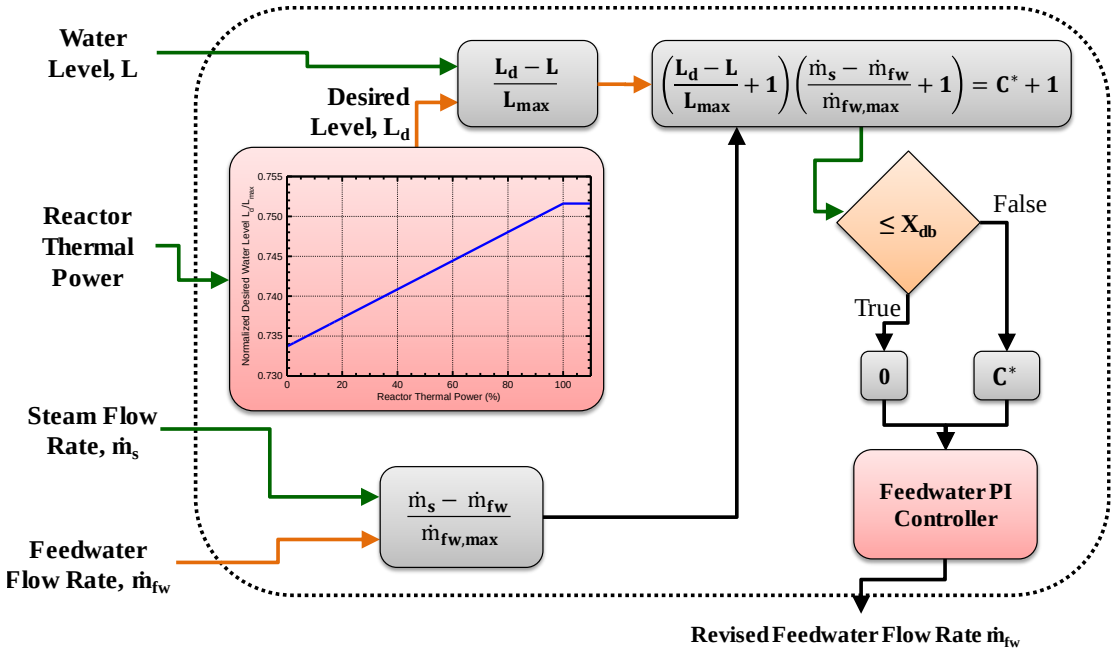

Fig. 3.12: A schematic of the control program for the steam generator feedwater control PLC (El-Genk et al., 2020a).

Deadband filter sets the calculated value of the control parameter C* to zero when it is within the preprogramed value, X_{db}, and inhibits control action when the differences between the actual and desired water levels, and between the steam and feedwater flow rates are small. This reduces the frequency of adjusting the throttle valve opening during an operation transient of the plant, e.g., following a change in the electrical load demand. The output from the deadband filter is passed to a Proportional-Integral (PI) controller to set the new throttle valve position for adjusting the feedwater flow rate to the steam generator, as:

$$\dot{m}_{fw}(t) = P \times (C^*) + I \int_0^t (C^*)\, dt \qquad (3.2)$$

In this expression, P is the proportional gain constant and I is the integral gain constant for the PI controller of the feed water PLC.

3.4.1 Response Time Testing Results for Feedwater Control PLC

A test is also performed to examine the response characteristics of the emulated PLC for controlling feedwater control to steam generator. In the performed test, the emulated PLC is linked to the developed steam generator physics-based dynamic model. It investigated the effects of the response time, sampling rate, and timestep on the dynamic response of the coupled PLC with the steam generator model during simulated transient. The steam generator model and feedwater control PLC are tested for an operation transient involving a 10% increase in the load demand (Fig. 3.13) (Schriener and El-Genk 2020). The response of the PLC is compared to that of an 'ideal' controller in the internal control logic built in the Simulink model. The PLC control program and the ideal controller within the Simulink steam generator model use the same logic. However, unlike the PLC controller, the ideal controller in Simulink has no delay in response.

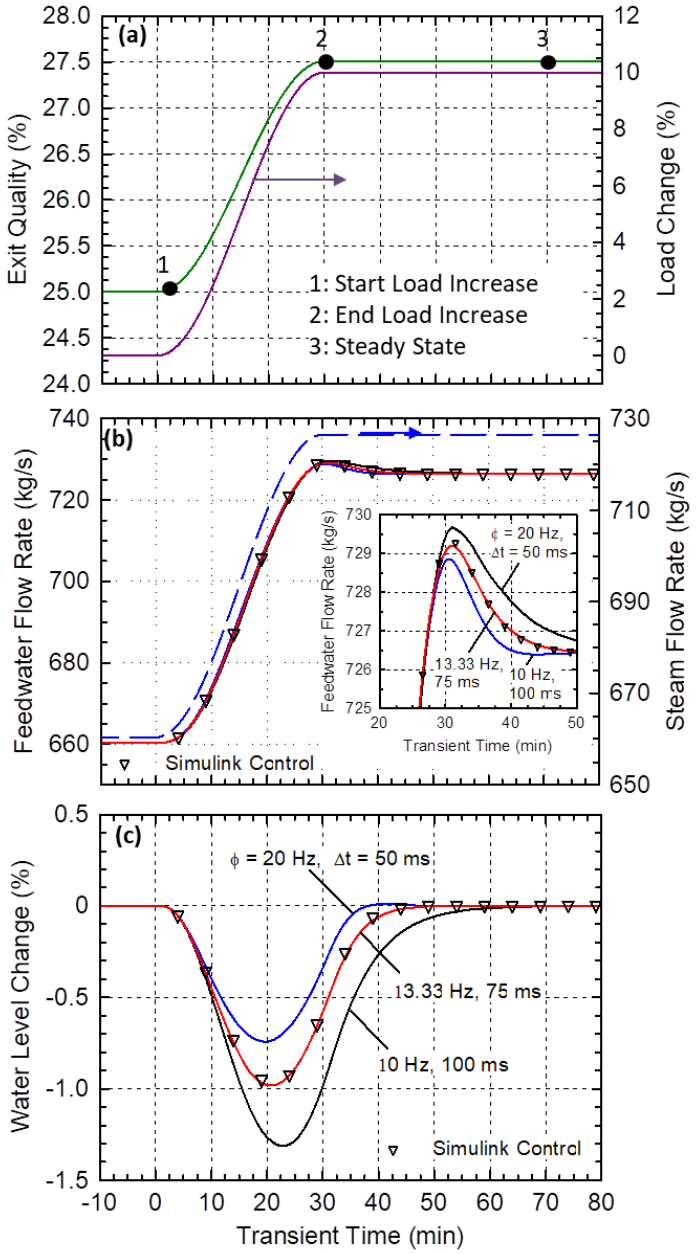

Fig. 3.13: Test results of the steam generator's feedwater control PLC in a simulated transient following a 10% increase in load demand (El-Genk et al., 2020a).

In the performed test of the feedwater PLC, the steam generator model is not coupled to that of a representative PWR plant model, and the hot leg coolant temperature, pressure, and flow rate are held constant. In the simulated transient, the steam generator is initially at nominal full power condition and the reactor is at its nominal thermal power of 3,400 MW_{th}. The primary coolant to the steam generator is held at a constant inlet temperature of 594 K and flow rate of 7,585 kg/s. The feedwater PLC is configured with proportional and integral constants of 4.196 and 0.0119, respectively, for the PI controller and with no deadband filter (X_{db} = 0). The feed water flow rate, $\dot{m}_{ch}$, in this test varied between 0.1 - 943.72 kg/s. Fig. 3.13a-c presents the obtained results for the simulated transient following a 10% increase in the electrical / or steam

load demand. The sampling frequency, φ, and the transient simulation time step, Δt, are varied to investigate the effects on the emulated PLC control response. The results also help determine the settings for the response of the emulated PLC to match that of the internal Simulink controller.

The increase in the load demand increases the flow rate and quality of the steam exiting the steam generator (Figs. 3.13a,b). When the steam flow rate exceeds that of the feedwater injection, both the water inventory and the level in the steam generator downcomer decrease (Fig. 3.13c). The feedwater control PLC responds to the deceasing water level by adjusting the opening of the throttle valve in the secondary loop (Fig. 2.1) to increase the feedwater rate to the steam generator (Fig. 3.13b). This gradually slows the decrease of the water level in the steam generator, and eventually returns it to the programmed desired setpoint.

Due to its integrator component, the PI controller for the feedwater control PLC is sensitive to the timestep size, Δt, in the simulated transient. The sampling rate, φ, and simulation time step size are varied to determine the setting of the emulated PLC for its response time to match that of the ideal controller logic in Simulink. The insert in Fig. 3.13b and the water level results in Fig. 3.13c show that increasing φ and / or decreasing Δt results in a faster response of the emulated water level PLC for changing in the water level in the steam generator. With a φ = 13.33 Hz and Δt = 75 ms, the response of the emulated PLC nearly matches that of the ideal Simulink controller, with no response delay.

3.4.2 Tuning PI Controller Constants for the Emulated Feedwater Control PLC

Additional testing is also performed to tune the values of the proportional and integral constants, P, and I, of the PI controller for the emulated PLC of the feedwater control for the steam generator. Proper constants would reduce the magnitude of the transient variations in the values of the dimensionless quantiles of $(\dot{m}_s - \dot{m}_{fw})/\dot{m}_{fw,max}$ and $(L_d - L)/L_{max}$ during the simulated operation transients. Fig. 3.14 and 3.15 present the obtained PLC results during a simulated reactor startup using the developed integrated model of a representative PWR plant (El-Genk et al., 2020b). Figs. 3.14a-b show the values of the various plant state variables calculated by the reactor and primary loop models during the simulated startup. In the simulated startup transient, the reactor thermal power increases in sequential stages from hot zero-power hot condition to 2%, 15%, 20%, 50%, 90%, and finally 100% of nominal full power (Fig. 3.13b) (El-Genk et al., 2020a).

In the simulated startup transient, the reactor thermal power increased in stages to 5%, 15%, and 20% of full power due to the inserted external reactivity by withdrawing the control rod assemblies in the reactor core (Fig. 3.14a,b). Subsequent increases in the external reactivity by decreasing the boron concentration in the water coolant in the primary loops from 1,321 ppm to 1,160 ppm (Fig. 3.14a), increased the reactor power in stages to 50%, 90%, and finally 100% of its nominal value (Fig. 3.14b). The increases in the reactor power and the temperature of the coolant entering the U-tubes of the steam generators increase the rate of heat transfer to the water on the shell side of the steam generator U-tubes, which increase the steam generation and flow rate to the turbine (Fig. 3.14c). The results in Figs. 3.15a-c show the effect of changing the values of the controller constants P and I for the water level PLC for the steam generator. They also show the changes in the normalized difference of the water level, $(L_d - L)/L_{max}$ in the steam generator, and the normalized difference between the steam generation and the feed water flow rates, $(\dot{m}_s - \dot{m}_{fw})/\dot{m}_{fw,max}$.

The PI controller with the smallest constants (P = 0.002, I = 0.06) experiences the lowest swings in the steam generator water level (Fig. 3.15a), but the largest swings in the two control

parameters during the simulated reactor startup transient. The normalized difference in the water level overshoots the desired level during the periods when the reactor power is increasing, forcing sharp shifts down after the thermal power level drops off (Figs. 3.14b, 3.15b). These small controller constants also result in the largest swings in the normalized difference between the steam generation and the feedwater flow rates (Fig. 3.15c).

