University of Stavanger

Faculty of Science and Technology

BACHELOR`S THESIS

Study program/Specialization: Bachelor of Science / Controls Engineering and Circuit Design	Spring semester, 2021 Open access
Writer: Vebjørn Njåtun Krøyer and Nils Stanelle	………………………………………… (Writer's signature)
Faculty supervisor: Damiano Rotondo External supervisor(s): Frank Rørtvedt (Siemens AS)	
Thesis title: Control of upscaled silicon coating process in battery cell production line with digital twin	
Credits (ECTS): 20	
Key words: Digital Twin, Automation, Control Theory, PID-Control, Modeling, Matlab, SIMIT, PLC and HMI programming.	Pages: 109 + enclosure: 68 Stavanger, 15. Mai 2021 Date/year

Control of Upscaled Silicon Coating Process in Battery Cell Production Line with Digital Twin

Bachelor Thesis in Automation and Control Systems

Vebjørn Njåtun Krøyer
Nils Stanelle

Hand inn date: 15. May 2021
This report can be shared with others

University of Stavanger
Norway
May 2021

Work description

The recently established battery production company Beyonder is currently developing next generation battery technology. The development is in the R&D phase, where they are testing and preparing for the construction of a pilot factory and thereafter a full-scale battery production factory. In this thesis, we will focus on the production of resorcinol/formaldehyde-coated silicon nanoparticles used in the battery manufacturing. The goals of this project is to create a digital twin of the coating process and to develop a control system with monitoring capabilities.

Research Question
Is it feasible, with a laboratory scale process as a basis, to create and develop a control and regulation system for a planned upscaled process with the use of a digital twin?

Scope of Work

- Describe the coating process steps, functionality, and assumptions made.
- Develop digital twin of the process in collaboration with Beyonder.
  - Map to what extent it is viable to simulate the process and define critical variables.
  - Based on defined variables create a mathematical model that describes the process behavior to the extent defined.
- Design and dimension a reactor and corresponding components to a defined scale.
- Formulate a mathematical model description including:
  - Temperature behavior in the reactor with heat transfer behavior and heat loss to the ambient air
  - Concentration of resorcinol present in the tank based on flow in and rate of reaction in the tank.
  - Yield from the process in relations to tank temperature, resorcinol concentration, and time based on behavior from similar processes.
- Develop control system for the process including:
  - Create a step diagram of process logic to define inputs, outputs, and critical variables.
  - Develop PLC logic for automatic control of the coating process.
  - Control temperature and concentration with PID controller, closed-loop feedback, and tuning with the Skogestad method.
  - Develop HMI screen for manual operation of process and monitoring.
- Testing and simulations of model in combination with PLC control system, to test model correctness, and control functionality.

Supervisor: Damiano Rotondo (UiS)

Co-Supervisor: Frank Rørtvedt (Siemens AS)

Acknowledgements

This thesis was written with Damiano Rotondo at the University of Stavanger as supervisor. We would like to thank him for his great enthusiasm and encouragement when working on this project. He always had an answer for our questions and was available to guide us during the entire project period.

We would also like to thank our co-supervisor Frank Rørtvedt, who found a fitting case and scope for this project when we asked Siemens in the Autumn of 2020 if they had any exiting project we could write about. Also a big thank you to Matthias Dieter Flach who teached us, everything there is to know about Siemens‘s different platforms. We are also very grateful that Beyonder were willing to share their new and upcoming technology solutions with us, and letting us learn a lot about battery cell production.

Sincerely,

Vebjørn Njåtun Krøyer and Nils Stanelle

Stavanger, 15. Mai 2021

Abstract

The dependency on high-performing batteries is currently growing exponentially, and consequently, the battery cell manufacturing industry will have to satisfy this increasingly high demand. Beyonder is a Norwegian company that has established a way to make batteries in a sustainable manner using sawdust. Siemens supports Beyonder with digitalization and automation solutions.

The process explored in this project is the coating of silicon nanoparticles, which is one of the stages in this method of battery cell manufacturing. The coating process is in a small-scale laboratory phase during the development of this thesis, but here it will be treated as an upscaled version in order to implement control technology and control theory on a theoretical future automated process.

This bachelor thesis aims to provide a suggestion for how the currently manually operated process can be automated using a PLC. Because the process was not available during this project, it has been mathematically modeled in order to develop a digital twin that resembles the process dynamics. This digital twin was connected with the PLC to simulate the process behavior resulting from control actions. An HMI has been designed and created to give a human operator the opportunity to monitor and adjust essential process parameters. The adjustable parameters are: The temperature in the tank reactor, the concentration of resorcinol, and the coating time.

The yield of the process indicates how well it has been performed, and it is mainly dependent on the three parameters mentioned above. The parameters should be kept at levels that, combined, produce the optimal yield, because this maximizes the material utilization and increases the product quality while reducing generated waste. The temperature and concentration are both determined by the values of several physical variables that can be described by differential equations, which makes their dynamics somewhat complex. For this reason, PID controllers have been developed, and these are able to maintain the desired temperature and concentration values in spite of varying behavior and disturbances. In order to design viable and robust PID controllers, a state-space representation of the physical system was developed, to be able to generate transfer functions of the process.

The first task for this project was to analyze the process and create a step diagram that visually represents the sequence of steps carried out by the process. After that, the system dynamics were dissected in order to represent the system behavior mathematically. The digital twin created in the simulation software SIMIT was tested against the mathematical modeling, and the results verified its accuracy. After that, a PLC was programmed to control the simulated process. When designing PID controllers, the state-space model was used to calculate the correct PID parameters. While this worked well for the temperature control, it proved inadequate for the concentration control. For this reason, the integrated PID tuning functionality in the TIA Portal was used. When testing the control system against the simulated process, the PID controllers were able to maintain the desired values for temperature and concentration.

Abstract - Norwegian

Etterspørselen etter batterier med god ytelse vokser eksponentielt i nyere tid, og som en konsekvens av dette, blir batteriprodusentene nødt til å tilfredsstille det økende behovet. Beyonder er et norsk selskap som har utviklet en metodikk for å produsere batterier på en bærekraftig måte ved hjelp av sagspon, og de får hjelp av Siemens innenfor digitalisering og automatiseringsløsninger.

Prosessen som utforskes i dette prosjektet er belegging av silikon nanopartikler, som er ett av stegene i denne produksjonsmetodikken av battericeller. Beleggingsprosessen er, mens denne rapporten blir skrevet, på en laboratorieskala. Den blir derimot i dette prosjektet behandlet som en oppskalert versjon for å kunne implementere styrings- og reguleringstekniske metodikker på en teoretisk fremtidig tiltenkt prosess.

Denne bachelor oppgaven ønsker å gi forslag til hvordan den nåværende manuelt opererte prosessen kan bli automatisert ved hjelp av styring. Siden prosessen ikke var tilgjengelig under prosjekttiden, ble prosessen matematisk modellert for å utvikle en digital tvilling som søker å gjenskape prosessens reelle dynamikker. Denne digitale tvillingen ble koblet opp mot PLSen for å simulere prosessoppførselen basert på styresignaler. En HMI har blitt designet og utviklet for å gi en operatør muligheten til å overvåke og justere essensielle prosessparametere. Disse parameterene er: Temperaturen i reaktoren, konsentrasjonen av resorcinol, og beleggingstiden.

Utbyttet av prosessen tilsier hvor godt den fungerer, og dette er, i hovedsak, avhengig av de tre parameterene nevnt ovenfor. Disse parameterene skal bli holdt på et nivå, som kominert, gir den høyeste utnyttelsen, ettersom det maksimerer forbruket av materialer og øker produkt kvaliteten, samt genererer mindre avfall. Temperaturen og konsentrasjonen, er begge bestemt av en rekke fysiske variabler, som kan bli beskrevet av differensialligninger, som gjør systemet noe komplekst. Derfor har PID-kontrollere blitt utviklet for å holde temperaturen og konsentrasjonen på et gitt nivå uavhengig av varierende oppførsel og forstyrrelser. For å designe egnede og robuste PID-kontrollere ble en state-space representasjon av det fysiske systemet utviklet, for å kunne regne ut overføringsfunksjoner som bestemmer hvordan systemet reagerer på et pådrag.

Den første oppgaven med dette prosjektet var å analysere prosessen og utvikle et tilstandsdiagram, som visuelt representerer sekvensene i prosessen. Deretter, ble systemets dynamikk segmentert for å representere system oppførselen matematisk. Den digitale tvillingen, utviklet i simulering softwaren SIMIT, ble testet mot den matematiske modellen og resultatene verifiserte den digitale tvillingen. Etter dette ble en PLS programmert for å styre den simulerte prosessen. Da PID-kontrollerene ble designet, ble state-space modellen brukt for å regne ut overføringsfunksjoner, som ble brukt til å bestemme PID-parameterene. Selvom dette fungerte godt for reguleringen av temperaturen, viste det seg å være utilstrekkelig for regulering av konsentrasjonen. Derfor ble PID-tuning metodikken integrert i TIA-portalen brukt. Da reguleringssystemet ble testet mot den simulerte prosessen, var PID-kontrollerene gode nok til å holde temperaturen og konsentrasjonen på et ønsket nivå.

Contents

List of Figures

Nomenclature

Variable	Unit	Definition
$q_i(t)$	$[\frac{m^3}{min}]$	Volumetric flow
$\Delta P(t)$	$[Pa]$	Pressure drop
$h(t)$	[m]	Height in Tank
$x_i(t)$	[–]	Valve position
$V_{tot}(t)$	$[m^3]$	Reactor volume
$E(t)$	[J]	Thermal energy in the tank reactor
$E_h(t)$	[J]	Thermal energy in the heating element
$Q_{conv}(t)$	[W]	Energy gained form convection
$Q_{power}(t)$	[W]	Energy gained form the heating element
$Q_{inlet}(t)$	[W]	Energy gained form the inlet valves
$Q_{outlet}(t)$	[W]	Energy lost by the outlet valve
$Q_{heat,loss}(t)$	[W]	Energy lost to ambient air
$T_{reac}(t)$	$[°C]$	Temperature in the reactor
$T_h(t)$	$[°C]$	Heating element temperature
$T_{i,in}(t)$	$[°C]$	Temperature of the individual compounds
$T_{Ambient}(t)$	$[°C]$	Temperature of the ambient air
$P(t)$	[W]	Power on the heating element
$m_{tot}(t)$	[kg]	Mass in the reactor
$c_{p,tot}$	$[\frac{J}{kg \cdot °C}]$	Specific heat capacity of the contents of the reactor
$w_i(t)$	$[\frac{kg}{min}]$	Mass flow
$A_{reac}(t)$	$[m^2]$	Area of the reactor
$r_{res}(t)$	$[\frac{g}{m^3}]$	Reaction rate
$C_{res}(t)$	$[\frac{g}{m^3}]$	Resorcinol concentration
$k(t)$	$[min^{-1}]$	Temperature dependency of the reaction rate
t	[min]	Time
Y	$[\frac{g}{m^3}]$	Yield
$H(s)$	[–]	Transfer Function

Constant	Unit	Definition
$K_{v,i}$	$[\frac{m^3}{h \cdot bar}]$	Flow coefficient
SG_i	[–]	Specific gravity
g	$[\frac{m}{s^2}]$	Gravitational acceleration
ρ	$[\frac{kg}{m^3}]$	Density
S	$[m^2]$	Base of the reactor
m_h	[kg]	Mass
$c_{p,h}$	$[\frac{J}{kg \cdot {}^\circ C}]$	Specific heat capacity of the heating element
$c_{p,i}$	$[\frac{J}{kg \cdot {}^\circ C}]$	Specific heat capacity of the individual compounds
A_h	$[m^2]$	Area of the heating element
h_h	$[\frac{J}{m^2 \cdot {}^\circ C \cdot min}]$	Convective heat transfer coefficient
h_{reac}	$[\frac{J}{m^2 \cdot {}^\circ C \cdot min}]$	Convective heat transfer coefficient
$C_{res,in}$	$[kg/m^3]$	Resorcinol concentration on the inlet
A	$[min^{-1}]$	Pre-exponential factor
E_a	$[\frac{J}{kg}]$	Activation energy
R	$[\frac{J}{kg \cdot K}]$	Universal gas constant
r	[m]	Reactor radius
K_p	[–]	Proportional gain
T_i	[–]	Integral time
T_d	[–]	derivative time
T_C	[min]	Time Constant
T_{j0}^{inv}	[min]	Transfer function numerator time constant
τ_{i0}	[min]	Transfer function denominator lag time

Acronym/abbreviation	Definition
CSTR	Continuously stirred-tank reactor
PLC	Programmable Logic Controller
HMI	Human Machine Interface
PID	Proportional–Integral–Derivative
TIA Portal	Totally Integrated Automation Portal
EU	European Union
RPM	Rounds Per Minute
RF	Resorcinol-Formaldehyde
TEM	Transmission Electron Microscope
SCL	Structured Control Language
FB	Function Block
DB	Data Block
OB	Organization Block
NO	Normally Open
TF	Transfer Function
WP	Working Point
CPU	Central Processing Unit
LAD	Ladder
DN	Diameter Nominal

Chapter 1

Introduction

This chapter is the introduction of the thesis. Firstly, we explain the motivation for the project, our contributions towards it, and finally, we present the outline of this report.

1.1 Motivation

We live in a time where the climate and environment are more important than ever. Therefore, companies and governments worldwide are investing heavily in renewable, green, and sustainable solutions. The result is a mass migration from fossil energy to electrical solutions. This global shift leads to an enormous demand for energy storage solutions, with application areas being electrical mobility, energy storage, and industry. Based on a report from World Economic Forum [28], the global battery demand is estimated to increase by nineteen times from 2018 to 2030, as shown in Figure 1.1.1. The most immense estimated growth is for electric mobility. The is because many countries are implementing regulations to make it more affordable to use electric vehicles over fossil to reach their CO_2 emission goals regarding the Paris Agreement, [25] , which states each country's contribution towards reducing their CO_2 emissions.