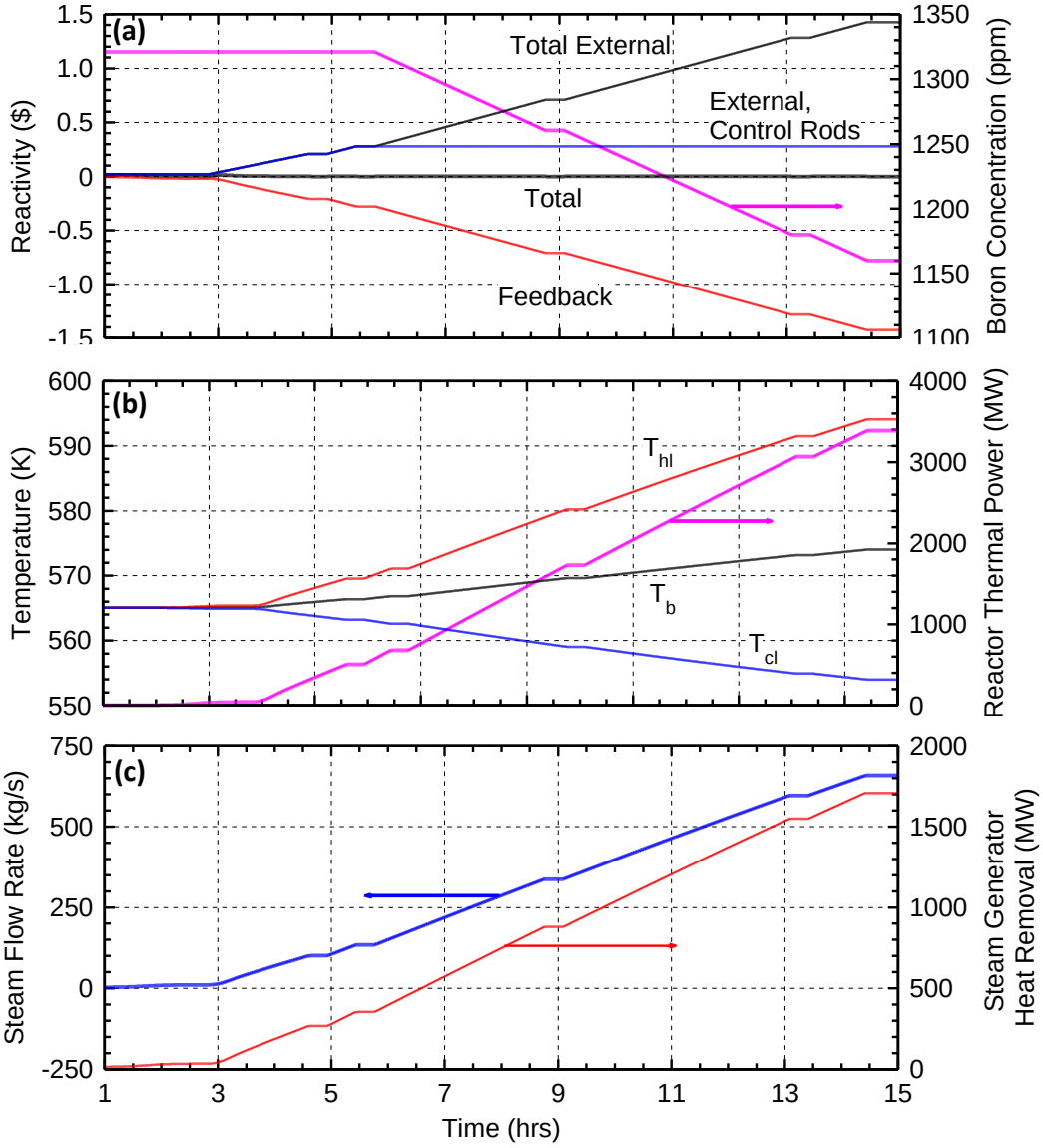

Fig. 3.14: Calculated changes in total and feedback reactivity and reactor's thermal power, coolant temperatures, and steam generation rate during a simulated startup transient for testing the emulated PLC of the steam generator's feedwater control (El-Genk et al., 2020a).

Increasing the values of the controller constants to P = 0.02 and I = 0.6 results in the smallest differences in the two control parameters on average. Further increase of the magnitude of the controller PI constants (P = 0.2, I = 6.0) only slightly effects the results, but causes a small increase of the differences in the water level in the steam generator and between the rates of steam production and the feedwater injection (Figs. 3.15b-c). Based on these analyses results, the PI controller constants P = 0.02 and I = 0.6 and most suitable for recusing the differences during

the simulated startup transient, and therefore, are used for use in the controller of the emulated feedwater control PLC for the steam generator.

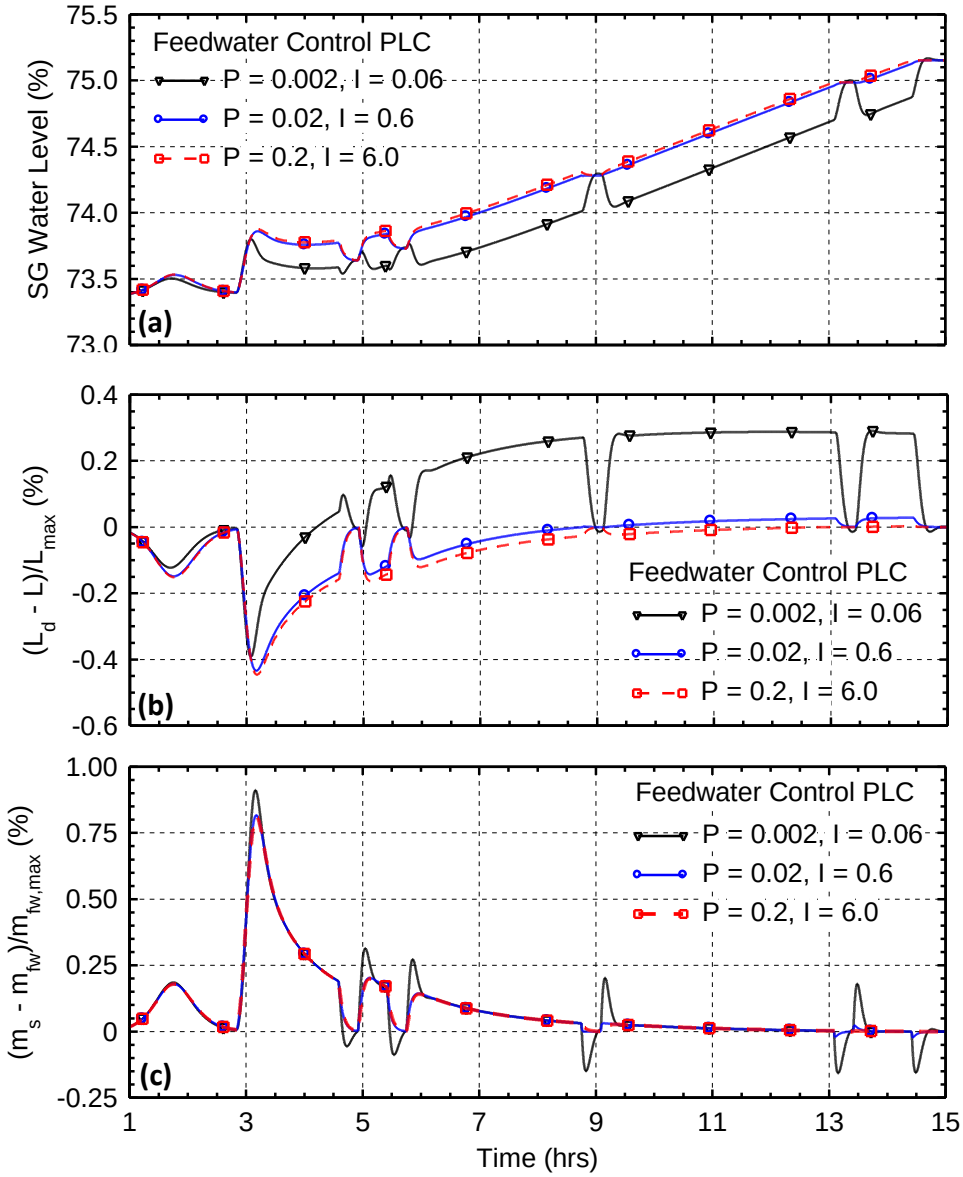

Fig. 3.15: Effect of PI controller constants for the steam generator feedwater PLC on the water level in steam generator, and the normalized differences in the water level and flow rate during a simulated startup transient.

3.5 Reactor Coolant Pump PLC

The PLC of the reactor coolant pumps regulates the shaft rotation speed of these large pumps which circulate the coolant within the primary loops and through the reactor core. The control program of the pump PLC shown in Fig. 3.16 receives the values of the state variables calculated by the developed reactor primary loop model to compute the pump characteristics. The calculated pressure losses along a straight length of the hot leg piping are used to calculate the primary loop mass flow rate using Eq. 2.6. The calculated hot leg and cold leg coolant

temperatures and the system pressure are used to compute the thermophysical properties for the water using correlations based on the IAPWS Industrial Formulation 1997 standard (International Association for the Properties of Water and Steam, 2007).

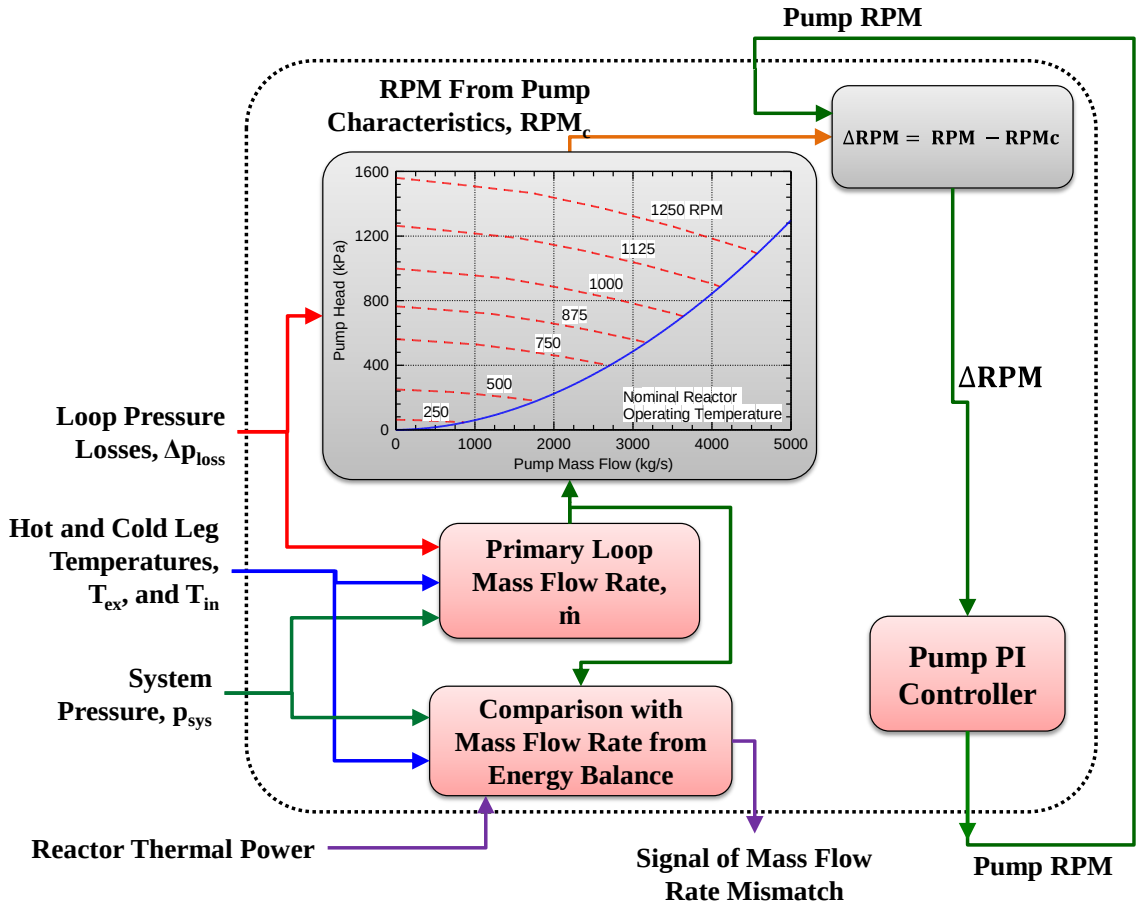

Fig. 3.16: A block diagram of the logic in the developed control program of the emulated PLC for the primary coolant pumps in a representative PWR plant.

The determined coolant flow rate and loop pressures losses are used to calculate the target pump rotational speed, RPMc, using the programed pump characteristic curves (Fig. 3.16). The difference between the current rotation speed of the pump shaft and the computed value of RPM_C, ΔRPM, is communicated to a PI controller to adjust the shaft speed of the reactor pumps as follows:

$$\mathrm{RPM(t)} = \mathrm{P} \times \Delta\mathrm{RPM(t)} + \mathrm{I} \int_0^t \Delta\mathrm{RPM(t)}\, \mathrm{dt} \qquad (3.3)$$

In this expression, P is the proportional gain constant, and I is the integral gain constant of the pump PLC controller.

In addition to controlling the pump shaft rotational speed, the reactor coolant pump PLC computes the mass flow rate from the overall energy balance in the primary loops using the temperature difference between the hot and cold leg temperatures, the reactor thermal power, and the coolant specific heat capacity based on the IAPWS Industrial Formulation 1997 standard. This flow rate is compared to that determined from the computed pressure losses. The pump PLC generates a signal if the two mass flow rates differ by more than a specified tolerance. A significant disagreement between the two flow rates could indicate an error in the measurements

of the operation condition of the reactor pump(s).

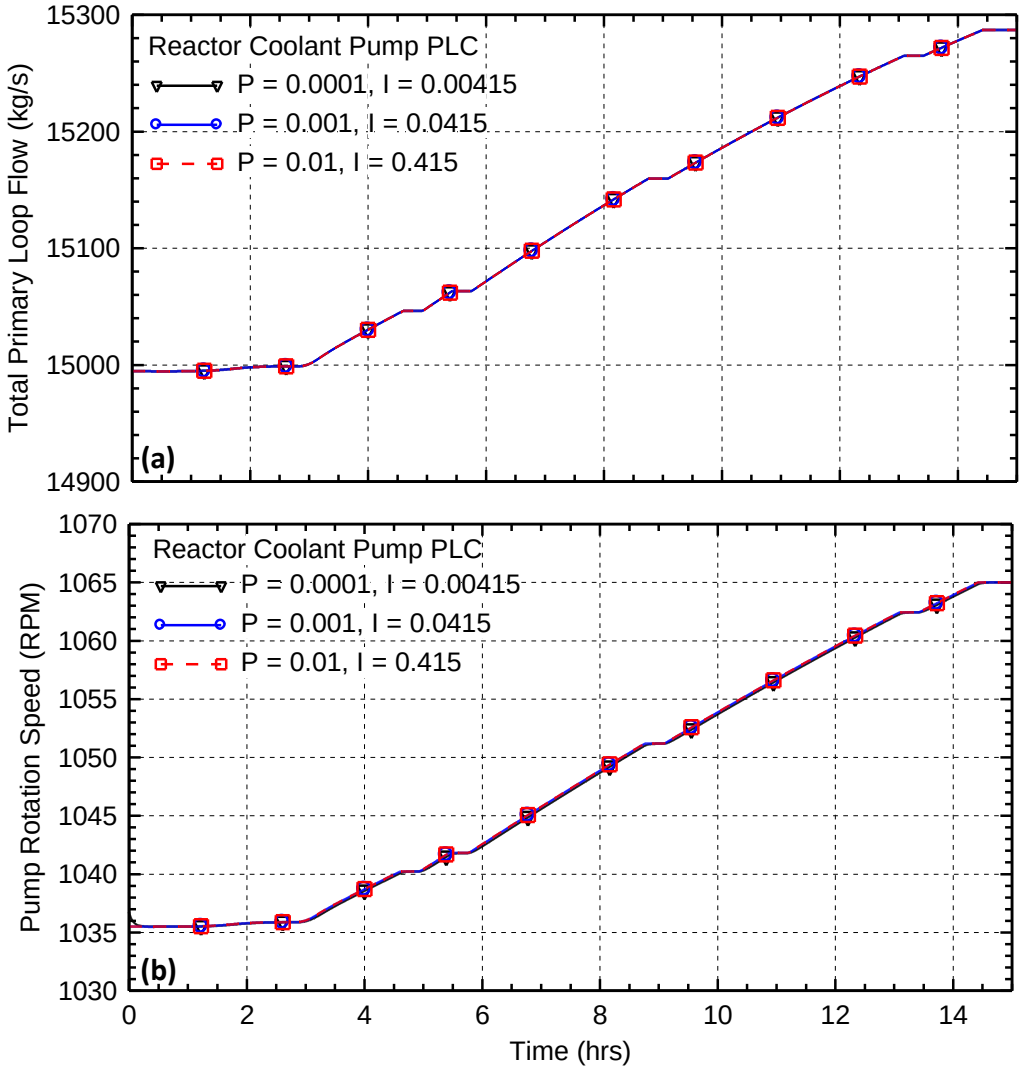

Fig. 3.17: Effect of PI controller constants on the response of the reactor coolant pump PLC and the primary loop coolant flow rate during a simulated reactor startup transient.

3.5.1 Tuning of Reactor Coolant Pump *PLC PI Controller Constants*

Testing is performed to tune the values of the proportional and integral gain coefficients, P, and I, of the PI controller for the reactor coolant pump PLC. The objective is to reduce the magnitudes of swings in the mass flow rate and the pump rotation speed during a simulated reactor startup transient. Fig. 3.17a-b show the effects of the magnitudes of the P and I coefficients for the PI controller of the pump's emulated PLC on the calculated rotational speed of the pump shaft and the coolant mass flow rate in the primary loop during a simulated reactor startup sequence (Fig. 3.9) (El-Genk et al., 2020b). These figures show that the magnitudes of the control coefficients have relatively small effect on both the pump shaft rotational speed and the coolant mass flow rate. The smallest coefficients, P = 0.0001 and I = 0.00415, result in the slowest swings in the pump shaft rotation speed. However, the system took longer to reach steady state conditions after transient changes in the reactor coolant temperatures (Fig. 3.9b). The larger coefficients, P = 0.001 I = 0.0415 and P = 0.01 I = 0.415, produce nearly identical results

showing negligible swings in the calculated results. Based on these results, the PI controller coefficients of P = 0.001 and I = 0.0415 are selected for the PI controller of the emulated PLC for the reactor pump.

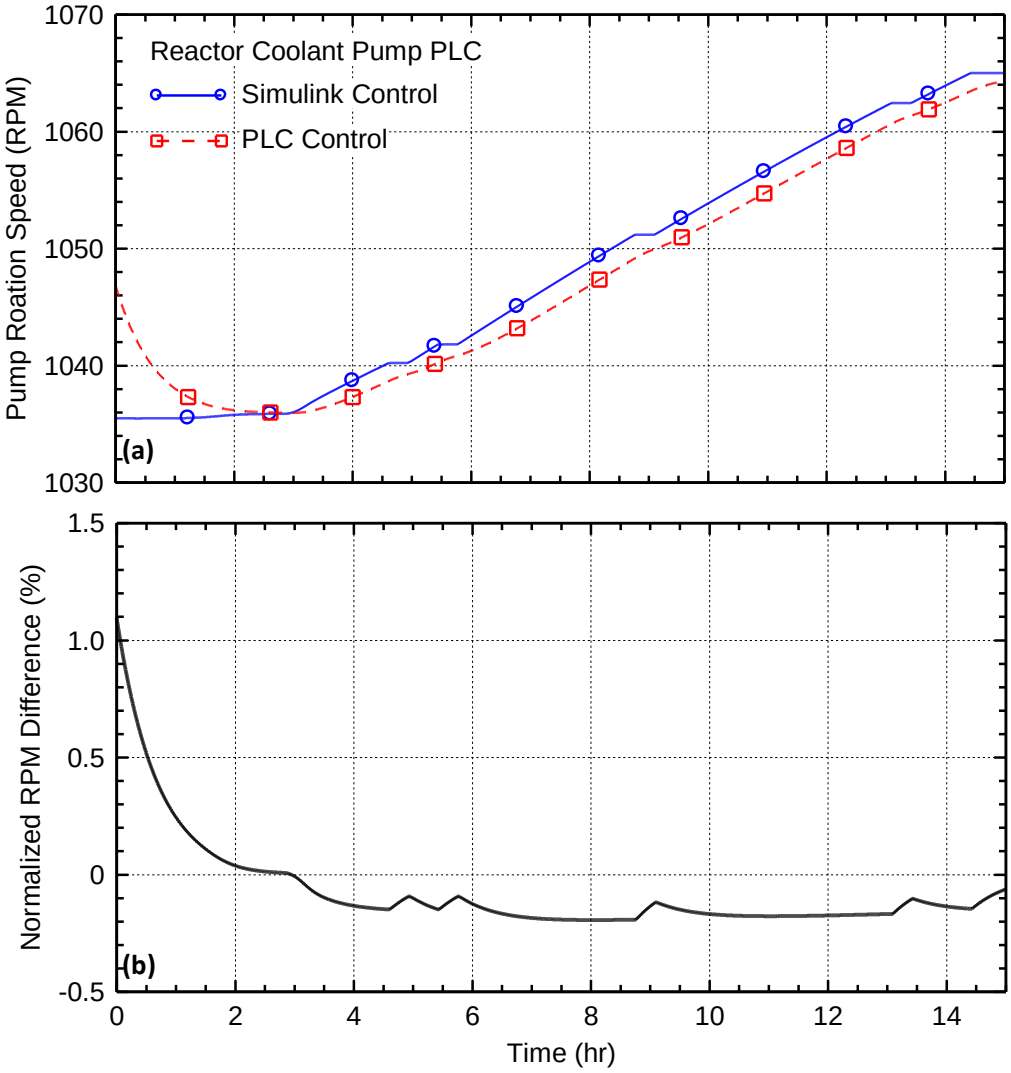

Fig. 3.18: Comparison of pump rotation speeds using external PLC and internal Simulink control during a simulated reactor startup transient.