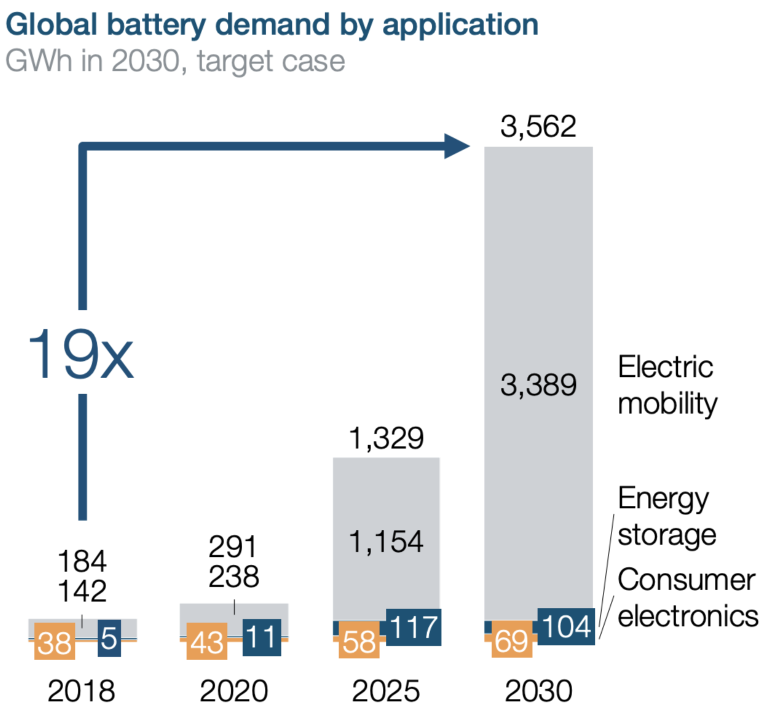


Figure 1.1.1: Estimated growth of battery demand [28]

Based on an EU report from 2016 [12], four countries were responsible for 95% of the annual manufacturing of Lithium-ion battery cell capacity. China has the largest share with 51%, followed by South Korea with 21%, then Japan with 16% and the United States with 7%. The EU only accounts for about 2% of the production. This imbalance is something the EU wants to change, to not be dependent on these countries for battery solutions. To battle this dependency, the EU launched the European Battery Alliance in 2017. The EU also wants to contribute to a green and sustainable way of producing batteries throughout the entire battery value chain. This focus is due to especially environmentally harmful segments of the value chain, such as the mining of cobalt or other minerals. This alliance aims to "develop an innovative, competitive and sustainable battery value chain in Europe" [4]. There are already several companies that have joined this alliance. The complete list of which companies can be found here.

Norway is in an ideal position to produce Li-ion batteries based on the European Battery Alliance goals. Regarding sustainability, Norway is in a perfect place due to the access to low-cost clean hydro-based energy and the cold climate. Norway is also trying to be less dependent on the oil and gas industry. Therefore, several private and government-owned businesses want to invest in these industries, preferably in those that contribute towards Norway's CO_2 emission goal. Due to these factors, several battery-based companies surfaced in Norway: Freyr, with a mega factory planned in Mo i Rana [19]; Morrow batteries, with a factory planned in Arendal; and Beyonder, with a factory planned in Stavanger. Other companies are also investing in other parts of the battery industry to create a green value chain.

Beyonder is situated in the ideal position for this huge demand for environmental battery technology. With a focus on sustainability and energy efficiency, they have developed an eco-friendly battery solution with excellent performance, fast charging time, and high recharge capabilities. By using sawdust to create activated carbon, they reduce the carbon footprint on other parts of the battery value chain [1]. Siemens have partnered with Beyonder to aid with industrial experience to support development and implementation of sustainable production with reduced energy and resource consumption while optimizing product quality and yield. In addition the parties intend to explore market opportunities with the goal of utilizing Beyonder products in Siemens energy storage and smart grid solutions [2]. Siemens has supported the battery production industry for years and gained plenty of insights in regards to battery production through its value chain [18]. Siemens in Norway and the Nordics have established a team that supports the industry with domain knowledge, best practices and technology expertise to enable start-ups and scale-ups to manage the complex challenges using automation and cutting-edge digitalization such as digital twins, IIoT, edge computing and AI [26].

We can see an industry that is growing rapidly with an entire continent that commits to it. With this industry marching forward, we see a golden opportunity to gather knowledge and competence about the battery industry to situate ourselves for the future working market better. For this reason, we wanted to learn more about battery production, its automatizing, and the control of these processes.

1.2 Contributions

The goals were to develop a digital twin that represents the actual process behavior, a control system to control the sequential process, and controllers to regulate the essential parameters of the process.

To do this, we described the process mathematically and implemented the model in SIMIT to get a simulator that would explain how the behavior of the process depends on different parameters. We developed the control system in the TIA Portal based on a step diagram to ensure that the control system is easily maintainable and understandable for people doing maintenance or changes in the future.

It was wanted to investigate the possibilities of using the digital twin to give Beyonder a indication of what type of issues they may come over when the laboratory scale process is eventually upscaled and also gain some knowledge about the process before it is built.

1.3 Outline

The thesis is structured as follows:

Chapter 2 describes the components in the process, the process functionality, and the different steps in the process.

Chapter 3 describes the mathematical modeling of the reactor.

Chapter 4 describes the overall system architecture.

Chapter 5 describes the implementation of the model in SIMIT.

Chapter 6 describes the development of the PLC system, regulation of the temperature and the concentration, and development of the HMI screen.

Chapter 7 describes the control of the temperature and the concentration.

Chapter 8 describes tests and simulations of the SIMIT implementation, PLC, and HMI system.

Chapter 9 describes the main conclusions and outlines possible future work.

Appendix A shows the adjustment of a yield model to better suit our needs.

Appendix B describes the design of the different process components as, valves, reactor tank, and heating element.

Appendix C describes the development and calculations done to create a state-space representation of the system.

Appendix D shows the developed code for PLC control in the TIA Portal.

Appendix E shows the developed charts for the model in SIMIT.

Appendix F shows the developed Matlab script used to create plots and calculate the transfer functions based on the state-space model.

Chapter 2

Process Description

In this chapter, the structure of the process will be explained, the overall functionality, how it can be separated into three main phases, the steps that completed in each of these parts, and which parameters are essential for the process.

2.1 Process structure

The structure of the process is comprised of a closed-tank reactor, seven external tanks each containing one of the following compounds: H_2O, EtOH, CTACl, SiO_2, resorcinol, ammonia, and formaldehyde, eight valves to control the inlet flows from the external tanks into the reactor and the outlet flow from the reactor, one heating element placed inside the reactor to heat the contents of the reactor, a sonicator to agitate the content of the reactor to make particles disperse better, and an impeller to mix the compounds. An illustration of the reactor is shown in Figure 2.1.1.

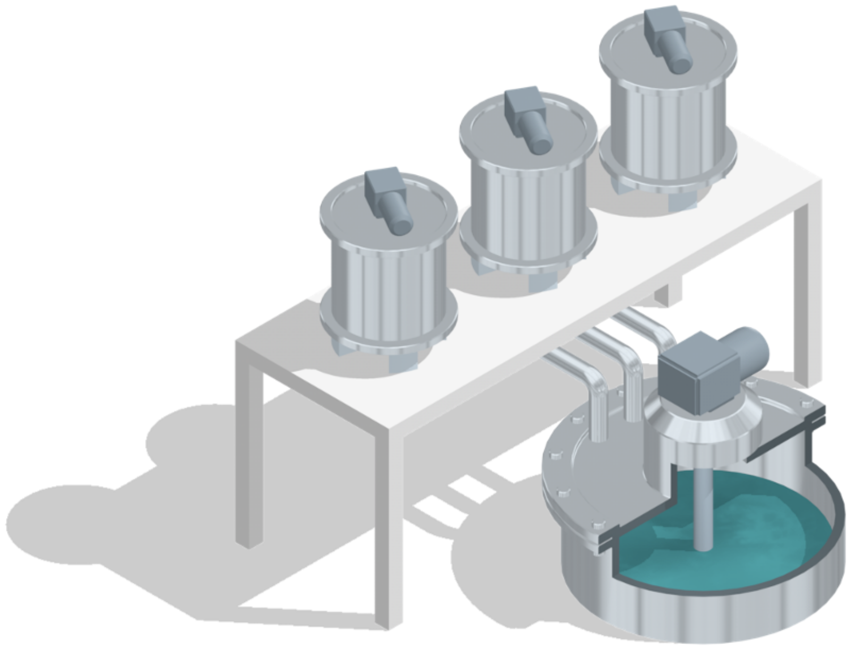


Figure 2.1.1: Illustration of reactor with external tanks

2.2 Functionality

The process is separated into three main phases: preparation of the solution, coating of silicon nanoparticles, and cleaning of the final product and waste removal.

Preparation of solution: The process starts with adding water and ethanol to the reactor. CTACl is then dissolved into the mixture. After stirring the solution at 400 RPM for 15 minutes, the silicon nanoparticles are added to the solution. This solution is then sonicated for 30 minutes to disperse the nanoparticles in the reactor. The mixture with the dispersed nanoparticles is then heated to 70 °C while stirred vigorously at 400 RPM. At this point, the solution is ready for the coating segment.

Coating of the silicon nanoparticles: After the solution is prepared, ammonia and resorcinol are added continuously to maintain a steady concentration during the coating. Where ammonia acts as the catalyst in the process. The catalyst serves to reduce energy needed to make the compounds react, also known as the activation energy (E_a) Next, one addition of formaldehyde, i.e., three times the amount of one resorcinol addition, is added to the solution. These two steps are repeated several times depending on the desired coating thickness, either 100 nm or 500 nm. The coating thickness increases by 100 nm per repetition. In the end, another addition of formaldehyde is added. The total amount of formaldehyde added in the tank can be described as $m_{form} = (m_{res} \cdot 3) \cdot N + (m_{res} \cdot 3)$, where m_{form} describes the total mass of formaldehyde added during the process, m_{res} is the total mass of added resorcinol, and N is the number of 100 nm coatings.

Cleaning and waste removal: The compound is then rested overnight to allow the particles to settle. Afterward, the supernatant is poured out of the tank, and the coated silicon nanoparticles remains. Water is then added, and the particles are washed for two hours by boiling the water at over 100 °C. The water is then drained and filtrated. Hereafter, the coated nanoparticles are dried overnight at around 70 °C. This is the last step of the process, after which the finished product (the coated silicon nanoparticles) is ready. The yield of the process is evaluated by weighing the product and comparing the measured weight with the maximum yield mass possible. The following equation defines the maximum yield mass:

$$m_{CNP} = 1.44 \cdot m_{res} + m_{SiO2} \tag{2.1}$$

where m_{CNP} is the mass of the coated nanoparticles, m_{res} is the mass of resorcinol added to the reactor, and m_{SiO2} is the total mass of SiO_2 nanoparticles added to the reactor. The process aims to obtain a yield of over 95 %. A high yield also means a low amount of waste, which is beneficial not only because of the economic gains but also for the lower environmental impact. Coating thickness should be even across the particles. RF should only form on SiO_2, and no SiO_2-free RF particles should be present. The coating quality is evaluated by TEM (Transmission Electron Microscope), a process that can show structural details as tiny as individual atoms, though it is very time-consuming.

2.2.1 Important Parameters

Temperature: A higher temperature increases the rate of polymerization and decreases the reaction time. The reaction time at 70°C is ∼100nm/h, versus ∼100nm/d at 20°C. The viscosity is also lower at higher temperatures.

Solvent concentration: Increased concentrations of reagents (SiO_2NPs, R, F) relative to solvent increase the viscosity and the risk for flocculation/gel formation. On the other hand, low concentrations increase waste.

Surfactant (CTACl) concentration: Below a particular surfactant concentration, RF will not form on SiO_2NPs, preferring to form SiO_2-free spheres. Coating evenness is also negatively affected. Large concentrations of surfactant increase viscosity as well as inhibit coating. Surfactant is toxic, and its usage should be decreased as much as possible.

Stirring speed: At low stirring speeds, the particles may contact each other long enough to bind together and even gel. Higher stirring speeds decrease the flocculation of particles, while very high stirring speeds can cause foam formation.

Feed rate: Above a specific feed rate, the coating will not form evenly, and particles of SiO_2-free RF will form. Experiments show that a feed rate of 4g/24h at 20°C will yield uneven particles and SiO_2-free RF spheres.

2.2.2 Process adjustments

In this project, the basis is a small-scale laboratory experiment. Meaning there were some adjustments that needed to be made to scale up the process and the implementation of functionalities that are not yet present in the current laboratory process. One of the changes that were made was scaling every mass value up by a factor of 1000. For example, the amount of CTACl added in the process is 30 g in the laboratory experiments, but in this project, 30 kg will be used.

Another adjustment that was made was the method resorcinol was added. In the laboratory experiments, it was added in bulk and left to polymerize for one hour. In this project, it was wanted to add this at a constant flow, hence, making the concentration of resorcinol in the reactor maintain a desired value. To compensate for disturbances in the concentration, a PID controller was designed. Consequentially, the resorcinol needed to be a liquid and not a solid. Therefore, a resorcinol solution was used, containing 13.33 L of a 60/40 mixture of H_2O and EtOH per kg of resorcinol.

2.3 Step Diagram

To translate the functionality into control actions and triggers, a step diagram of the process was developed. This diagram separates the functionality sequence into different steps and triggers. The steps are the actions that are completed in each circle, and the triggers are the conditions that need to be satisfied to move from one step to another. The step diagram makes it easy to gather an overview of the different components and signals needed for the control system design and model requirements. Moreover, it will work as the basis for the development of the PLC control system. The three different sections of the diagram represent the different main phases of the process, the yellow frame denoting the preparation of the solution, the purple frame representing the coating of the silicon nanoparticles, and the blue frame representing the cleaning of the product and removal of waste. The trigger variables, such as "S" and "Thickness", are described in detail in Chapter 6.1, along with the actions that are completed in each step.