Figure 3.18 compares the calculated pump shaft rotational speed using an external PLC running the reactor coolant pump program that is coupled to the developed physics-based integrated model of a representative PWR plant. The PWR plant model and reactor coolant pump PLC are tested for the same reactor startup sequence in Fig. 3.17 (El-Genk et al., 2020a). In this test, the coefficients for the PI controller are set to P = 0.001 and I = 0.0415, the PLC cycle frequency is set to 50 ms, and a simulation timestep is also set at 50 ms. The transient response of the pump PLC is compared to that of an 'ideal' controller in the Simulink model running the same control logic. The results in Fig. 3.18a show that the external PLC responds slower than the internal Simulink controller that has no response delay. Also, the pump shaft rotation speed is consistently lower during most of the simulated transient. Fig. 3.18b shows that the difference between the calculated pump rotation speeds is relatively small, typically less than 0.2%.

3.6. Summary

This section presented the designs and described the operation functions of several PLCs within a representation PWR plant's operation I&C system. The PLCs provide autonomous control of the developed, physics based transient model of plant. They regulate the reactor power, the system pressure, the water levels in the pressurizer and the steam generator, and the rotation speed of the reactor coolant pumps. These PLCs are emulated using a virtual machine running open-source OpenPLC software within a Raspian virtual machine. The developed control programs for the different PLCs are written in structured text standard PLC programming language. The different PLCs in the I&C system of a representative PWR plant are produced by changing the control program in the OpenPLC runtime.

Control programs are developed for the PLCs for regulating the reactor operation the pressurizer pressure, the water level in the pressurizer, the steam generator feedwater control, and the reactor coolant pump. The reactor regulator PLC monitors the thermal power to determine if a mismatch exists between the operator specified power and those calculated by the reactor's kinetics model and from the pressure losses in a segment of the primary loop. The pressure PLC regulates the system pressure by controlling the proportional and the backup heaters and the cold-water spray in the pressurizer.

The pressurizer's water level PLC adjusts the charging and letdown flow rates in the primary loop to regulate the coolant inventory. The feedwater control PLCs maintain the water level on the shell side of the steam generators within preprogramed setpoints by adjusting the rate of feedwater injection into the steam generator. The reactor coolant pumps PLCs regulates the rotational speed of the pump shaft according to the prescribed pump characteristics. The developed PLCs for both the operation and safety I&C systems in a representation PWR plant are successfully tested for reliability, fidelity, and the controller response. The PLCs with setpoint controllers are tested for the response time, while those with PI controllers are tested to determine the appropriate values of the proportional and integral gain coefficients to achieve a smooth response during operational transients.

4. Summary and Conclusions

This report details the work performed at UNM-ISNPS to develop a physics-based dynamic model of a representative PWR plant. This model is an important part of the Nuclear Instrumentation & Control Simulation (NICSim) platform, being developed at the University of New Mexico, in collaboration with Sandia National Laboratory with a DOE NEUP 2018 award. Described are the development emulated PLCs in the I&C system architectures for the plant protection and safety monitoring and the plant operation. The former facilitates autonomous reactor trip and the engineered safety features (ESF) actuation functions. The later assists the operators in regulating the plant's operation by automatically actuating control mechanisms to adjust state variables in the plant within programed setpoints. Examples are the primary loop system pressure and coolant flow rate, and the reactor thermal power.

Emulated PLCs are developed using a virtual machine with open-source OpenPLC software to run their control logic programs. The OpenPLC software runs control programs written in IEC 61131-3 standard structured text PLC programming language. The virtual machines run using the VMWare virtualization platform with an image of the Raspian operating system with OpenPLC. The state variables and control signals are communicated to and from the control program within the OpenPLC runtime over the network using Modbus over TCP/IP.

The protection and safety monitoring I&C system architecture in a representative PWR plant includes a CPC PLC for performing the reactor trip function, a ESFAS PLC for autonomously actuating the plants engineered safety features, and a coincidence logic processor PLC, which compares the voting signals from four separate safety divisions and determines whether there is a 2/4 vote to initiate a trip. The CPC and ESFAS PLCs continuously receive state variables from the developed, physics based transient model of a representative PWR primary loop. It compares their values and those of the calculated safety parameters to programed setpoints to determine whether the PLCs should vote to trip the reactor or actuate one of the plant's ESF systems. Results of transient testing shows that the emulated PLCs, when configured with a sampling rate of > 25 Hz, have a signal response delay of < 100 ms, which is acceptable in practical applications.

This work developed and tested emulated PLCs in the operation I&C system to provide autonomous control of the developed physics-based integrated model of a representative PWR plant. These PLCs regulate the reactor thermal power, the system pressure, the water levels both in the pressurizer and the steam generator, and the shaft rotation speed of the reactor pumps. The reactor regulator PLC monitors the thermal power and determines if there is a mismatch between the operator's specified value and those calculated by the reactor kinetics model from the pressure losses in a segment of the primary coolant loop. The pressure PLC regulates the pressure in the primary loop by controlling the proportional and backup heaters and the rate of water spray in the pressurizer. The pressurizer's water level PLC adjusts the charging and letdown flow rates in the primary loop to control the water inventory in the primary loop. The feedwater control PLCs maintain the water level on the shell side of the steam generators to within the preset points by regulating the rate of feedwater injection into the steam generator. The reactor coolant pumps PLCs regulate the shaft rotational speed of the pumps for achieving the desired coolant flow rate in the primary loop.

The emulated PLCs developed are tested for reliability and fidelity of the controller responses. The performed transient testing of the PLCs with setpoint controllers investigated their response timing, and the values of proportion and integral gain coefficients for the PLCs with PI controllers to achieve a smooth system response during operational transients. Testing of

the pressurizer pressure PLC while linked to the transient physics-based model of the pressurizer involved a surge in and surge out transients. Results show a PLC signal response delay of 50 ms, which is acceptable for industrial applications. The determined coefficients for the PI controller of the pressurizer water level PLC to achieve a smooth response during operation transients are P = -2800 and I = -50. The testing results of the steam generator feedwater control PLC in a simulated reactor startup sequence, show that values of P = 0.02 and I = 0.6 produce the smallest difference between the normalized feedwater flow rate and the normalized steam generator water level. Testing of the reactor coolant pump PLC during a simulated reactor startup sequence show that the most appropriate magnitudes of the PI controller's proportional and gain coefficients are P = 0.001 and I = 0.0415. These coefficients produced smooth transient responses of the pump shaft rotation speed and the coolant flow rate in the primary loop during a simulated reactor startup sequence.

The research detailed in this report developed and demonstrated important elements for future implementation into the NICSim platform. The developed emulated I&C system and PLCs in a representative PWR plant will be integrated within the SCEPTRE framework to support cybersecurity analyses. A planned technical task will be to integrate the developed, transient physics-based model of a representative PWR plant along with the developed emulated PLCs in the I&C system into the NICSim platform. It will also test its functionality for investigating potential cyberattacks. Future uses of the NICSim platform could include investigating: (a) physical impacts of targeted cyber-attacks on the I&C system architectures of a representative PWR plant, (b) potential for training plant operators to identify signs of a potential cyberattack, and (c) development of metrics to quantify propagation of a potential cyberattack within the I&C system of a nuclear power plant.

5. Acknowledgements

This research is funded by the DOE Office of Nuclear Energy's Nuclear Energy University Program under Contract No. Nu-18-NM-UNM-050101-01 to the University of New Mexico. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the U.S. Department of Energy.

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. DOE's National Nuclear Security Administration under contract DE-NA-0003525. The views expressed in the article do not necessarily represent the views of the U.S. DOE or the United States Government.

6. References

Altamimi, R., El-Genk, M.S., Schriener, T.M., 2020, "Pressurizer Model and PLCs for Investigation of Cybersecurity of PWR Plants". In Trans. ANS Annual Meeting, Phoenix, AZ, June 7-11, 2020.

Alves, T.R., Buratto, MM, Mauricio de Souza, F., and Rodrigues, T.V., 2014. "OpenPLC: An open source alternative to automation," in proceedings IEEE Global Humanitarian Technology Conference (GHTC 2014), San Jose, CA, USA, DOI: 10.1109/GHTC.2014.6970342

Camacho-Lopez, T.R., 2016, "SCEPTRE," Electricity Subsector Coordinating Council & Government Executives Meeting," Albuquerque, NM, USA.

Combs, G., 2019, "Wireshark,"https://www.wireshark.org/

Dragos, Inc., 2017a, "TRISIS-Hatman Malware Analysis of Safety Systems Targeted Malware," https://dragos.com/wp-content/uploads/TRISIS-01.pdf.

Dragos, Inc., 2017b. CRASHOVERRIDE, Analysis of the Threat to Electric Grid Operations version 2.20170613, www.DRAGOS.com.

El-Genk, M.S, Schriener, T.M., Lamb, C., Fasano, R., Hahn, A., 2019, Implementation and Validation of PLC Emulation and Data Transfer, Report No. UNM-ISNPS-02-2019, Institute for Space and Nuclear Power Studies, University of New Mexico, Albuquerque, NM, USA

El-Genk, M.S, Schriener, T.M., Hahn, A., Altamimi, R., Lamb, C., Fasano, R., 2020a, A Physics-based, Dynamic Model of a Pressurized Water Reactor Plant with Programmable Logic Controllers for Cybersecurity Applications, Report No. UNM-ISNPS-02-2020, Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA.

El-Genk, M.S., Schriener, T.M., Lamb, C.C., 2020b, "Nuclear Instrumentation and Control Simulation (NICSim) Platform for Investigating Cybersecurity Risks." In Trans. ANS Annual Meeting, Phoenix, AZ, June 7-11, 2020.

El-Genk, M. S., T. Schriener, R. Altamimi, A. Hahn, C. Lamb, R. Fasano, 2020c, "NICSIM: Nuclear Instrumentation and Control Simulation for Evaluating Response to Cyber – Attacks," Proc. 28th International Conference on Nuclear Engineering (ICONE28), ICONE28-POWER2020-16756, Anaheim, CA, USA, 2-6 August 2020.

El-Genk, M.S., Altamimi, R., Schriener, T.M., 2020, "Pressurizer Dynamic Model and Emulated Programmable Logic Controllers for Nuclear Power Plants Cybersecurity Investigations," Annals in Nuclear Energy, submitted for review.

Falliere, N., Murchu, L., Chien, E., 2011, "W32 Stuxnet Dossier," Symantec, https://www.symantec.com/content /en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf.

Fasano, R., Lamb, C., El-Genk, M.S., Schriener, T.M., Hahn, A., 2020. "Emulation methodology of programmable logic controllers for cybersecurity applications," Proceedings of the 2020 28th Conference on Nuclear Engineering Joint with the ASME 2020 Power Conference ICONE28-POWER2020, August 2-6, 2020, Anaheim, California, USA, paper ICONE28-POWER2020-11150

Hahn, A., El-Genk, M.S., Schriener, T.M., 2020a, "Programmable Logic Controller of a Pressurized Water Reactor Core Protection Calculator". In Trans. ANS Annual Meeting, Phoenix, AZ, June 7-11, 2020.

Hahn, A., Schriener, T.M., El-Genk, M.S., 2020b. "Selection and validation of fast and synchronous interface to the controller of a space nuclear reactor power system," Proceedings of the 2020 28th Conference on Nuclear Engineering Joint with the ASME 2020

Power Conference ICONE28-POWER2020, August 2-6, 2020, Anaheim, California, USA, paper ICONE28-POWER2020-16237

Hung, P.L., 2010, "Core Protection Calculator System: Past, Present, and Future," Proc. 18th International Conference on Nuclear Engineering (ICONE18), Xi'an, China, May 17–21, 2010, ICONE18-29001

International Association for the Properties of Water and Steam, 2007, The International Association for the Properties of Water and Steam Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam, Lucerne, Switzerland, IAPWS R7-97(2012)

Jens, W.H., Lottes, D.A., 1951. "Analysis of Heat Transfer, Burnout, Pressure Drop, and Density Data for High Pressure Water," Argonne National Laboratory, Chicago, IL, ANL-4627.