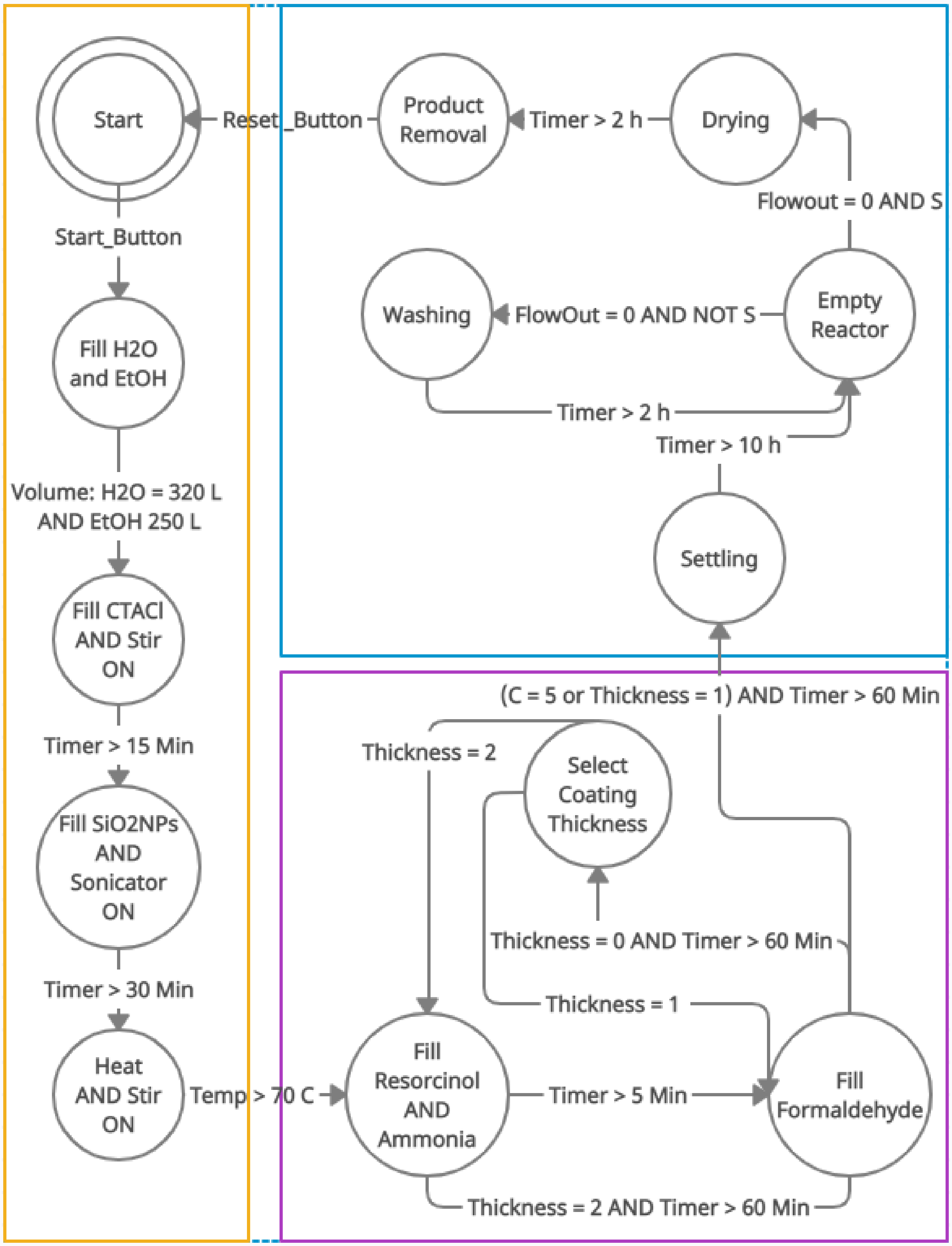


Figure 2.3.1: Developed step diagram based on desired functionality

Chapter 3

Mathematical Modeling

This chapter will describe how a mathematical model for the process was developed to make a simulated system closely resembling the actual process. Since the process is chemical in nature, looking at chemical modeling in the scientific literature was needed to characterize the system‘s dynamics. The fact that the laboratory scale process was inaccessible created the need for a precise simulated model, which could respond correctly to control actions from the PLC (Programmable Logic Controller). If the actual experimental process was readily available, step tests could have been performed for different input variables to identify the process parameters.

3.1 Flow

There was a need to model flow rates through different valves, calculating how much of a compound has flowed through the valve. This value is used to decide when the valve should be closed to ensure the correct amount of a given compound has been added to the reactor. It is also used specifically for resorcinol, calculating the concentration of resorcinol in the reactor solution, as described in more detail in Section 3.3. Ultimately, the flow rate is an essential parameter for the simulation, in addition to it being a feedback control signal that the PLC uses to decide whether to open or close the valves. In order to model the flow rate, it was decided to use the flow coefficient equation, which states the following [8]:

$$K_v = q(t) \cdot \sqrt{\frac{SG}{\Delta P(t)}} \quad \left[\frac{m^3}{h \cdot bar}\right] \tag{3.1}$$

where $q(t)$ is the flow rate through the valve, SG is the specific gravity for the compound, ΔP is the pressure drop over the valve, and K_v is the flow coefficient. The equation is modified further according to a book on chemical process dynamics [13]. An index i is added to account for the several valves and flow rates associated with the different tanks.

$$q_i(t) = K_{v,i} \cdot f(x_i(t)) \cdot \sqrt{\frac{\Delta P_i(t)}{SG_i}} \qquad \left[\frac{m^3}{h}\right] \tag{3.2}$$

where $f(x_i(t))$ is the valve characteristic, and $x_i(t) \in [0, 1]$ is the normalized position of the valve.

The pressure drop over the valve, $\Delta P_i(t)$, can be defined as follows:

$$\Delta P_i(t) = \rho_i \cdot g \cdot h_i(t) \quad [Pa] \tag{3.3}$$

where ρ_i is the density of the compound in the tank, g is the gravitational acceleration, and $h_i(t)$ is the difference between the level in the given tank and the height at which the valve is positioned with respect to the base of the tank.

Assumption 1: It is assumed that the inlet flow into each separate source tank is equal to the outlet flow from that tank, hence the levels in the external tanks are constant. However, this assumption does not apply to the main tank reactor.

Assumption 2: It is assumed that the air above the contents inside all the tanks and inside the reactor is at atmospheric pressure, this is achieved by using a pressure relief valve. Therefore, the only effect by pressure on the flow rate considered in this project is the pressure caused by the compound's level in the tank above the valve, which will cause a pressure drop across the valve. Other pressure differences are neglected.

Since the flow coefficient, K_v, is defined by the unit notation bar, the pressure drop ΔP_i needs to be converted from Pa into bar by dividing it by 100000. The time unit is also changed from hours to minutes through a division by 60, which will be suitable for this process. These adjustments lead to the final equation that expresses the volumetric flow through the valves as:

$$q_i(t) = \frac{K_v \cdot x_i(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i(t)}{100000 \cdot SG_i}} \qquad \left[\frac{m^3}{min}\right] \tag{3.4}$$

The valve characteristics $f(x_i(t))$ are set to be linear and normalized, which is why they have been replaced with just $x_i(t)$ in the equation. However, the characteristics block will be included in the simulation program regardless. In this way, the linear characteristics can be replaced easily by the characteristics of an actual valve without changing the simulation scheme in any other aspect.

3.1.1 Reactor Level

The level in the reactor is critical to model mathematically due to two factors; it affects the flow out of the reactor, and it affects the heat loss to the surrounding environment. The volume balance equation is used to describe the model mathematically:

$$\frac{dV_{tot}(t)}{dt} = q_{in}(t) - q_{out}(t) \tag{3.5}$$

where the volume in the reactor can be described as such:

$$V_{tot}(t) = S \cdot h(t) \tag{3.6}$$

where S is the base of the reactor and is constant everywhere in the reactor due to the shape being uniform, and $h(t)$ is the level in it. By replacing $V_{reac}(t)$ in Equation 3.5 with the definition above, the result becomes:

$$S \cdot \frac{dh(t)}{dt} = \sum_{i=1}^{7} \big(q_{i,in}(t)\big) - q_{out}(t) \tag{3.7}$$

where S, the base of the reactor, can be defined as:

$$S = \pi \cdot r^2 \tag{3.8}$$

where r is the radius of the reactor. Thus, the equation defining the change of height in the reactor is:

$$\frac{dh(t)}{dt} = \frac{1}{\pi \cdot r^2}\Big(\sum_{i=1}^{7}(q_{i,in}(t)) - q_{out}(t)\Big) \qquad \left[\frac{m}{min}\right] \tag{3.9}$$

3.2 Temperature

3.2.1 Heating Element

The temperature dynamics in the heating element varies based on two main factors: the power supplied to the element and the convection of temperature between the heating element and the content of the reactor. In order to model this behavior, the energy equation is used as the basis, which states that:

$$\frac{dE(t)}{dt} = Q_{in}(t) - Q_{out}(t) \quad [W] \tag{3.10}$$

where $E(t)$ is the total energy of the system, $Q_{in}(t)$ is the energy supplied to the system, and $Q_{out}(t)$ is the energy leaving the system. By replacing these variables with the variables relevant for the heating element, the following equation is reached:

$$\frac{dE_h(t)}{dt} = Q_{power}(t) - Q_{conv}(t) \quad [W] \tag{3.11}$$

where $E_h(t)$ is the total energy in the element, which can be defined as:

$$E_h(t) = m_h \cdot c_{p,h} \cdot T_h(t) \quad [J] \tag{3.12}$$

where $T_h(t)$ is the temperature of the heating element, m_h is the mass of the element, and $c_{p,h}$ is the specific heat capacity for the heating element.

$Q_{power}(t)$ is the energy supplied to the heating element and is equal to:

$$Q_{power}(t) = P(t) \quad [W]$$

$Q_{conv}(t)$ is the convective heat transfer, which is often referred to as the convection, whose dynamics is described as follows [6]:

$$Q_{conv}(t) = h_h \cdot A_h\big(T_h(t) - T_{reac}(t)\big) \quad [W] \tag{3.13}$$

where h_h is defined as the convective heat transfer coefficient between the heating element and the content in the reactor, A_h is the surface area of the heating element, and $T_{reac}(t)$ is the temperature of the content in the reactor, described with more details in Section 3.2.2.

Assumption 3: Since the heating element is placed inside the reactor, there is no heat loss from the heating element to the ambient air.

Assumption 4: The effect of aging, corrosion and temperature on the model's parameters are neglected. Therefore, m_h, A_h, h_h, and $c_{p,h}$ are constant throughout the entire process and lifetime of the element.

These assumptions, in combination with the mathematical calculations made above, lead to the following equation that describes the dynamics of the temperature of the heating element:

$$\frac{dT_h(t)}{dt} = \frac{1}{m_h \cdot c_{p,h}} \cdot \Big(60 \cdot P(t) - h_h \cdot A_h\big(T_h(t) - T_{reac}(t)\big)\Big) \qquad \left[\frac{°C}{min}\right] \tag{3.14}$$

3.2.2 Reactor

The temperature in the reactor varies in regard to the flow rate of the added compounds and each compound's specific heat capacity and temperature, in addition to the temperature of the heating element in the reactor. The basis of the equation used to model the temperature is Equation 3.10, where $E(t)$ is the total internal thermal energy in the reactor, which can be defined as:

$$E(t) = T_{reac}(t) \cdot m_{tot}(t) \cdot c_{p,tot}(t) \quad [J] \tag{3.15}$$

where $T_{reac}(t)$ is the temperature in the reactor, $m_{tot}(t)$ is the total mass in the reactor, and $c_{p,tot}(t)$ is the specific heat capacity of the content in the reactor.

$Q_{in}(t)$ is the energy supplied by other elements to the process, which for this specific process is defined as:

$$Q_{in}(t) = Q_{inlet}(t) + Q_{conv}(t) \quad [W] \tag{3.16}$$

$Q_{conv}(t)$ is defined in Equation 3.13, and $Q_{inlet}(t)$ is the sum of all the energies supplied from the inlets of compounds into the reactor. It can be described as follows:

$$Q_{inlet}(t) = \sum_{i=1}^{7} c_{p,i} \cdot w_i(t) \cdot T_{i,in}(t) \quad [W] \tag{3.17}$$

where $w_i(t)$ is the mass flow, $c_{p,i}$ is the specific heat capacity, and $T_{i,in}(t)$ is the temperature of compound i added to the reactor.

By inserting Equation 3.13 and 3.17 into Equation 3.16, the following equation is obtained:

$$Q_{in}(t) = \sum_{i=1}^{7} \Big(c_{p,i} \cdot w_i(t) \cdot T_{i,in}(t) \Big) + h_h \cdot A_h \big(T_h(t) - T_{reac}(t) \big) \quad [W] \tag{3.18}$$

$Q_{out}(t)$ is the energy removed from the process by other elements:

$$Q_{out}(t) = Q_{outlet}(t) + Q_{heat,loss}(t) \quad [W] \tag{3.19}$$

Assumption 6: It is assumed that the $T_{reac}(t)$ is equal to the temperature of the tank reactor itself, hence the thermodynamic from the reactor contents to the reactor itself is neglected.

$Q_{heat,loss}(t)$ is the convective heat transfer from the reactor to the surrounding environment. The mathematical modeling approach is shown in Equation 3.13, the variables are adjusted to account for the reactor material and area, and the ambient temperature.

$$Q_{heat,loss}(t) = h_{reac} \cdot A_{reac}(t) \big(T_{reac}(t) - T_{Ambient}(t) \big) \quad [W] \tag{3.20}$$

where h_{reac} is defined as the convective heat transfer coefficient between the tank reactor and the ambient air, $T_{Ambient}(t)$ is the temperature of the ambient air, and $A_{reac}(t)$ is the contact area between the contents of the reactor and the reactor walls. Based on the dimensions of the reactor, described in Appendix B, $A_{reac}(t)$ can be defined as follows:

$$A_{reac}(t) = 2 \cdot \pi \cdot r \cdot h(t) \quad [m^2] \tag{3.21}$$

where $2 \cdot \pi \cdot r$ is the circumference of the reactor, and $h(t)$ is the level detailed in Section 3.1.1.

$Q_{outlet}(t)$ is the energy removed by the outlet of the reactor into a waste tank. This energy can be described as follows:

$$Q_{outlet}(t) = w_{out}(t) \cdot c_{p,tot}(t) \cdot T_{reac}(t) \quad [W] \tag{3.22}$$

hence the combined equation for $Q_{out}(t)$ is:

$$Q_{out}(t) = w_{out}(t) \cdot c_{p,tot}(t) \cdot T_{reac}(t) + h_{reac} \cdot A_{reac}\big(T_{reac}(t) - T_{Ambient}(t)\big) \quad [W] \tag{3.23}$$

Assumption 7: The process is assumed to be isothermal. This means that the chemical reaction does not release or consume any energy, and therefore does not affect the reactor temperature.