Karnouskos, S., 2011. "Stuxnet Worm Impact on Industrial Cyber-Physical System Security," in proceedings IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society, Melbourne, VIC, Australia, 7-10 November, 2011, DOI: 10.1109/IECON.2011.6120048.

Korsah, K., et al., 2008. Instrumentation and Controls in Nuclear Power Plants: An Emerging Technologies Update, US NRC Technical Report NUREG/CR-6992, Washington, DC.

National Research Council, 1997. Digital Instrumentation and Control Systems in Nuclear Power Plants, Safety and Reliability Issues, Final Rep., National Academy Press, Washington, D.C.

Nise, N.S., 2015, Control Systems Engineering Seventh Edition, Wiley.

Nuclear Energy Institute, 2010, "Cyber Security Plan of Nuclear Power Reactors," NEI Technical Report NEI 08-09 [Rev.6].

Palo Verde Nuclear Generating Station, 2017. Palo Verde Nuclear Generating Station Units 1, 2, and 3 Updated Final Safety Analysis Report, Rev. 19 Corrected (Redacted per RIS 2015-17)

Perlroth, N., 2019. "Hackers are Targeting Nuclear Facilities, Homeland Security Dept. and F.B.I. Say", New York Times, June 19, 2019.

Sandia National Laboratories, 2016. SCEPTRE, Sandia Document SAND2016-8095C.

Schindhelm, E.P., Single, R.E., 2010. AP1000 Protection and Safety Monitoring System Architecture Technical Report, Technical Report WCAP-16675-NP, APP-GW-GLR-147, rev 2, Westinghouse Electric Company LLC, Cranberry Township, PA

Schriener, TM, El-Genk, MS, 2020, "Steam Generator Model and Controller for Cybersecurity Analyses of Digital I&C Systems in PWR Plants". In Trans. ANS 2020.

MathWorks, 2019, "Matlab & Simulink R2019b," Natick, Massachusetts, United States, https://www.math works.com/

US Department of Homeland Security, 2015, "Nuclear Sector Cybersecurity Framework Implementation Guidance."

VMware, 2019. VMware Workstation 15 Pro.

Appendix A:
Emulation Methodology of Programmable Logic Controllers for Cybersecurity Applications

A.1. Introduction

PLC emulation enables high fidelity cybersecurity and physical effect modeling of OT networks, such as the I&C systems in nuclear power plants, without hardware-in-the-loop (HITL) integration. In a HITL setup, a physical PLC is integrated into the digital network being tested. Experiments that use PLCs as HITL are prohibitively expensive, difficult to scale, and change relative to emulated systems. Running an experiment on an entire I&C system architecture, which may include dozens of devices, can become impractical if HITL is used for every digital device on the network. A PLC emulation methodology, thus, provides a practical path forward to study the current cybersecurity posture of OT networks for critical infrastructure and aid in the design of secure architectures for future systems.

Emulated computer systems, also referred to as virtual machines (VMs), are commonly used to perform cybersecurity experiments in contained virtual environments for enterprise IT systems. The DOE SCEPTRE framework, developed at SNL to enable cybersecurity analyses of ICSs, can start up and handle virtual network communication between many VMS using ICS protocols written to specification (Camacho-Lopez, 2016). PLCs are unique digital computers that require additional considerations when being emulated since PLCs are hard real-time systems, use networking protocols specific to industrial processes, and control physical processes utilizing a variety of inputs and outputs (I/O). The present work seeks to address these considerations by developing a methodology for PLC emulation to be used in the NICSim platform.

The objective of the present work is to update the emulation methodology for a PLC previously detailed in El-Genk et al. (2019) and establish metrics to validate the developed emulated PLC when coupled to the NICSim platform. This emulation methodology is applied to a representative open source PLC implementation using the OpenPLC runtime (Alves et al., 2014), and the emulated PLC is then validated against the recorded physical and digital signatures of real, hardware based, PLC.

A.2. PLC Emulation Methodology

Table A.1 outlines the general steps of the PLC emulation methodology. Emulating a PLC implies that the digital and physical behavior of the PLC is reproduced by another system through computational means. The degree of emulation is determined by the project requirements. A full emulation would reproduce the functionality of the hardware, firmware, kernel, and operating system used by the PLC. A partial emulation would replicate only some of these functionalities.

Investigations of potential vulnerabilities within software programs or the computer's operating system might only require kernel and software emulation, while investigating potential exploits of vulnerabilities in a chipset's instruction set could require full emulation at the

hardware and firmware levels. Several partial emulators of computer systems are available which emulate the kernel and software of a device (VMWare, 2019). Full emulators, however, are far less common due to the drastic increase in complexity and computational cost.

The present work requires that the emulated PLC approximate the real PLC, such that the differences between the two systems do not affect the behavior of the connected physics-based model and could support planned cybersecurity analysis. For the representative emulated I&C systems within the NICSim Platform, this is accomplished using kernel and software emulation. In addition, the real and emulated systems should be interchangeable and not impacted by the computer hardware running the emulation.

Table A.1: Steps within the PLC emulation methodology.

Step	Action
1	Determine degree of emulation needed
2	Use commercial/open-source emulation software or develop the emulation that is needed
3	Based on emulation requirements determine the physical and digital signatures of the PLC that need to be emulated
4	Benchmarked emulated PLC against the real PLC using a representative test environment
5	Collect data for both the real and emulated PLC and compare physical and digital signatures
6	Change emulation or configuration of the PLCs as needed until the signatures of the real and emulated PLCs converge to an acceptable range outlined by the project requirements

Once a PLC emulation is successfully developed, it is validated against the real system. When obscuring the firmware and the underlining hardware of a PLC, the only way to determine if a PLC is real or an emulation is to observe the digital and physical signatures of the device. Table A.2 shows the metrics used to validate the PLC emulation. The selected digital signatures of a PLC include the network response, the network traffic, and the software versions. The network response of a device quantifies how the network traffic is transmitted and received and determines the rate of data transfer. Similarly, the underlying network traffic for the system determines the type and frequency of the network data packets being transmitted and/or received. Finally, the software versions determine if the exact same software is running on the emulated and real device. If a cybersecurity flaw exists in the software, it should be exploitable on both the real and emulated PLC in the same manner.

The selected physical signature metrics for the PLC include the actuation response time and sampling time (Table A.2). The actuation response time is the time required for a PLC to receive data, compute an output, and send an actuation signal. Differing actuation response times would lead to different physical outcomes by influencing the time history of control signals to the

physical process. Similarly, the sampling time of a digital controller, also referred to as the scan time, would affect the process being controlled by influencing the gain values of complex controller designs (Nise 2015). The sampling time of the PLC also determines whether the system is running synchronized with real-time, which is required in order to achieve a deterministic response. For time critical applications designers can be confident that actuation response times will be approximately between one to two sampling periods plus network latency. The digital signatures are important from a cybersecurity perspective, while the physical signatures determine the fidelity of the PLC's response to the connected process. Since PLCs are the interface between the physical and digital world, it is paramount that the signatures of the emulated and real PLC be quantified to validate the cyber-physical coupling of the emulation.

Table A.2: Metrics to compare a real and emulated PLC.

Digital Signatures:
Network response
Network traffic
Software versions
Physical Signatures:
Actuation response time
Sampling time

A.3. Testing Methodology

The developed emulation methodology for PLCs is performed and validated using a representative, open source PLC architecture consisting of a Raspberry Pi 4 minicomputer running the OpenPLC software. OpenPLC implements IEC 61131-3 standard programming for PLCs (Alves et al., 2014). The Raspberry Pi is chosen because of the availability of its open source operating system to create images, the functionality of its on-board digital/analog IO, compatibility with OpenPLC, and the ability to emulate the operating system.

Table A.3: Real and emulated PLC specifications.

System	**Real**	**Emulated**
Hardware	Raspberry Pi 4	VMware Virtual Machine
CPU	Broadcom BCM2711, Quad core Cortec-A72 (ARM v8) 64-bit SoC @ 1.5Ghz	AMD FX-8370 64-bit SoC @ 4.3Ghz (4 virtual cores)
RAM	4 Gb	4 Gb
Operating System	Ubuntu 19.10	Ubuntu 19.10
Control Software	OpenPLC Version 3	OpenPLC Version 3
Network Interface	Gigabit Ethernet	Gigabit Ethernet

The specifications of real and emulated PLCs are summarized in Table A.3. VMware emulation software is used to emulate an Ubuntu Server kernel and operating system for the emulated PLC using a type-2 hypervisor (VMWare, 2019). Both the real and emulated PLCs run

the same operating system and OpenPLC software. To reduce the differences between the Raspberry Pi 4 hardware and PC running the VMware emulation software, four gigabytes of RAM and four processor cores are allocated to the emulated PLC in VMware.

The testing environment used to validate the PLC emulation against the physical hardware links the real or emulated PLC to a transient simulation model running in Matlab Simulink (Fig. A.1) (The Mathworks, 2019). The Simulink simulation running on a separate Linux server takes the place of an external physical process being controlled by the PLC within the testing environment. Using a simulated process complicates the comparison of the actuation response times of the real and emulated controllers. This is because of an additional requirement of running the simulation in sync with real-time, when using asynchronous communication between the simulation and the PLC. Actual physical processes are continuous as opposed to the inherently discrete numerical simulation. However, when doing cybersecurity research on a physical system, such as a nuclear reactor power plant, using a simulated process is the only practical option due to considerations of safety and cost.