With a combination of the equations 3.15, 3.18, and 3.23, it amounts to the following equation describing the temperature dynamics in the reactor:

$$\frac{d\big(m_{tot}(t) \cdot c_{p,tot}(t) \cdot T_{reac}(t)\big)}{dt} = Q_{in}(t) - Q_{out}(t) \tag{3.24}$$

Since there is a derivative of several functions, the product rule is used to get the following:

$$\frac{dm_{tot}(t)}{dt} \cdot c_{p,tot}(t) \cdot T_{reac}(t) + \frac{dc_{p,tot}(t)}{dt} \cdot m_{tot}(t) \cdot T_{reac}(t) + \frac{dT_{reac}(t)}{dt} \cdot m_{tot}(t) \cdot c_{p,tot}(t) = Q_{in}(t) - Q_{out}(t) \tag{3.25}$$

The change of mass is equal to the flow into the tank minus the flow out, which is expressed mathematically as:

$$\frac{dm_{tot}(t)}{dt} = \sum_{i=1}^{7} \big(w_{i,in}(t)\big) - w_{out}(t) \tag{3.26}$$

The same principle applies to the change of mass for individual compounds, but the flow out has to be weighted by the individual compound's mass relative to the total mass:

$$\frac{dm_i(t)}{dt} = w_{i,in}(t) - \frac{m_i(t)}{m_{tot}(t)} \cdot w_{out}(t) \tag{3.27}$$

To calculate $c_{p,tot}$, the mathematical method described in [24] was used:

$$c_{p,tot}(t) = \sum_{i=1}^{7} c_{p,i} \cdot \frac{m_i(t)}{m_{tot}(t)} \tag{3.28}$$

By taking the derivative of this, it is possible to compute the change in heat capacity of the contents of the reactor.

$$\frac{dc_{p,tot}(t)}{dt} = \sum_{i=1}^{7} \left(c_{p,i} \cdot \frac{d(\frac{m_i(t)}{m_{tot}(t)})}{dt}\right)$$

The derivation division rule is used, resulting in the following:

$$\frac{dc_{p,tot}(t)}{dt} = \sum_{i=1}^{7} \left(c_{p,i} \cdot \frac{\frac{dm_i(t)}{dt} \cdot m_{tot}(t) - \frac{dm_{tot}(t)}{dt} \cdot m_i(t)}{m_{tot}(t)^2}\right) \tag{3.29}$$

The two derivation expressions, $\frac{dm_{tot}(t)}{dt}$ and $\frac{dm_i(t)}{dt}$, are replaced with the representation of flow and individual masses described in Equations 3.26 and 3.27.

$$\begin{aligned}
\frac{dc_{p,tot}(t)}{dt} &= \sum_{i=1}^{7}\left(c_{p,i}\cdot\frac{\big(w_{i,in}(t)-\frac{m_i(t)}{m_{tot}(t)}\cdot w_{out}(t)\big)\cdot m_{tot}(t)-\big(\sum_{i=1}^{7}\big(w_{i,in}(t)\big)-w_{out}\big)\cdot m_i(t)}{m_{tot}(t)^2}\right)\\
&= \sum_{i=1}^{7}\left(c_{p,i}\cdot\frac{m_{tot}(t)\cdot w_{i,in}(t)-m_i(t)\cdot w_{out}(t)-m_i(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)+m_i(t)\cdot w_{out}(t)}{m_{tot}(t)^2}\right)\\
&= \sum_{i=1}^{7}\left(c_{p,i}\cdot\frac{m_{tot}(t)\cdot w_{i,in}(t)-m_i(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)}{m_{tot}(t)^2}\right)\\
&= \sum_{i=1}^{7}\left(c_{p,i}\cdot\frac{w_{i,in}(t)}{m_{tot}(t)}-c_{p,i}\cdot\frac{m_i(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)}{m_{tot}(t)^2}\right)\\
&= \frac{1}{m_{tot}(t)}\sum_{i=1}^{7}\left(c_{p,i}\cdot w_{i,in}(t)-c_{p,i}\cdot\frac{m_i(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)}{m_{tot}(t)}\right)
\end{aligned}$$

In the last part of the equation above, the equivalent of the derivative of the specific heat capacity in the reactor is present, this is confirmed by looking at Equation 3.28, therefore, it is replaced by $c_{p,tot}(t)$.

$$\frac{dc_{p,tot}(t)}{dt}=\frac{1}{m_{tot}(t)}\left(\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\Big)-c_{p,tot}(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)\right) \tag{3.30}$$

$\frac{dc_{p,tot}(t)}{dt}$ and $\frac{dm_{tot}(t)}{dt}$ in Equation 3.25 can now be replaced with the notations above, leading the the following equation:

$$\begin{aligned}
\frac{dT_{reac}(t)}{dt}\cdot m_{tot}(t)\cdot c_{p,tot}(t) = Q_{in}(t)-Q_{out}(t)-T_{reac}(t)\cdot\left(\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\Big)-c_{p,tot}(t)\cdot\sum_{i=1}^{7}\big(w_{i,in}(t)\big)\right)\\
-c_{p,tot}(t)\cdot T_{reac}(t)\cdot\left(\sum_{i=1}^{7}\big(w_{i,in}(t)\big)-w_{out}(t)\right)
\end{aligned} \tag{3.31}$$

by replacing $Q_{in}(t)$ with Equation 3.18, and $Q_{out}(t)$ with Equation 3.23, it results in the following:

$$\begin{aligned}
\frac{dT_{reac}(t)}{dt} = \frac{1}{m_{tot}(t)\cdot c_{p,tot}(t)}\cdot\Bigg(&\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot T_{i,in}(t)\Big)+h_h\cdot A_h\big(T_h(t)-T_{reac}(t)\big)\\
&-w_{out}(t)\cdot c_{p,tot}(t)\cdot T_{reac}(t)-h_{reac}\cdot A_{reac}\big(T_{reac}(t)-T_{Ambient}(t)\big)-\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot T_{reac}(t)\Big)\\
&+\sum_{i=1}^{7}\big(w_{i,in}(t)\big)\cdot c_{p,tot}(t)\cdot T_{reac}(t)-\big(\sum_{i=1}^{7}\big(w_{i,in}(t)\big)-w_{out}(t)\big)\cdot c_{p,tot}(t)\cdot T_{reac}(t)\Bigg)
\end{aligned} \tag{3.32}$$

$$\begin{aligned}\frac{dT_{reac}(t)}{dt} = \frac{1}{m_{tot}(t)\cdot c_{p,tot}(t)} \cdot \Bigg(&\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot T_{i,in}(t)\Big) - \sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot T_{reac}(t)\Big) \\ &- h_{reac}\cdot A_{reac}\big(T_{reac}(t) - T_{Ambient}(t)\big) + \sum_{i=1}^{7}\big(w_{i,in}(t)\big)\cdot c_{p,tot}(t)\cdot T_{reac}(t) \\ &- \big(w_{out}(t) + \sum_{i=1}^{7}\big(w_{i,in}(t)\big) - w_{out}(t)\big)c_{p,tot}(t)\cdot T_{reac}(t) + h_h\cdot A_h\big(T_h(t) - T_{reac}(t)\big)\Bigg)\end{aligned} \tag{3.33}$$

The final equation that represents the dynamics of the temperature in the reactor is:

$$\begin{aligned}\frac{dT_{reac}(t)}{dt} =& \frac{1}{m_{tot}(t)\cdot c_{p,tot}(t)} \cdot \Bigg(\sum_{i=1}^{7}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot \Big(T_{i,in}(t) - T_{reac}(t)\Big)\Big) \\ &- h_{reac}\cdot A_{reac}\big(T_{reac}(t) - T_{Ambient}(t)\big) + h_h\cdot A_h\big(T_h(t) - T_{reac}(t)\big)\Bigg) \quad \left[\frac{°C}{min}\right]\end{aligned} \tag{3.34}$$

3.3 Resorcinol Concentration

The behavior of the resorcinol concentration in the CSTR is a key part when it comes to the output yield of the process. Consequently, this behavior needs to be implemented in the model to optimize and control the process. The following equation, based on Equation 11.8 from [20], is used to describe the concentration of a general compound added to a solution:

$$\frac{d\big(C_A(t)\cdot V(t)\big)}{dt} = C_{A,in}\cdot q_{in}(t) - C_A(t)\cdot q_{out}(t) + r_A(t)\cdot V(t) \qquad \left[\frac{mol_A}{s}\right] \tag{3.35}$$

where $C_A(t)$ is the concentration of compound A, $V(t)$ is the volume in the reactor, $r_A(t)$ is the reaction rate from A $\to$ B in the reactor, $C_{A,in}$ is the concentration of compound A from the inlet, and $q_{in}(t)$ is the inlet flow rate. Using the product rule, the equation can be expressed as:

$$\frac{dC_A(t)}{dt}\cdot V(t) + \frac{dV(t)}{dt}\cdot C_A(t) = C_{A,in}(t)\cdot q_{in}(t) - C_A(t)\cdot q_{out}(t) + r_A(t)\cdot V(t) \tag{3.36}$$

where $\frac{dV(t)}{dt}$ is equal to, and is replaced by:

$$\frac{dV(t)}{dt} = q_{in}(t) - q_{out}(t) \qquad \left[\frac{m^3}{min}\right] \tag{3.37}$$

$$\frac{dC_A(t)}{dt}\cdot V(t) + C_A(t)\cdot q_{in} - C_A(t)\cdot q_{out} = C_{A,in}\cdot q_{in}(t) - C_A(t)\cdot q_{out}(t) + r_A(t)\cdot V(t) \tag{3.38}$$

$$\frac{dC_A(t)}{dt}\cdot V(t) = C_{A,in}\cdot q_{in}(t) - C_A(t)\cdot q_{in} + r_A(t)\cdot V(t) \tag{3.39}$$

$$\frac{dC_A(t)}{dt} = \frac{q_{in}(t)}{V(t)}\cdot\big(C_{A,in} - C_A(t)\big) + r_A(t) \tag{3.40}$$

Usually, a CSTR has both an inlet and an outlet flow. In this case, during the coating process, there is only an inlet flow. The equation is still suitable for the process needs because, as shown above, q_{out} gets canceled out, hence does not affect the concentration.

The volume of contents in the reactor, $V(t)$, can be modeled as follows:

$$V(t) = \frac{m_{tot}(t)}{\rho_{tot}(t)} \qquad [m^3] \tag{3.41}$$

where $m_{tot}(t)$ is the total mass accumulated in the reactor. The accumulation of mass is affected by the flow of each compound and can be represented as:

$$m_{tot}(t) = \sum_{i=1}^{7} \int_{t=0}^{t} \rho_i \cdot q_i(\tau) \quad [kg] \tag{3.42}$$

$\rho_{tot}(t)$ is the solution's total density and is defined as:

$$\rho_{tot}(t) = \frac{1}{m_{tot}(t)} \sum_{i=1}^{7} \rho_i \cdot m_i(t). \quad \left[\frac{kg}{m^3}\right] \tag{3.43}$$

The sum in equation 3.42 and 3.43 accounts for the fact that seven different compounds are added into the reactor during the process, as described in Chapter 2. This sum is a necessary detail for the inlet flow rate as well:

$$q_{in} = \sum_{i=1}^{7} q_i(t) \quad \left[\frac{m^3}{min}\right] \tag{3.44}$$

For the same reason, the concentration of resorcinol will change with time depending on whether another compound flows into the reactor simultaneously as the resorcinol, as such:

$$C_{A,in}(t) = \frac{q_{res}(t)}{\sum_{i=1}^{7} q_i(t)} \cdot C_{res,in} \quad \left[\frac{g}{m^3}\right] \tag{3.45}$$

Its important to notice that since the flow is modeled based on minutes and not seconds, this equation differs therefore from 3.35 in the time domain. By implementing the corrections made above, Equation 3.40 can be modified to represent the rate of change in concentration directly:

$$\frac{dC_{res}(t)}{dt} = \frac{1}{V(t)}\Big(C_{res,in} \cdot q_{res}(t) - C_{res}(t) \cdot \sum_{i=1}^{7} q_i(t)\Big) + r_A(t) \qquad \left[\frac{g}{min \cdot m^3}\right] \tag{3.46}$$

where $r_A(t)$ is the reaction rate and can be specified to the reaction rate for resorcinol as such [20]:

$$r_{res}(t) = -k(t) \cdot C_{res}(t) \quad \left[\frac{g}{min \cdot m^3}\right] \tag{3.47}$$

where the sign is negative since compound A is being consumed in the process and $k(t)$, the temperature dependency of the reaction rate, is defined by the Arrhenius equation, which states:

$$k(t) = A \cdot e^{\frac{-E_a}{R \cdot (T_{reac}(t) + 273.15)}} \quad [min^{-1}] \tag{3.48}$$

where A is the pre-exponential factor, which describes how often the molecules collide with the correct orientation, E_a is the energy needed to activate the reaction, R is the universal gas constant, and $T_{reac}(t)$ is the temperature in the reactor.

Assumption 8: It is assumed that the solution is perfectly mixed in the entire reactor, this means that the concentration of resorcinol in the reactor is equal to the concentration through the outlet valve. This assumption is only true if the residence time (τ) is 5 - 10 times bigger than the mixing time, $\tau = \frac{V(t)}{\sum_{i=1}^{7} q_i(t)}$ [20]. Therefore, the outflow from the reactor does not affect the concentration. This is also confirmed mathematically as q_{out} is not canceled out in the calculation of Equation A.4

Assumption 9: The stirring speed is assumed to be constant. Therefore, it does not affect the resorcinol reaction rate in the reactor.

By implementing the specified reaction rate for resorcinol into 3.46, the equation that describes the behavior of the resorcinol concentration in the reactor is equal to the following:

$$\frac{dC_{res}(t)}{dt} = \frac{1}{V(t)}\Big(C_{res,in} \cdot q_{res}(t) - C_{res}(t) \cdot \sum_{i=1}^{7} q_i(t)\Big) - A \cdot e^{\frac{-E_a}{R \cdot (T_{reac}(t)+273.15)}} \cdot C_{res}(t) \qquad \left[\frac{g}{min \cdot m^3}\right] \tag{3.49}$$

3.4 Yield

The goal of the process in its entirety is to maximize the portion of the silicon nanoparticles that are evenly coated. The relation between coated and uncoated particles is called the yield of the process, while the remaining uncoated silicon nanoparticles, and the excess compounds, are considered waste. The yield is expressed as a percentage, and an acceptable yield for this process is above 95 %.