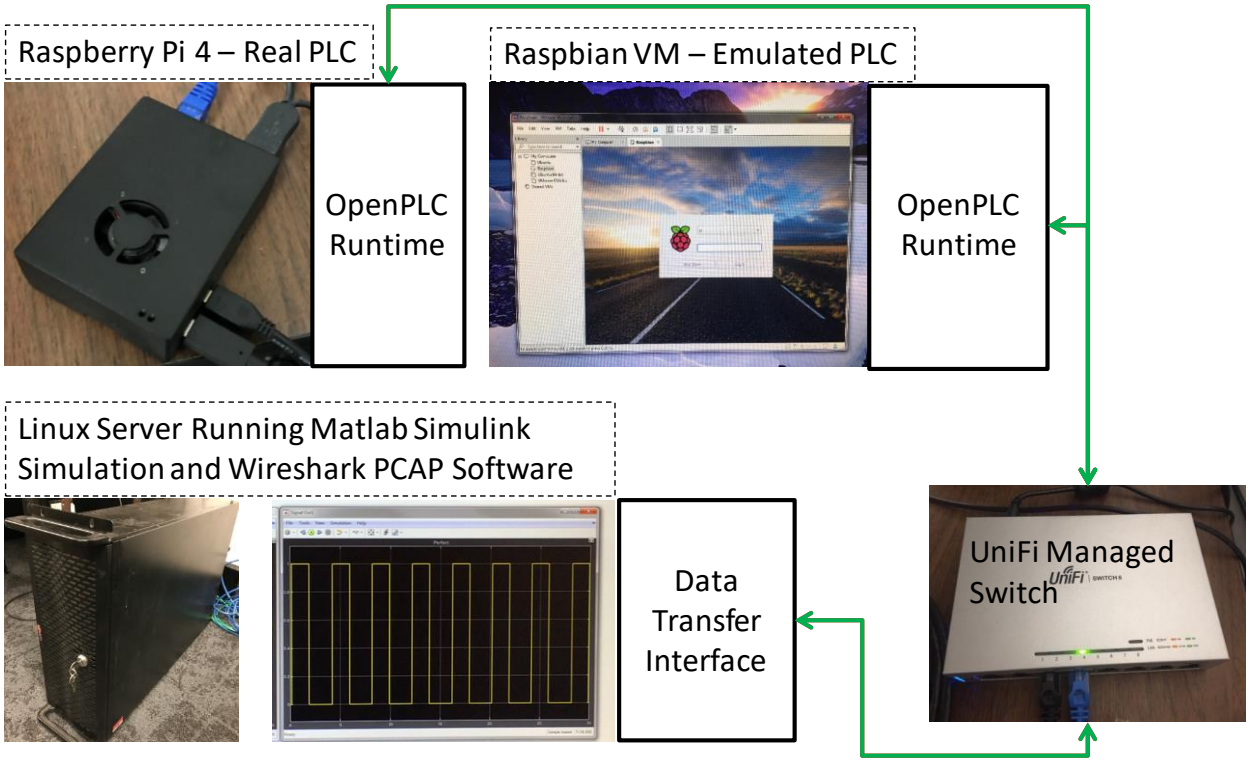

Fig. A.1: Testing setup for comparing signatures of real and emulated PLCs.

Figure A.1 shows the components of the network setup used for testing and validation of the signatures of the real and emulated PLCs. The testing setup is comprised of a Linux server, which runs the Simulink model, a UniFi switch, and a PLC, all linked by isolated ethernet network. For each test, only one real or emulated PLC is connected to the UniFi switch at a time. The Simulink simulation within the benchmark testing environment generates a repeating square wave signal with an amplitude of one, a period of five seconds, a pulse width that is 20% of the period, and with no phase delay.

The square wave is communicated to the network using the data transfer interface via ModbusTCP and sent to the UniFi switch to be routed to the real or emulated PLC. An isolated testing network is used to eliminate the network routing differences between the real and emulated PLCs. A local Dynamic Host Configuration Protocol (DHCP) server, which runs on the Linux server, handles IP address allocation for the Linux server and PLCs. The Wireshark software captures the network data traffic between the Linux server and real or emulated PLC (Combs, 2019).

When the data interface writes new values to the PLC, the implemented ladder logic programming in OpenPLC determines if the square wave is above or below a predefined setpoint. A setpoint value of 0.5 is used to determine the output register. An input value greater than 0.5 results in a value of one, while a value less than 0.5 results in a value of zero. The total Simulink simulation length for each run is 300 s using a major time step size of 50 ms.

The developed NICSim data transfer interface program is employed to handle the asynchronous inter-process communication between the Simulink simulation and the remote PLC (Fig. A.2) (Hahn, Schriener, El-Genk, 2020b). This interface communicates the state variables calculated by the transient Simulink simulation to an external interface Python program using a Matlab S-function. The Matlab S-function, written in the C programming language, communicates with the external python interface using shared memory inter-process. The state variables are written and read from a shared memory location named ‘publish’ (Fig. A.2). The control signals determined by the real or emulated PLCs are passed back to the Simulink model through a second shared memory location named ‘update’. Inter-process communication semaphores control access to the two shared memory locations, ensuring that only one side of the interface can access a given shared memory location at a time.

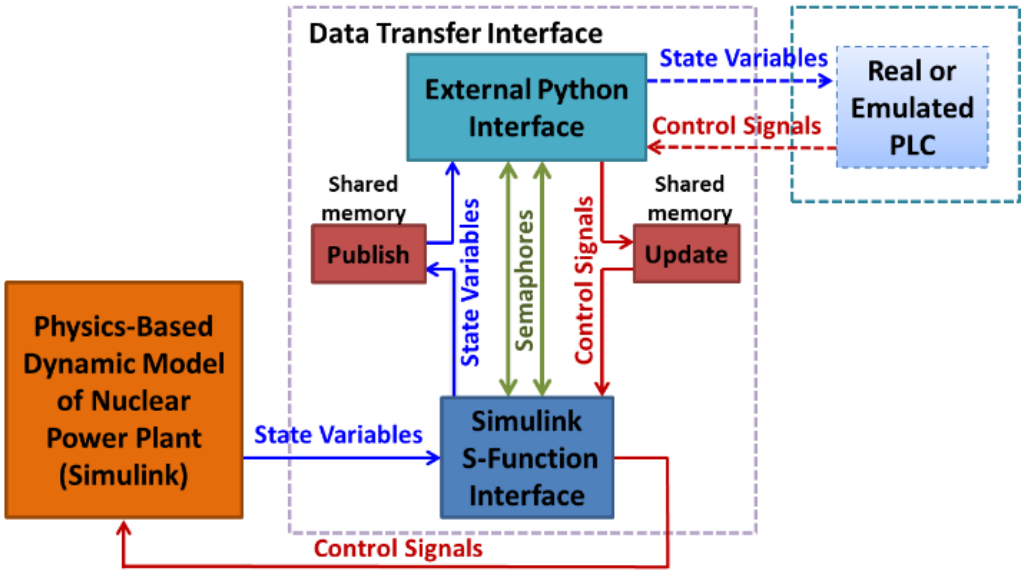

Fig. A.2: Shared Memory Interface for Data Flow Between Simulink Simulation Model and Real/Emulated PLCs (Hahn, Schriener, El-Genk, 2020a, 2020b).

The python interface program communicates with the PLCs using the Modbus TCP protocol to write or read input and output registers on either the emulated or real PLC. Modbus TCP was originally a serial protocol specifically created to be used with PLCs. This non-proprietary protocol is used in OT networks making it an ideal protocol to benchmark real and emulated PLCs network response characteristics. Each PLC uses the OpenPLC runtime, compatible with Modbus TCP communication, to run the PLC's control programming. The OpenPLC takes the values written to input registers and the PLC's control program uses these values to calculate a response and write the control signal to the output registers. The inter-process communication programs used in the testing are asynchronous with the Simulink simulation for both the real and emulated PLC.

A.4. Testing Results and Discussion

A total of five tests are conducted for both the real and emulated PLC to independently characterize the digital and physical characteristics of each PLC. The results of the validation testing of the opensource Raspberry Pi 4/OpenPLC implementation using the developed emulation methodology are presented and discussed in the following subsections

A.4.1 Real-Time Condition

A custom code in the python interface program attempts to run the simulation model in sync with real-time to ensure that both the real and emulated controllers receive the same values from the simulation at the same rate. Real-time synchronization is required for comparing the real PLC and emulated PLC when using asynchronous communication. If the simulated process operates at a different rate, it will affect the actuation response time of the controller. For example, if the simulation for the real PLC runs at a slower rate that the simulation for the emulated PLC, the real PLC may record faster actuation response times. Furthermore, in applications where the response of transient physics-based models is important, real-time sync of the simulation becomes paramount. Accurate timing of asynchronous signals enables reproducibility of the physical response of the transient physics models coupled to real or emulate PLCs. Since validating the system response is outside of the scope of this work, implementing real-time sync becomes less important. Thus, an approximate real-time implementation is acceptable for the purpose of the current comparison.

Simulink simulations were run with major time steps of 25, 50, 100, 250, and 500 ms to investigate how the timestep of the simulation impacted the real-time sync. It was found that the deviation from real-time sync and the timestep of the simulation were inversely proportional to each other. The computer running the simulation has a fundamental limit of how fast it can run the simulation based on its computational power. Therefore, the computational time required to calculate a simulation timestep must be less than the elapsed simulation time in order to maintain real-time synchronization.

Given the computation resources of the Linux server used, 50 ms was chosen as a representative major time step for more complex physics models with acceptable real-time sync performance given the external Python interface used. Positive deviation from real-time indicates that the Simulink simulation is running slower than real-time and vice versa. The error bars in all figures are determined from the first and third quartile of the multiple data sets collected, except for the analysis of the sampling time. The error bars for the sampling time are three standard deviations of the data collected at time.

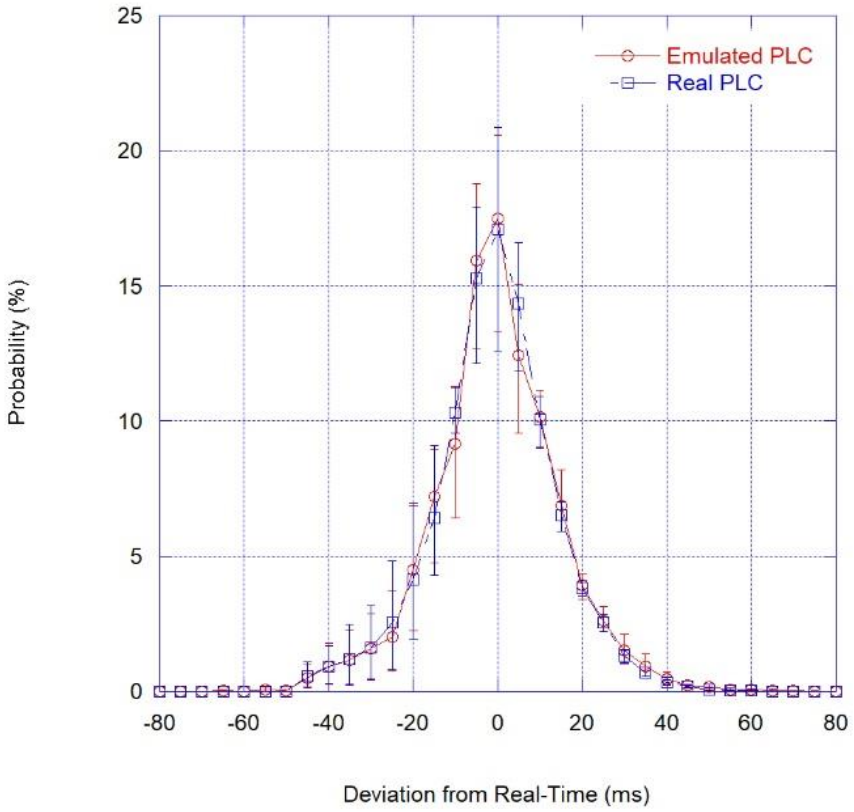

Fig A.3: Comparison of the deviation of Simulink simulation from real-time sync.

Figure A.3 shows the real-time deviation of the Simulink simulated process, calculated by subtracting wall time from simulation time, while communicating with the real and emulated PLC. On average the deviation of the Simulink simulation from real-time when communicating with the emulated PLC is, approximately -2.3ms (14.85 sigma) and -2.66ms (14.29 sigma) when communicating with the real PLC. The real-time deviations between the real and the emulated PLCs are nearly the same with the same statistical error, enabling the controllers to be compared without the simulated process affecting the validity of the comparison.