Due to the process not being physically available during the development of this project, there was a lack of data regarding different parameters and behavior of the system dynamics. In order to simulate the yield of this process, there was a need to find a model for the yield of an arbitrary chemical process that presumably had similar behavior as the coating process depending on temperature, time, and concentration, as these are the main factors impacting the yield.

In [9], the glucose yield of rice straw was analyzed and optimized in regards to NaOH concentration, temperature, and pretreatment time. The intervals for these values were in ranges that are also reasonable for the coating process examined in this report. The following formula describes the correlation between the yield and the different parameters:

$$\begin{aligned} Y_{glucose} = & -120.384 + 38.687 \cdot \overline{C}_{NaOH} + 6.216 \cdot \overline{T}_{glucose} + 2.329 \cdot t_{glucose} \\ & - 0.13 \cdot \overline{C}_{NaOH} \cdot \overline{T}_{glucose} - 0.06 \cdot \overline{C}_{NaOH} \cdot t_{glucose} - 2.208 \cdot 10^{-3} \cdot \overline{T}_{glucose} \cdot t_{glucose} \\ & - 3.657 \cdot \overline{C}_{NaOH}^{\,2} - 0.0349 \cdot \overline{T}_{glucose}^{\,2} - 0.0174 \cdot t_{glucose}^{\,2} \end{aligned} \tag{3.50}$$

where $\overline{C}_{NaOH}$ and $\overline{T}_{glucose}$ are equivalent to the average values of the resorcinol concentration and temperature in the reactor during the step where resorcinol is added, respectively, and t is equivalent to the time from when formaldehyde has been added to when the addition of resorcinol has been stopped, as shown by $t = x$ in Figure 3.4.1.

If the coating thickness is chosen to be 500 nm, this step will be performed five times. As a result, the yield will be calculated five individual times, and the final yield of the process should then be calculated as the average of the five individual calculations:

$$Y = \frac{1}{5}\sum_{i=1}^{5} Y_i \tag{3.51}$$

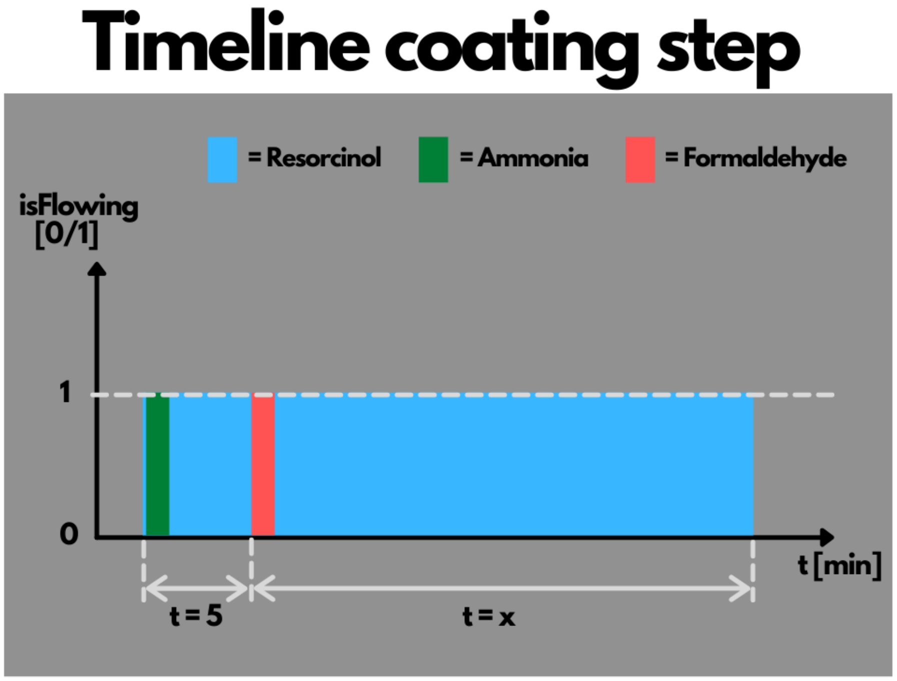


Figure 3.4.1: The timeline during the coating step of the process. (The widths of the green and the red box, and $t = x$, are not representative of their duration.)

Some refinements can be applied to Equation 3.50. There is some available data concerning the actual process, which states that when the coating process is completed with $T_{reac} = 70°\text{C}$, $t = 60$ min and 2 kg of resorcinol added during one coating step, results in a yield of 86 %. These values for temperature, time, and amount of resorcinol will work as a baseline to modify Equation 3.50.

Firstly, it was needed to translate the 2 kg added during the entirety of the 60 minutes to a concentration percentage. This was derived as follows. It is assumed no change of concentration in the coating phase since it is supposed to be constant, hence $\frac{dC_{res}(t)}{dt}$ from equation 3.49 is 0. A detailed explanation of the paper-based model's modifications is shown in Appendix A. Therein, it is explained how the yield unit notation was adjusted from $\frac{g}{kg}$ to %, and the unit notation for concentration from percent to $\frac{g}{m^3}$. The temperature and the time ranges did not need any adjustments as they were deemed within the unit range already and had the same unit notation.

The relation between the modified model parameters and the paper models parameters are calculated in Appendix A. They are as follows, $Y_{glucose} = \frac{(Y_{CNP}-70)}{0.6} + 215$, $T_{glucose} = T_{reac}$, $t_{glucose} = t_{coating}$, and $C_{NaOH} = C_{res} \cdot \frac{1}{4.5}$. When these values are implemented in the original equation from the paper Equation 3.50 the modified equation becomes:

$$\begin{aligned}\frac{(Y_{CNP}-70)}{0.6} + 215 &= -120.384 + 38.687 \cdot \overline{C}_{res} \cdot \frac{1}{4.5} + 6.216 \cdot \overline{T}_{reac} + 2.329 \cdot t_{coating} \\ &- 0.13 \cdot \overline{C}_{res} \cdot \frac{1}{4.5} \cdot \overline{T}_{reac} - 0.06 \cdot \overline{C}_{res} \cdot \frac{1}{4.5} \cdot t_{coating} - 2.208 \cdot 10^{-3} \cdot \overline{T}_{reac} \cdot t_{coating} \\ &- 3.657 \cdot (\overline{C}_{res} \cdot \frac{1}{4.5})^2 - 0.0349 \cdot \overline{T}_{reac}{}^2 - 0.0174 \cdot t_{coating}{}^2\end{aligned} \tag{3.52}$$

$$\begin{aligned}Y_{CNP} = &-131.23 + 5.158 \cdot \overline{C}_{res} + 3.73 \cdot \overline{T}_{reac} + 1.397 \cdot t_{coating} - 0.01733 \cdot \overline{C}_{res} \cdot \overline{T}_{reac} \\ &- 0.008 \cdot \overline{C}_{res} \cdot t_{coating} - 1.325 \cdot 10^{-3} \cdot \overline{T}_{reac} \cdot t_{coating} - 0.1084 \cdot \overline{C}_{res}{}^{2} - 0.0209 \cdot \overline{T}_{reac}{}^{2} \\ &- 0.0104 \cdot t_{coating}{}^{2}\end{aligned} \tag{3.53}$$

The obtained relations between the parameters can be seen in Figure 3.4.2. It is crucial to note that this yield model does not precisely represent the physical process. It only has an assumed similar behavior. The Matlab code used to create the model below is shown in F.

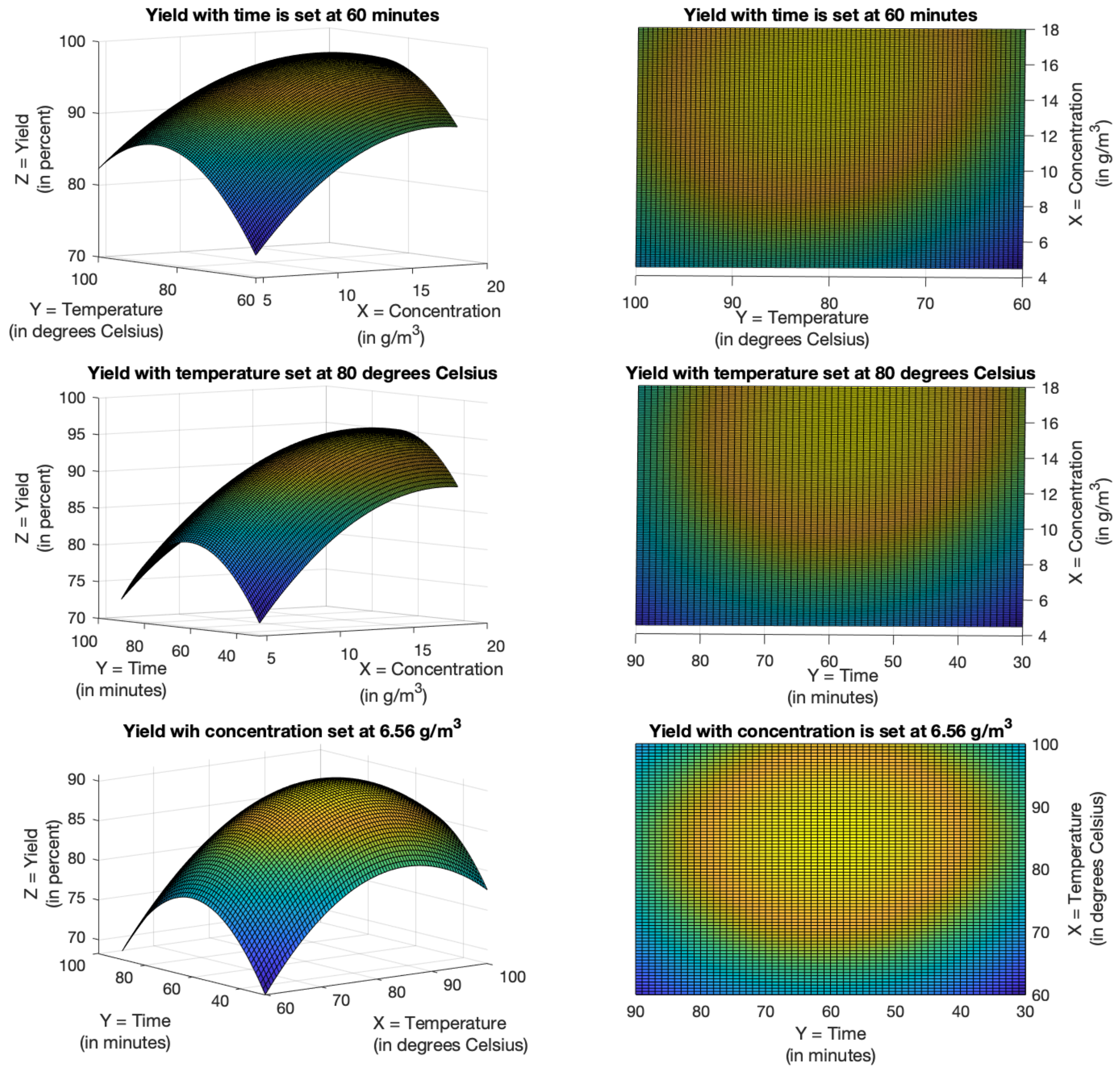


Figure 3.4.2: Adjusted relation between the three parameters and the yield

Chapter 4

System Architecture

In this chapter, the overall system architecture is presented, describing the different software and hardware components involved, and their functions.

The system was designed to handle four main tasks: control of the process, manipulate the process, monitoring of essential parameters, and simulation of the process behavior in general and behavior changing as a consequence of control actions performed.

To handle the control of the process, an S7-1500 PLC was used, and the PLC logic was programmed in SIMATIC STEP7 in the TIA Portal (v16). The TIA Portal handles the programming of the PLC, the simulation of the PLC's functionality, the development of and simulation of the HMI screens, among other capabilities. STEP7 is a comprehensive engineering tool for configuring and programming all SIMATIC controllers, like the S7-1500 chosen for this project. An HMI screen was created in WinCC Unified, which is a visualization system integrated in the TIA Portal. The WinCC Unified panel was made with the intention of allowing for manipulation and monitoring of vital parameters of the process while it is active. PLCSIM Advanced was used to simulate the PLC performance and behavior due to its great capabilities for dynamic simulations. The SIMIT model, described in Chapter 5, is connected to the PLC. This makes the simulated process change according to control actions from the PLC, and in return, the PLC will receive resulting values of the variables which need monitoring. Figure 4.0.1 shows how the four different parts are connected together.

The actual connection between SIMIT and the PLC is made in SIMIT with what is called a PLCSIM Advanced coupling. The way it works is that SIMIT will import the TIA project, which is used to find the specifications of the PLC regarding IP address and other hardware configurations. When SIMIT is initiated, it will create a virtual instance of the PLC in PLCSIM Advanced. When this has happened, the PLC project can be compiled in TIA and uploaded to this created instance, which creates the connection. For the simulated process to be controlled by PLC actions, SIMIT needs the exact variable tags created in TIA, which happens upon importing the TIA project in the SIMIT coupling editor.

MindSphere, the IoT operating system can be seen in Figure 4.0.1, but is not a part of the system architecture of this project. But it is suggested that it is a part of the future development of the system.