A.4.2 Network Response

The network response characteristics of the emulated and real PLCs are collected using Wireshark (Combs 2019) to capture the network traffic between the Linux server and the PLC. Wireshark is run on the Linux server for all simulation cases. The network response of the emulated and real PLCs is evaluated by comparing the ModbusTCP packet round-trip time and the network statistics for the ModbusTCP communication between the Linux server and real or emulated PLC. The network round-trip time is important because the slowest exchange of ModbusTCP packets determines the physical limit of outputting the data by the simulation to the PLC and fingerprinting the devices on the network. Therefore, the lower limit of the sampling time for the PLC is that of the slowest communication of data from the server to the client. In other environments, the communication protocol may be different from the ModbusTCP, but the underlining principle of quantifying the rate at which data can be transferred to quantify the network response remains applicable.

The ModbusTCP protocol uses query and response packets to read and write holding registers on the PLCs. Fig. A.4 shows the obtained network response to query-response packet

pairs and Fig. A.5 shows the response-query packet pairs for the real and emulated PLCs. The python data transfer interface program, PLC hardware, and inherent network latency control the rate at which the query-response and response-query packet pairs are sent and received.

As seen in Fig. A.4, the emulated PLC has a higher probability of sending and receiving ModbusTCP packets at a faster rate than the real PLC, suggesting an overall faster query-response network response. Overall, the emulated PLC has short average response time of ~ 0.73 ms, compared to ~1.78 ms for real PLC. Despite this difference in query-response packet round-trip time, the response-query packet round-trip time confirms that real and emulated PLC had virtually same response characteristics (Fig. A.5). For response-query packet round-trip the emulated PLC has an average round-trip time of ~23.56 ms, compared to ~21.87 ms for the real PLC.

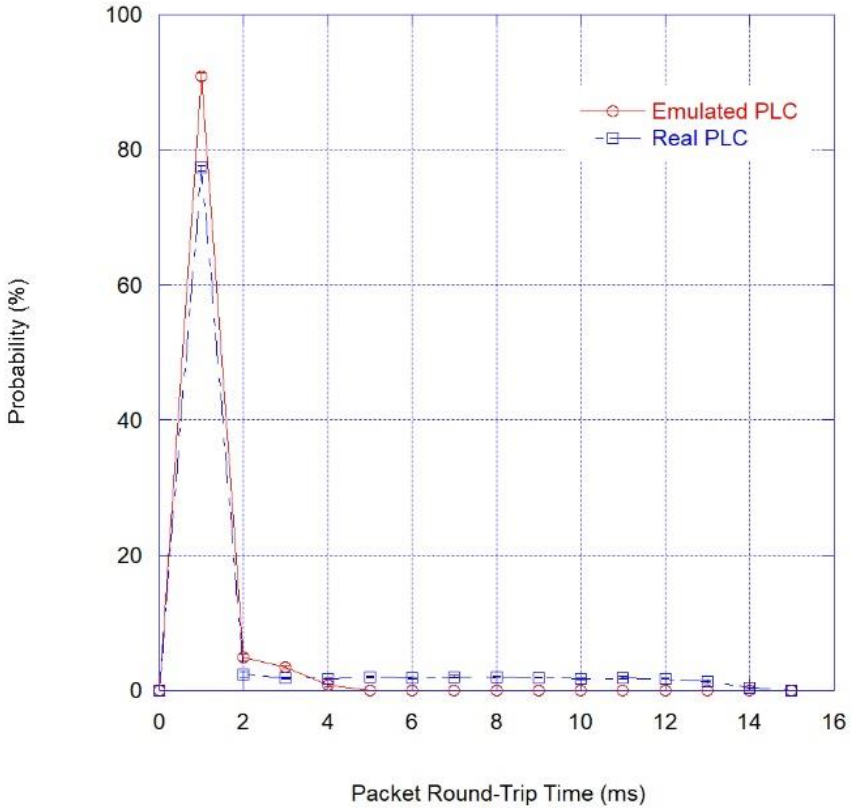

Fig. A.4: Query-response ModbusTCP packet round-trip time.

The query-response characteristics are dictated by the individual hardware for each PLC, while the response-query characteristics are dependent on the python interface code. In this case the emulated PLC hardware is faster than the emulated PLC leading to a noticeable difference in the query-response characteristics. Overall, however, the major time step used in the Simulink process simulator was 50 ms. Relative to the magnitude of the major time step the difference in network response characteristics for query-response packet pairs has a negligible effect on the physical process.

From a digital perspective, the network response characteristics are significantly different such that network monitoring would be able to distinguish the real and emulated PLC. If project requirements dictated that the query-response characteristics be in better agreement the network packets emanating from the emulated PLC could be slowed down. Using additional routing and

leveraging custom algorithms the network response for query-response packet pairs could plausibly approach the networking response generated by the real PLC.

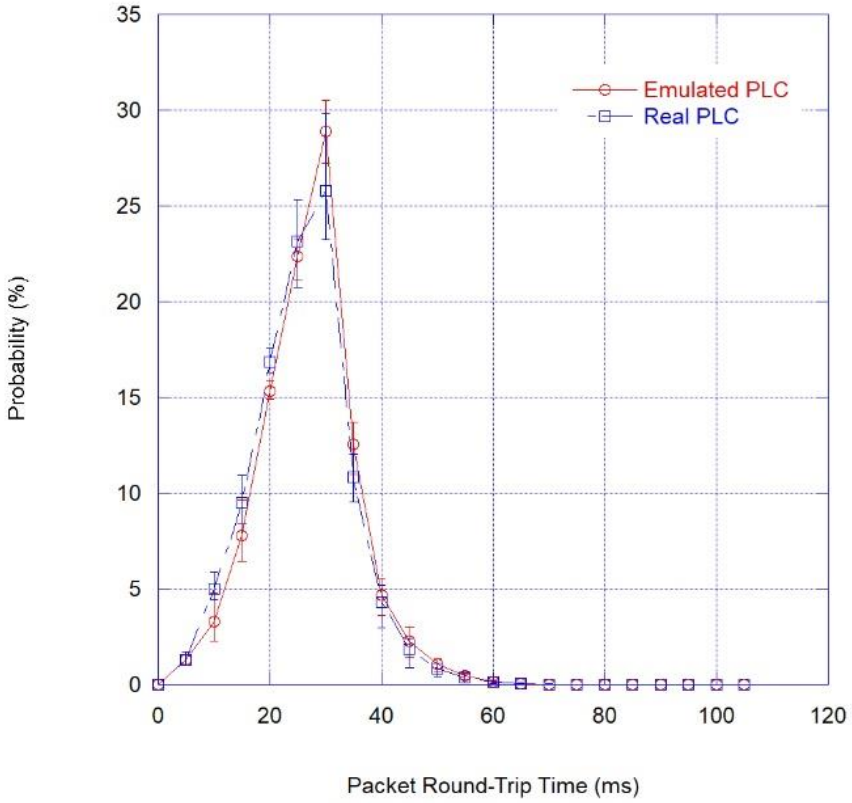

Fig A.5: Response-query ModbusTCP packet round-trip time.

A.4.3 Network Traffic

The collected Packet Capture (PCAP) data is analyzed to determine the types of packets sent during the experiment. The data for all experiments was combined for the real and emulated PLC to decrease the differences in the time of day each experiment is ran. It is possible that some programs will send out network communication based on the time of day or the current state of the operating system. The numbers and categories of data packets collected by the Wireshark utility are given in Table A.4 for the emulated and real PLC tested. The vast majority (> 99%) of the collected data packets are TCP/IP packets. The fraction of the total packets collected in each category between the real and emulated PLCs differed by < 1%.

The network traffic signatures for the emulated PLC are determined to be comparable to the real PLC, although the emulated PLC has a slightly higher bandwidth. The collected PCAP data in Table A.4 shows however, that the emulated PLC can use the same network protocols and generate packets of the same data types and with similar proportions to the total level of network traffic as the real PLC.

These similarities in types and proportions of network traffic are important for the capability of the emulated PLC to represent the real PLC in cybersecurity investigations. As mentioned in the previous section, if a more accurate networking response is required the network response of the emulated PLC could be altered. In this case slowing down the response time of the emulated PLC to match the response of the real PLC would also ensure that both PLCs had the same networking bandwidth.

Table A.4: Network traffic capture for the real and emulated PLC.

PLC	Packet Type	Average	STD	Percentage of Total (%)
Real	TCP	25666.60	1946.36	99.86
	UDP	35.20	8.15	0.14
Emulated	TCP	27455.40	1266.14	99.84
	UDP	43.60	16.88	0.16

A.4.4 Sampling Time

The rate at which a PLC samples the controlled process is a very important parameter based on classical control theory, and in terms of determining if the PLC is fast enough to monitor the physical process in question (Nise 2015). The sampling time of the real and emulated PLC is measured using an iterative algorithm implemented in python that writes a value to a register, waits for the PLC to execute its ladder logic, and records the time it takes the PLC to submit a new control action based on the new input. This method assumes that python script is running at a much faster rate than the ladder logic program and that the computational time required to record the elapsed time is negligible relative to the sampling time. These are valid assumptions for the current testing setup. The real and emulated PLCs are tested with sampling times specified in the OpenPLC program of 1, 25, 50, 100, 250, and 500 ms, and actual sampling time is recorded using the python sampling time algorithm to check for any deviations from the ideal, specified sampling time.

Figures A.6a and A.6b present the results of the sampling time analyses for the emulated and real PLCs, respectively. The sampling time results for 1 –100 ms are shown in Fig. A.6a, with the results for sampling times of 175-500 ms shown in Fig. A.6b. The actual and ideal sampling times are nearly equivalent in the mean for both the emulated and real PLCs from 1-500 ms. Comparing the variance in the measured sampling time, for sampling times above 50 ms the real PLC experiences less variance than the emulated PLC. The difference in the variance between the emulated and real PLCs decreases as the sampling time increases, with the smallest variance observed at a sampling time of 250ms.

These results for the sampling time suggest that the emulated PLC matches the same average sampling time as the real PLC. To achieve the most consistent results and enable the best comparison between controllers a sampling time of ≥ 250 ms should be used. A sampling time of 250 ms is used for the real and emulated PLC in all tests. For experiments that require an emulated PLC sampling time less than 250 ms a type-1 hypervisor is recommended to avoid the scheduling and lower resource priority level of a VM using a type-2 hypervisor. The emulated PLC with a type-2 hypervisor used in this work was unable to match the consistent performance of the real PLC for sampling times less than 175 ms to approximately 50 ms. Below 50 ms emulated PLC outperformed the real PLC by maintaining a sampling time with less variance.

A.4.5 Actuation Response Time

The actuation response time is a physical signature for quantifying the performance of a PLC. It is defined here as the time it takes for the PLC to execute a control action based on the sampled inputs from the process being controlled. Internal logic in the Simulink simulation model is used to generate a ‘true’ response signal, which is used to quantify the observed deviation of the actuation response from the ideal response. This ideal ‘true’ signal is recorded internally by Matlab Simulink and is independent of the data transfer interface and the

Modbus/TCP communication to and from the PLC. Thus, the ideal response represents a control action with zero-time lag relative to that of the process simulation.

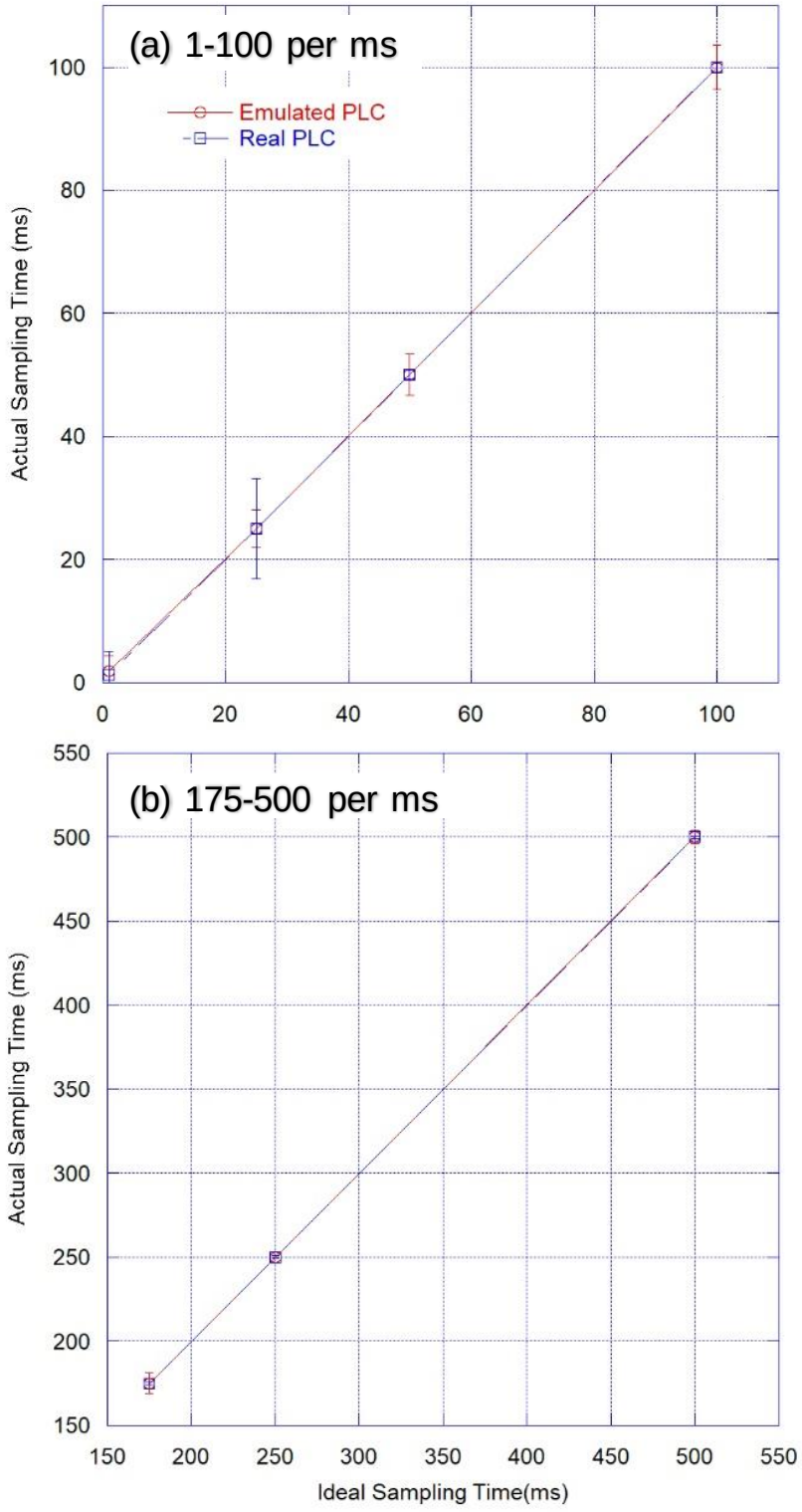

Fig. A.6: Sampling times of real and emulated PLC.

The ideal response is compared against the generated responses of the real or emulated PLC to quantify any time lag due to network communication or differences in PLC computational time. First, the state variable signal received by the PLCs is compared to the original signal generated by the Simulink model to verify that the real and emulated PLCs are receiving the same and correct input data. These tests are performed with a discrete simulation time step of 50 ms, PLC sampling time of 250 ms. The results show that the real and emulated PLCs, for all tests, receive the same square wave signal generated by the Simulink model. The controller response signal transmitted by the connected PLC is recorded within the running Simulink simulation and compared to the ideal baseline signal to identify actuation response time lag. This time lag in the PLC response is unavoidable because the python interface code is asynchronous with the PLC.

The actuation responses for the emulated and real PLCs are shown in Fig. A.7. As expected, the controller response for both the emulated and real PLCs consistently lags the ideal response signal. The sampling times for both PLCs are highly consistent and are near real-time to within ~ ±3 ms of the used PLC sampling time of 250 ms. The python interface code's real-time sync function is also consistent but deviates from real-time by up to ±50 ms (Fig. A.3). When the Simulink simulation is running slower than real-time, relative to the PLC, it is possible for the real-time deviations to accumulate, such that the recorded response of the PLC appears to be less than the sampling time. This non-physical result is only applicable when time dependent processes are being simulated and communicating asynchronously with external devices. Further improvements in the time-synchronization routine would be expected to reduce or eliminate this effect.

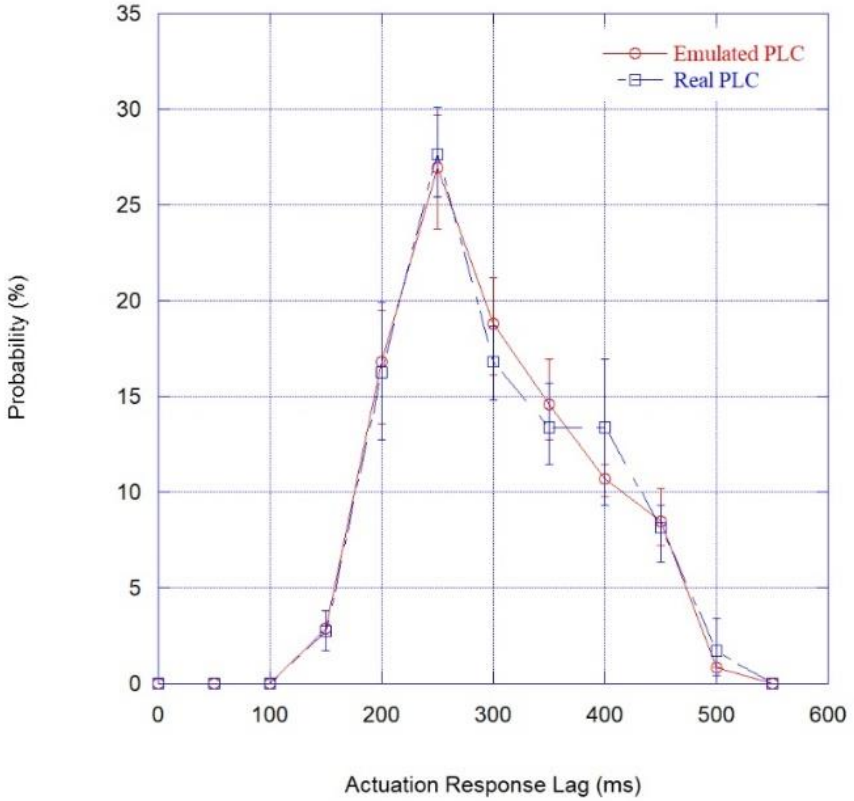

Fig. A.7: Comparison of actuation response times of real and emulated PLCs.

For the purposes of this report, although a non-physical result is observed from the perspective of the simulated process, the relative actuation response between the real and emulated PLC are in good agreement. The emulated PLC has a slightly faster actuation response time with an average response time of ~280.59 ms, compared to an average response time of ~284.06 ms for the real PLC. The results suggest that each controller can be used interchangeably in cyber-physical emulation experiments without compromising the outcomes of the physical process being controlled regarding the actuation response time.

As noted in Section A.4.1, the inter-process python data transfer code used in this analysis is asynchronous with the Simulink simulation. This is for better validation comparison between the real and emulated PLCs. The delay in the actuation response signal generated by the OpenPLC programming can be characteristic to the PLCs computing hardware and software. A synchronous inter-process communication method could be employed to eliminate the time lag between the observed and ideal actuation response signals by ensuring that the Simulink simulation waits for the PLC control signal before continuing to the next simulation timestep. The minimum simulation timestep is still limited however, by the network response time for the communication between the PLC and server PC running the process simulation model. A synchronous interface code would also have the added advantage of being able to run faster than real-time.

A.4.6 Results Summary

The analyses of the network response shows that the emulated PLC has a slightly higher bandwidth capacity relative to the real PLC, however this was found to not have a significant impact on the relative communication speeds relative to the real PLC. The difference in bandwidth did influence the recorded network response and traffic, with the emulated PLC transmitting slightly more data packets than real PLC. Although the total number of packets was found to be different, the PCAP analyses showed that the emulated PLC generated the same type of network traffic as observed using the real PLC. When looking at the relative proportions of the data packets the real and emulated PLC did not have a difference of greater than 1% between TCP and UDP packet transmission.

Table A.5: Comparison of digital and physical signature test data for real and emulated PLCs.

PLC	Signature		Average (ms)	STD (ms)
Real	Network Response	QR	1.78	3.19
		RQ	21.87	9.04
	Network Traffic	TCP/IP	25666.60	1946.36
		UDP/IP	35.20	8.15
	Sampling time		249.99	0.23
	Actuation Response Time		284.06	79.03
Emulated	Network Response	QR	0.73	0.61
		RQ	23.56	8.75
	Network Traffic	TCP/IP	27455.40	1266.14
		UDP/IP	43.60	16.88
	Sampling time		250.00	1.12
	Actuation Response Time		280.59	77.10

Validation analyses investigating the sampling time of OpenPLC Runtime found that on average the sampling time set in OpenPLC matches the ideal sampling time. For sample times <

250 ms however, the variance of the sampling time is significant for the emulated PLC. The smallest difference in the sampling times of the emulated and real PLC systems occurs for a sampling time of 250 ms.

Finally, the actuation response time and sampling time are also characterized for the real and emulated PLC. The validation testing analyses shows that the difference between actuation signal response times of the real and emulated PLC agree to within 2% (Table A.5). Nonphysical actuation response times are also observed due to the real-time implementation of the simulated process using the python interface code and the asynchronous communication with the external PLCs. This result suggests that future cyber-physical emulation experiments using asynchronous communication with time dependent processes should ensure that the simulated process runs in real-time ensuring that the cumulative error is never greater than one major time step.

A.5. Summary and Conclusion

This work updated and tested the developed PLC emulation methodology. This methodology is used to validate an emulation of a Raspberry 4, as the hardware for a PLC running the open-source OpenPLC program. The physical and digital signatures within the framework of the PLC emulation methodology are characterized for development of the PLC emulation for OT cybersecurity testing. Although this work investigates the implementation of an opensource PLC, the developed emulation methodology can be applied to industry representative PLCs.

The validation testing analyses demonstrate that the actuation responses and sampling rates of the emulated PLC are practically indistinguishable from those of the real PLC to the process simulation being controlled. From a cybersecurity perspective, the emulated PLC runs the same software, communicates using the same communication protocols, and generates the same types and proportions of network traffic data types as the real device.

For cybersecurity applications, the developed PLC emulation methodology confirms: (a) the importance of selecting the proper configuration parameters to ensure that the emulated PLC behaves comparable to the real system, (b) the need for detailed characterization and comparison of the physical and digital signatures, as performed in this work, (c) the need to pay special attention to the time dependencies of the PLC, such as the sampling rate or how the PLC interfaces with the simulated process. This would ensure representative behavior of the process being modeled. The developed PLC emulation is employed within the NICSim Platform for modeling the PLCs of a representative PWR plant I&C systems for future cybersecurity research.