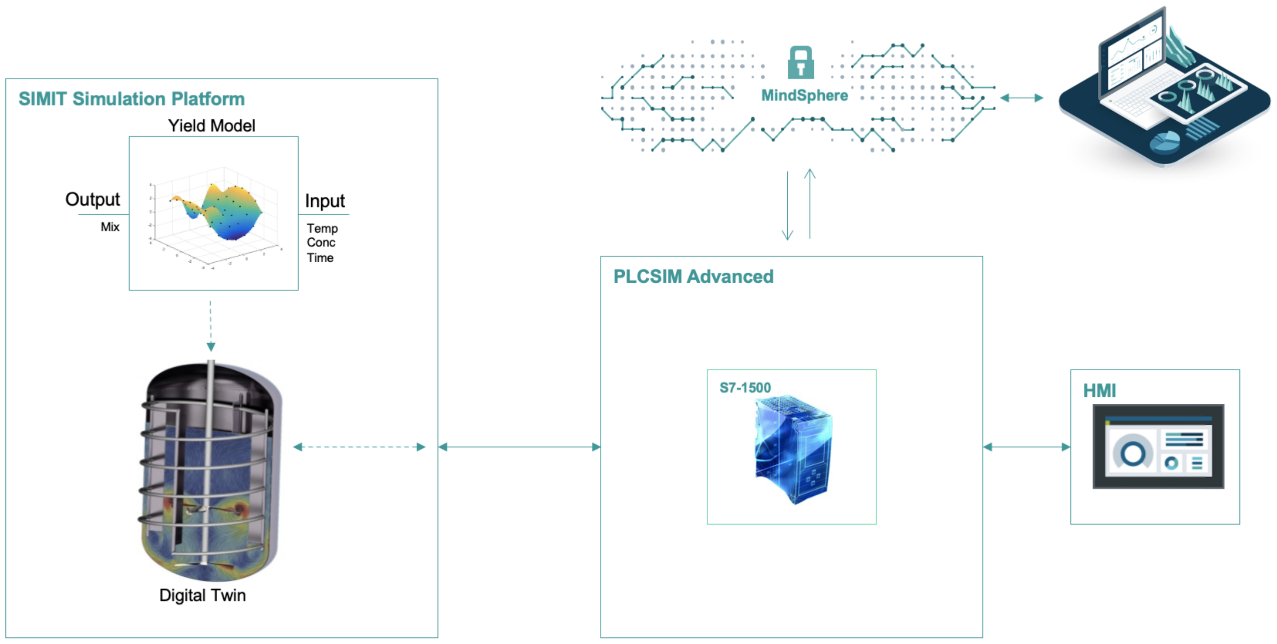


Figure 4.0.1: Overview of the system architecture and communication connections

Chapter 5

SIMIT Implementation

This chapter will describe how the mathematical model was implemented in a simulation software to create a digital twin of the process. The general purpose of the digital twin is to test the control system and see how the process responds to various inputs without the need to operate with the real process. In this way, the system behavior can be examined for a wide range of circumstances using no physical resources other than computing hardware, which means the cost of optimizing the process will be decreased significantly. To optimize the process using data from a digital twin, the mathematical model used for the simulation has to convey the system dynamics properly. This is explained in greater detail in Chapter 3. However, a model will never perfectly represent a real process due to random disturbances and varying ambient conditions.

The program used is the SIMIT Simulation Platform (V10.2), which is Siemens native simulation software. This program can simulate a complete plant with signals, devices, and plant response, in addition to other functionalities. SIMIT can also be coupled with a PLC, which ensures that the control system functionality can be tested along with the simulated process. This enables a more thorough experimentation that can expose more bugs than a standalone test of the logic in the PLC.

In SIMIT, the modeling is done in charts, which are blank graphical user interface files where different blocks can be dragged and dropped. The blocks can be connectors, arithmetic and binary operators, user-defined functions, and much more. The standard library includes many functional blocks for a wide variety of purposes. In addition to the standard library, there are extra libraries for specific use cases. After dropping several blocks into a chart, they can be connected as desired, as long as the required data type of a block's input signal is satisfied. Small green triangle arrows on the left side of blocks represent inputs, while small red triangle arrows on the right side of blocks represent outputs. The lines attached to these arrows are what interconnects different blocks. Some visual examples will be provided in this chapter, while every chart will be shown in Appendix E.

5.1 Structure

The charts have been categorized into three folders: dynamics, tanks, and variables to make the project easy to navigate. The dynamics folder includes the charts that calculate the important parameters like temperature and concentration, the tanks folder includes a chart for every compound's source tank and the reactor tank with their respective calculated flow rates, and the variables folder is used to keep the variables used in calculations in the other charts organized.

5.2 Flow Dynamics

The flow is calculated individually for each compound, as it relies on the compound's physical properties, along with the position of the valve. As such, there is a separate chart for each tank. The calculated flow rate is implemented in each chart using the modeled volumetric flow from Equation 3.4.

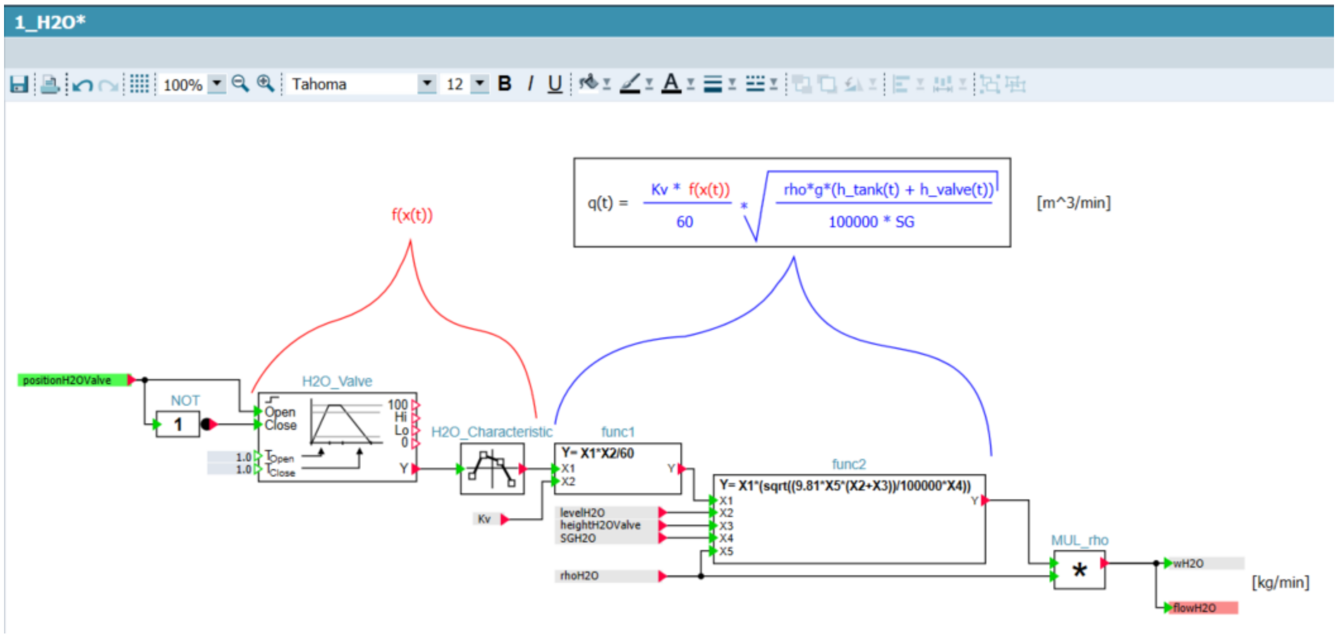


Figure 5.2.1: The modeled flow rate of water.

Figure 5.2.1 shows the whole chart calculating the flow rate based on valve position for the tank with H_2O. The charts for the other tanks look similar, except for the variables related to the different compounds, so these charts are omitted. The interested reader can find them in Appendix D. The red and blue lines pointing to the text fields at the top of the chart are only visual indicators for what the different blocks represent in regard to Equation 3.4 shown inside the top rectangle. The lines and the text fields have no technical functionality, as they are implemented only for quickly gaining a general understanding of the chart's functionality.

The green and red signals on the left and right side of the chart are signals connected to the PLC. The green signal, $positionH2OValve$, determines whether to open or close the valve. The PLC sets the value of the signal either high (TRUE) or low (FALSE). This is true for every valve with the exception of the valve for resorcinol, which is regulated. Every other aspect of the flow calculation done in this chart is also done in the other charts. The signal $positionH2OValve$ is connected to the first two inputs of the block named *H2O_Valve*. It is passed directly to the input named *Open*, while it is passed through a binary NOT operator into the input named *Close*. The NOT operation inverts the signal, meaning the value passed into the *Close* input is always the binary opposite of the value passed into the *Open* input. The block is from a SIMIT library called Drives. The output position value Y depends on the inputs *Open* and *Close*, as well as the last two inputs, T_{Open} and T_{Close}. The last two inputs specify the time taken by the valve to go from closed (0) to fully open (100) and vice versa, and they are both set to the default of 1 second.

As specified in Chapter 3, the characteristic block is included in the simulation model even though it is linear and normalized. They are included to make it easy to insert the characteristic of the actual valve. This block, named *H2O_Characteristic*, has an internal line chart that maps the input X to the output

Y. Since the characteristic is linear and normalized, the line chart only has two points with a straight line connecting them, one point for the closed position and one for the fully open position. The output of the block $H2O\_Valve$ ranges from 0-100, and this value is mapped on the X-axis of the line chart in $H2O\_Characteristic$. The output Y of $H2O\_Characteristic$ is linearly connected to the input X and scaled from 0-1, yielding a normalized output.

The output of the characteristic block is connected to a user-defined function block named $func1$. This block is the first of two user-defined function blocks that, combined, do almost the entire flow rate calculation. The calculation done across two function blocks could be done in one block, but it is separated into two blocks to make it more organized. The first function block calculates the part of Equation 3.4 leading up to the square root, while the next block, $func2$, calculates the rest of the equation. The extra inputs to these blocks are connectors that contain the values for their respective parameters, these values are defined in other charts. The output from the second function block is the calculated volumetric flow rate. After that, it is passed into the block named MUL_rho where it is multiplied with ρ, the density of the compound, to convert volumetric flow to mass flow. The mass flow is passed into a connector used in other charts in SIMIT, and it is also passed into the red signal $flowH2O$ that the PLC uses to calculate how much water is added to close the valve at the correct time.

5.3 Temperature Dynamics

The temperature of the content in the reactor varies depending on many factors. For that reason, the chart calculating the temperature is a bit convoluted, but the different main elements affecting the temperature dynamics are enclosed in rectangles of different colors to indicate which parts of the equations the blocks correspond to. In addition to blocks, the chart also includes two macros, which are like function blocks, but the user can make them.

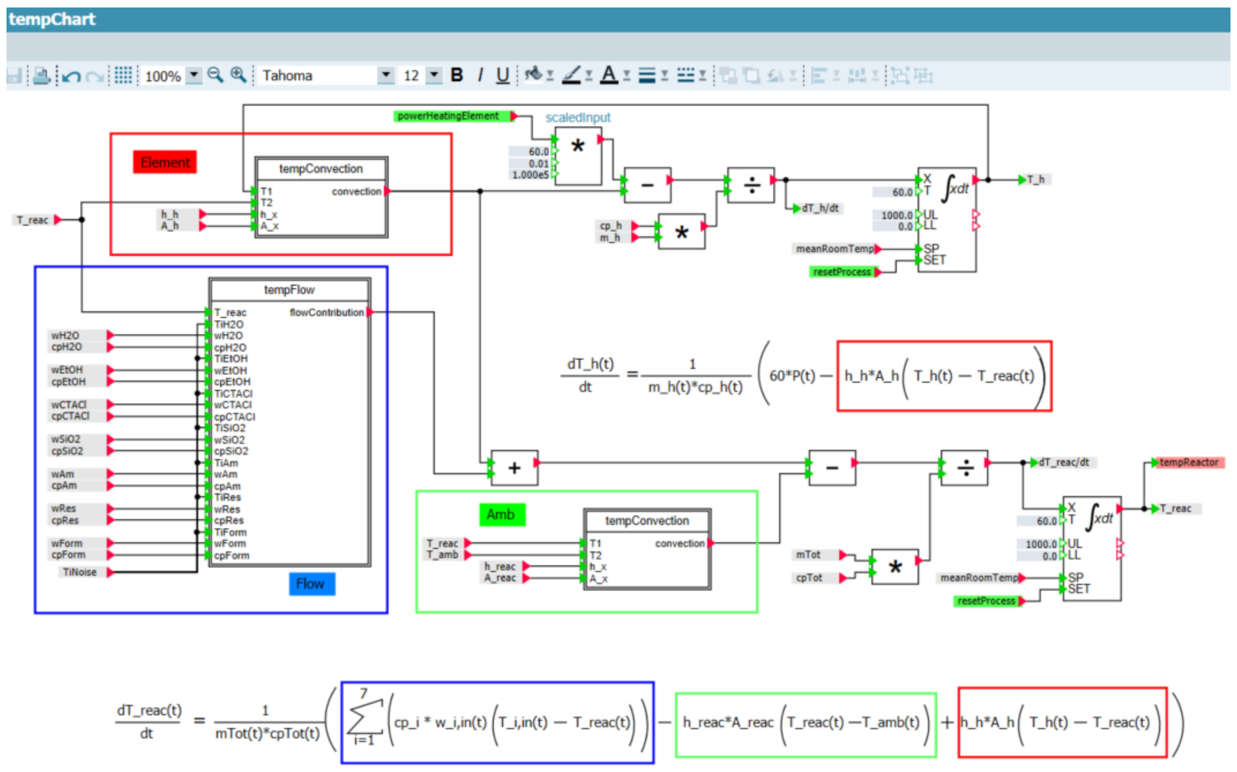

tempChart
Element
tempConvection
Flow
tempFlow
Amb
tempConvection
powerHeatingElement
scaledInput
T_reac
T_h
dT_h/dt
meanRoomTemp
resetProcess
dT_reac/dt
tempReactor
mTot
cpTot
dT_h(t)/dt = 1/(m_h(t)*cp_h(t)) (60*P(t) − h_h*A_h(T_h(t) − T_reac(t)))
dT_reac(t)/dt = 1/(mTot(t)*cpTot(t)) (Σ_{i=1}^{7}(cp_i * w_i,in(t)(T_i,in(t) − T_reac(t))) − h_reac*A_reac(T_reac(t) − T_amb(t)) + h_h*A_h(T_h(t) − T_reac(t)))

Figure 5.3.1: The modeled temperature dynamics of the heating element and the content in the reactor.

A macro will have input(s) and output(s), and the logic between the input and output is made in a macro editor, which is almost identical to the chart editor. Macros are generally used in instances when a specific series of calculations are needed in several places. The macro *tempConvection* in Figure 5.3.1 is used in two different places. This is because the method to compute the convective heat loss is the same as for calculating the convection from the heating element to the content in the reactor. The only needed difference is to connect the correct variables to the macro's inputs. The other macro, *tempFlow*, is only made to avoid cluttering the chart with an excessive amount of connecting lines. The goal was to contain every part of the temperature dynamics within a single chart while still making it readable. The blocks inside the macros will be shown separately in Figure 5.3.5 and Figure 5.3.4.

Temperature of the heating element: The top part of the chart in Figure 5.3.1 contains the heating element's temperature dynamics, while the bottom part contains the dynamics in the reactor. It is impossible to draw a single line separating these two parts because the macro regarding the convection from the heating element affects both temperatures in opposite ways. This macro is enclosed in a red rectangle in the top left area of the chart, and it is labeled with a red text field. The macro's output is split and connected into a subtraction block and an addition block, which is where the element's and reactor's temperature dynamics separate. The reason the convection is subtracted in the element's temperature dynamics and added to the reactor's dynamics is because the heat is transferred from the heating element to the reactor content. Therefore, the effect of the convective heat transfer will always have an opposite impact on the two different temperature dynamics. The differential equation describing the heating element's temperature, Equation 3.14, is included in the middle right part of the figure with a red rectangle outlining the portion defining the convection.

The convection is subtracted from the power applied to the element from the PLC, which is the green signal called *powerHeatingElement*. This signal passes through the multiplication block called *scaledInput*, where the percentage value received from the PLC is scaled to the full power range of the heating element and also multiplied with 60 to convert the time scale to minutes. The difference between the power and the convection is then divided by the product of the element's mass and its specific heat capacity, which results in the element's temperature's rate of change, dT_h/dt. The result is passed to the integrator block to get the current temperature of the element, which is sent as a feedback signal to the *tempConvection* macro. At the end of the process, the integrator's output value is reset to room temperature. The connector *meanRoomTemp* is set to the value of 22.5 °C, and the signal *resetProcess* is the boolean signal that resets the integrator.

Temperature in the reactor: The next part of the temperature chart shown in Figure 5.3.1 calculates the temperature of the content in the reactor. The equation at the bottom of the figure is the modeled Equation 3.34, with the colored outlines indicating where they appear in the chart. The connector furthest to the left, T_reac, is the feedback signal from the final calculation of the temperature. It is passed into the two different macros, *tempConvection* and *tempFlow*, because the model is a differential equation, meaning that the temperature's rate of change depends on the current temperature. The first macro, *tempConvection*, is the same as used in the temperature of the heating element, and the output is passed into an addition block, which adds it to the output of *tempFlow*.

tempFlow is the macro outlined in blue, and it contains the effect that the inlet flows of different compounds have on the reactor temperature. This macro has many inputs because the effect depends on the temperature, specific heat capacity, and flow rate of seven different compounds, in addition to the reactor temperature. The inputs for specific heat capacities are connected to connectors containing their values, and the flow rate inputs are connected to their respective calculated mass flow rates, while the temperatures of the different compounds are connected to the connector called *TiNoise*. This is a connector whose value varies randomly in the range of typical room temperatures in order to introduce some imperfection to the simulation model, because real processes are always affected by disturbances. The modeled noise is shown in the second figure displaying the temperature chart, Figure 5.3.2. After the result of *tempConvection* and *tempFlow* are added together, this sum is subtracted by the result of another instance of the *tempConvection* macro outlined in green, which represents the heat loss to the surrounding environment. The result is then divided by the product of the total mass and total specific heat capacity of the content in the reactor, *mTot* and *cpTot*, in a division block whose outcome is the reactor temperature's rate of change. Like the heating element, the rate of change is integrated to get the current temperature, which is reset to the value of *meanRoomTemp* by the *resetProcess* signal from the PLC. The red signal, *tempReactor*, is a signal sent to the PLC in order to monitor the temperature.

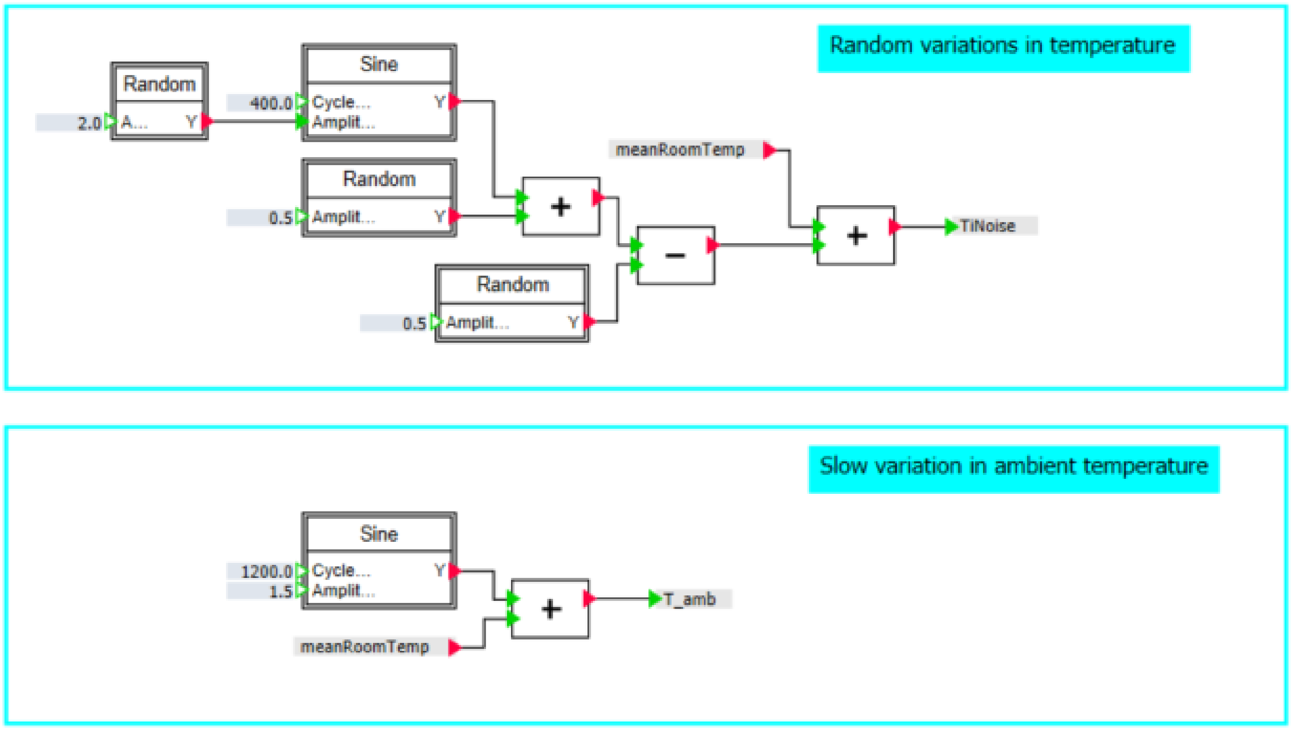


Figure 5.3.2: The rest of the temperature chart.

The connectors *TiNoise* and *T_amb* in Figure 5.3.2 are based on pre-made macros included in the SIMIT installation. The macros in question are *Sine* and *Random*. *Sine* will generate a sine wave with cycle duration and amplitude defined by inputs, and *Random* will cyclically generate a random number between zero and the value specified by the input. *T_amb* represents the ambient air temperature, set to vary as a sine wave with a cycle duration of 20 minutes. The amplitude is 1.5, and it is added to *meanRoomTemp*, meaning $T_amb \in [21, 24]^{\circ}$C. *TiNoise* has similar behavior, but with the added randomness. The result is shown in Figure 5.3.3, where the horizontal axis represents the time in minutes, and the vertical axis represents the value of *TiNoise* in °C.

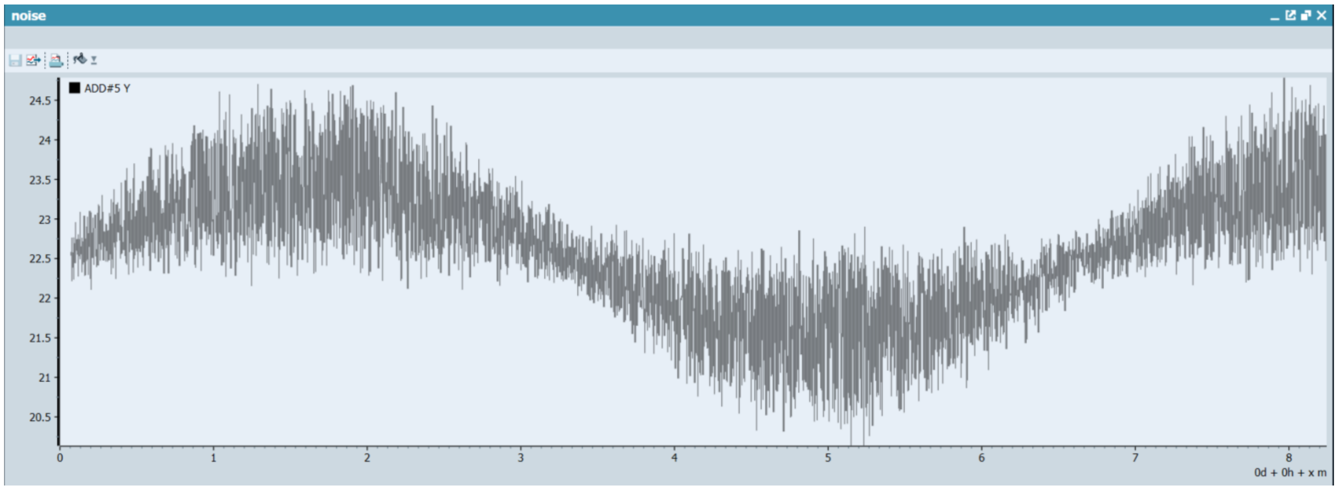


Figure 5.3.3: The temperature affected by the generated noise.

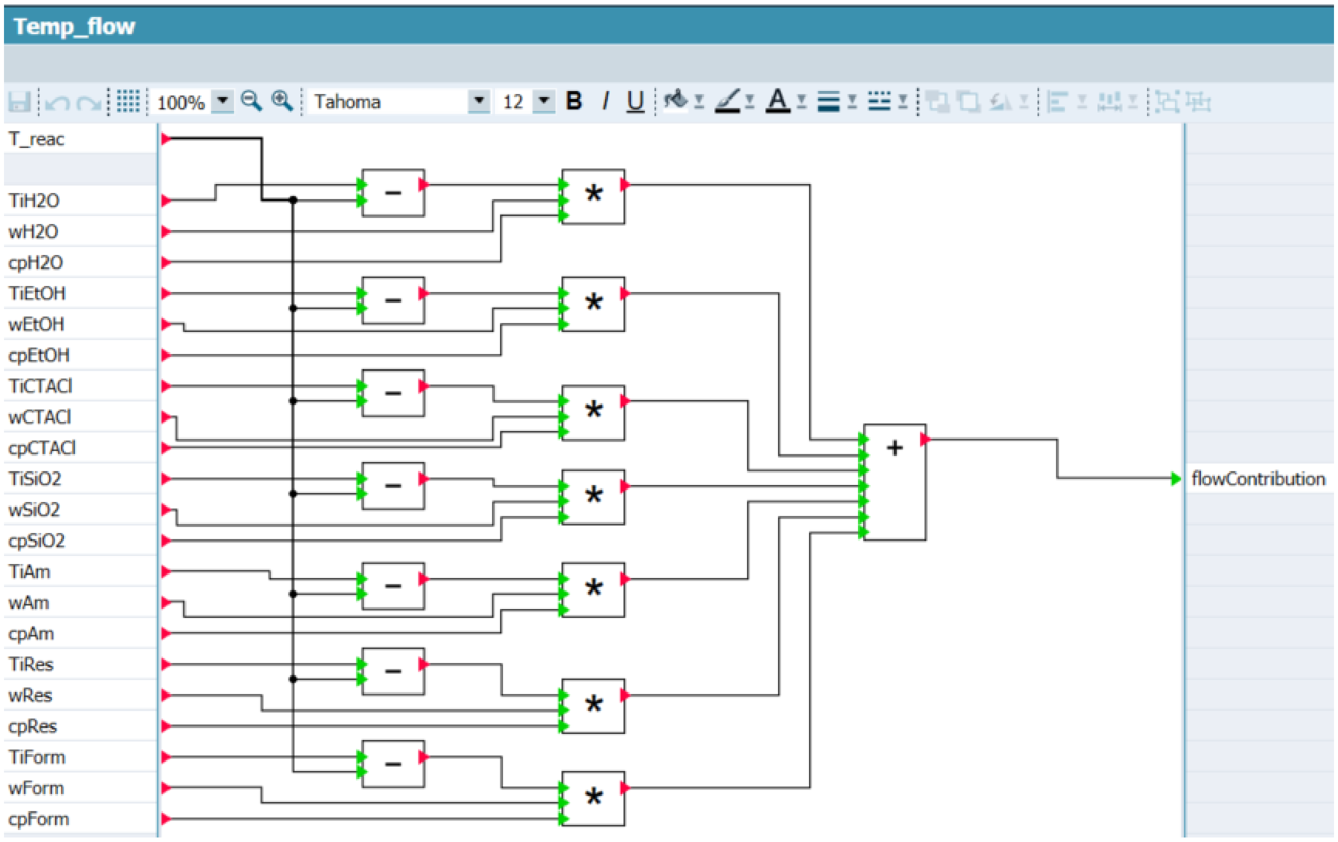


Figure 5.3.4: The macro editor with calculations for the inflow.

The macro *tempFlow* shown in Figure 5.3.4 handles the calculations that determine how the different compounds flowing into the reactor affect the temperature in the reactor. The two top blocks in the figure take the difference between the temperature of $H2O$ and T_reac, and multiplies the difference with the specific heat capacity and mass flow rate of $H2O$. All the blocks directly below do the same for the other compounds, and they are then added together, resulting in the final value containing the effect that inlet flows has on the temperature in the reactor.

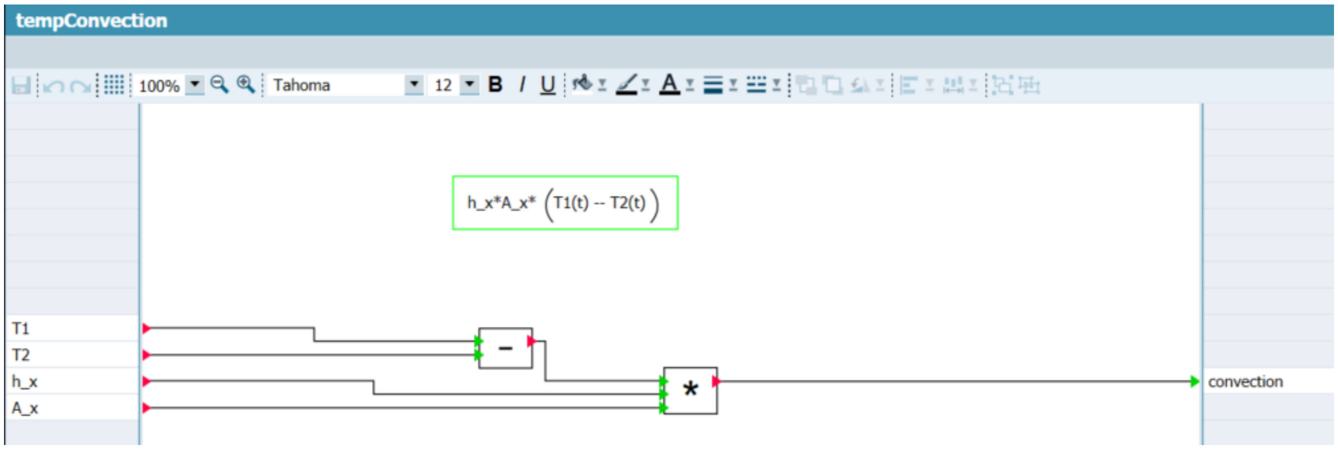


Figure 5.3.5: The macro editor with calculations for the element's heat convection.

The macro *tempConvection* shown in Figure 5.3.5 contains the calculation of convective heat transfer. The figure shows the calculations happening inside the macro, with the blocks doing the actual calculations from the macro's input to output. The output is the product of the heat transfer coefficient, the area, and the difference between two temperatures, where all these parameters are inputs to the macro as shown on the left side of the figure.

5.4 Concentration Dynamics

The concentration chart consists of two main parts. They are the same two parts separated by a minus sign in Equation A.4, the modeled rate of change in resorcinol concentration. Like the other charts, the concentration equation has been written in the chart, and the associated SIMIT blocks have been outlined to indicate their function in the equation.

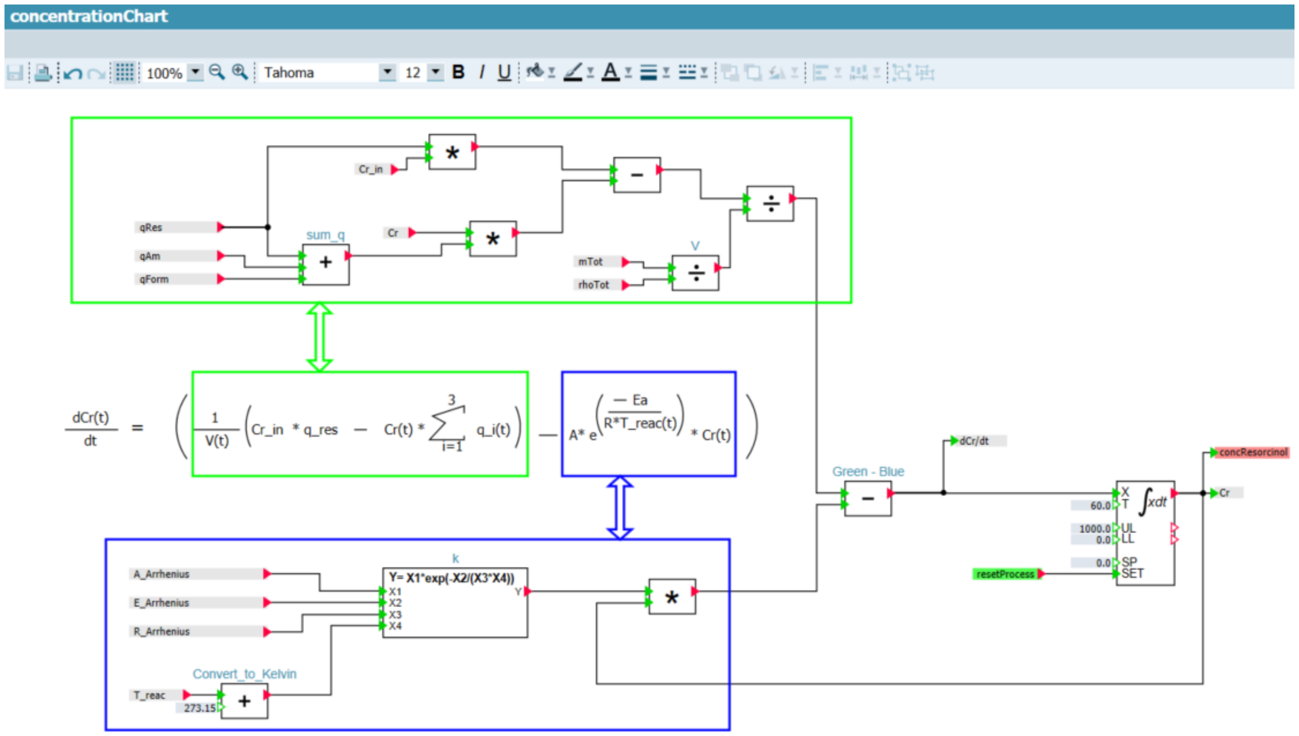


Figure 5.4.1: The modeled resorcinol concentration dynamics.

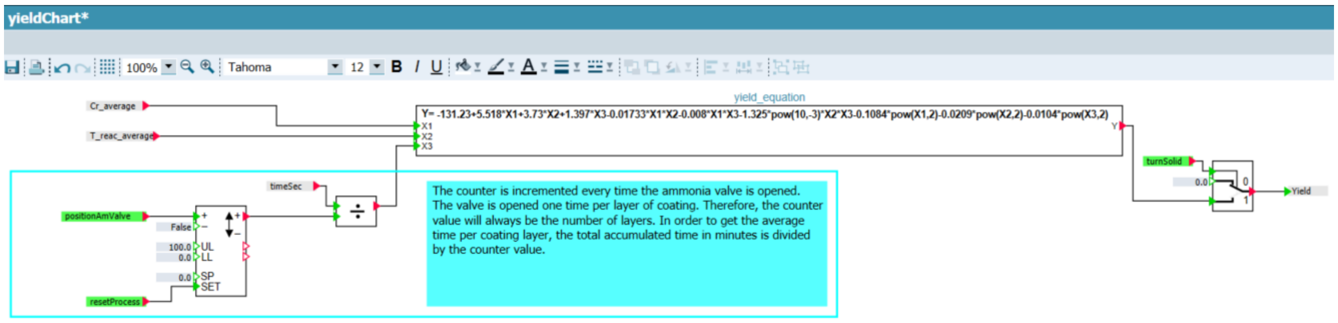
The concentration equation is written in the middle of the left side of the chart as shown in Figure 5.4.1. The first main part of the equation is outlined in green, and a green up/down arrow points to the associated SIMIT blocks, also outlined in green. Starting from the left, the volumetric flow rates of resorcinol, ammonia, and formaldehyde, $qRes$, qAm, and $qForm$, are added in the addition block called sum_q. The sum is multiplied by the feedback signal of the resorcinol concentration in the reactor, Cr. This product is passed into the second input of a subtraction block, meaning that it is subtracted from the first input, which is the product of $qRes$ and the concentration of resorcinol in the source tank, Cr_in. The division block named V calculates the volume of the content in the reactor based on total mass, $mTot$, and total density, $rhoTot$, which are calculated in two separate charts. The difference in the subtraction block is then divided by the calculated volume.

As for the second part of the equation, this is outlined in blue in Figure 5.4.1. The bottom blue outline contains the SIMIT blocks doing the actual calculations. The user-defined function block, k, includes the function for $k(t)$ defined in Equation 3.48. This block takes four inputs, three of them being $A\_Arrhenius$, $E\_Arrhenius$, and $R\_Arrhenius$, which have set values from another chart. The final input is temperature, T_reac, which is the reactor temperature being calculated in the temperature chart shown in Figure 5.3.1. The temperature is passed through the addition block $Convert\_to\_Kelvin$ that does what the name suggests to ensure that the units match, because the gas constant R is defined using Kelvin. The output of k is then multiplied by the resorcinol concentration in the reactor.

Finally, the value of the collective calculation inside the blue outline is subtracted from the calculation in the green outline in the block $Green - Blue$. The output of this block is the rate of change in resorcinol concentration, which is then integrated to get the current concentration of resorcinol in the reactor, and this signal is again used as a feedback signal for previous calculations. The red signal $concResorcinol$ is sent to the PLC for monitoring, which is needed to regulate the concentration.

5.5 Yield Calculation

The yield of the process is derived from a relation between the resorcinol concentration, temperature, and time during the coating step of the process. The yield is not a dynamic parameter, but rather it amounts to one final value depending on the values of the three variables during the coating step, meaning that the yield calculation is finished once the coating step is finished. As the concentration and temperature change with time, these parameters cannot be used directly, so the final values used in the equation need to be the average values during the coating phase.



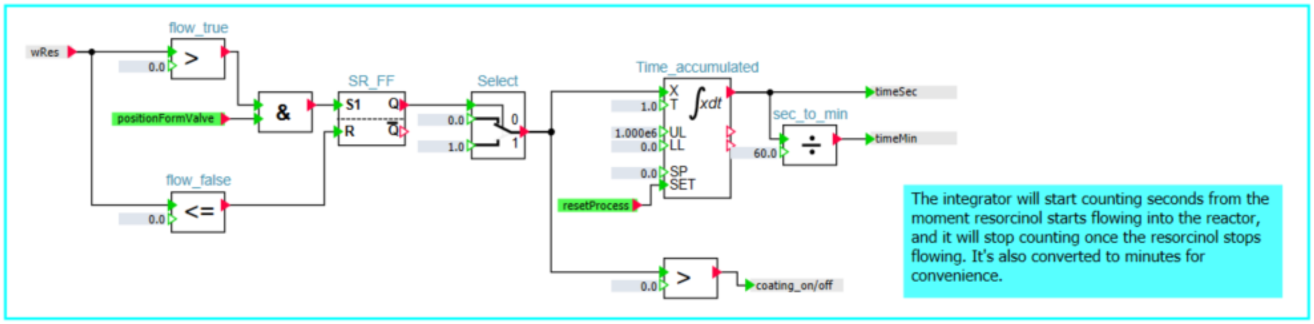
Y= -131.23+5.518*X1+3.73*X2+1.397*X3-0.01733*X1*X2-0.008*X1*X3-1.325*pow(10,-3)*X2*X3-0.1084*pow(X1,2)-0.0209*pow(X2,2)-0.0104*pow(X3,2)

The counter is incremented every time the ammonia valve is opened. The valve is opened one time per layer of coating. Therefore, the counter value will always be the number of layers. In order to get the average time per coating layer, the total accumulated time in minutes is divided by the counter value.

Figure 5.5.1: The first part of the yield chart.

The yield chart has been split into three separate figures that will be explained individually. The first part of the chart is shown in Figure 5.5.1, and it includes the main equation that defines the relationship between yield and the three parameters (concentration, temperature, and time) as defined in Equation

3.53. This is calculated in the block named *yield_equation*. All three of the variables passed into the inputs are calculated further down in the chart, shown in the following two figures. *Cr_average* and *T_reac_average* are connected directly to the input of *yield_equation*, but the time is divided by a number before the input. As explained in the cyan text box, this is done to ensure that the yield calculation holds up both for the single-layer coating as well as the multilayered coatings. The output of *yield_equation* is sent to the bottom input of a selection block. The output of this block is connected to the yield connector, and it will switch from 0 to the calculated yield when the signal *turnSolid* is set to TRUE by the PLC, which happens during the settling state of the process.



The integrator will start counting seconds from the moment resorcinol starts flowing into the reactor, and it will stop counting once the resorcinol stops flowing. It's also converted to minutes for convenience.

Figure 5.5.2: The second part of the yield chart.

The second part of the yield chart is shown in Figure 5.5.2, and this part handles the logic that measures the time elapsed during the coating step. There is a block in the SIMIT Standard library called *SimulationTime* that measures the elapsed time. However, it will measure the time of the simulation as a whole. This means that it cannot be initiated and stopped at specific times during the process. For that reason, the time measuring logic has been made from scratch. The most critical block in this figure is the selection block called *Select*, which sets the signal that starts and stops the time measurement. The blocks to the right of *Select* are the blocks that calculate the elapsed time, and a block that sets a signal used in the last part of the chart. Everything to the left of *Select* is the logic that specifies when the time should be counted, which is described in the next paragraph.

The connector on the far left side of the figure, *wRes*, is the flow rate of resorcinol. It is connected to two separate compare blocks named *flow_true* and *flow_false*. The block *flow_true* will output a TRUE signal while resorcinol is flowing. Otherwise, the output will be FALSE. The block *flow_false* will output the binary opposite of *flow_true*. The output of *flow_true* is passed into a binary AND operator, which is the block with the large ampersand enclosed in it. The second input of the AND block is connected to the signal called *positionFormValve*, a signal from the PLC used to open the formaldehyde valve. The AND block will only output a TRUE signal if both inputs are TRUE. Every other combination will result in a FALSE output.

The block after the AND block is a flip-flop called *SR_FF* in the chart. Once the input $S1$ receives a TRUE signal, this block will set the output TRUE until the input R receives a TRUE signal, resetting the output. The output will be set once the formaldehyde valve is opened while resorcinol is flowing into the reactor, and it will be reset after resorcinol has stopped flowing. The flip-flop's output is sent to the binary input of *Select*, which will decide which of the inputs with real numbers will be sent to *Select*'s output. The two inputs of *Select* with real numbers are set to 0 and 1, so the output passed to the integrator block, *Time_accumulated* is either 0 or 1. When the integrator receives a constant 1, it will integrate 1 with respect to time, yielding the elapsed time. *Time_accumulated* calculates the integral every second as specified by the input T = 1.0, meaning that the output will be the time in seconds. When the integrator receives a 0, the output will remain unchanged. The time is sent to a connector called *time_sec* and is divided by 60 to get the time in minutes, which is sent to the connector *time_min*. A compare block sets the connector *coating_on/off*. The block will then send a TRUE signal to the connector when the output of *Select* is above 0. As the name implies, the connector will contain a TRUE value when the simulation has entered the coating phase and a FALSE signal at all other times.

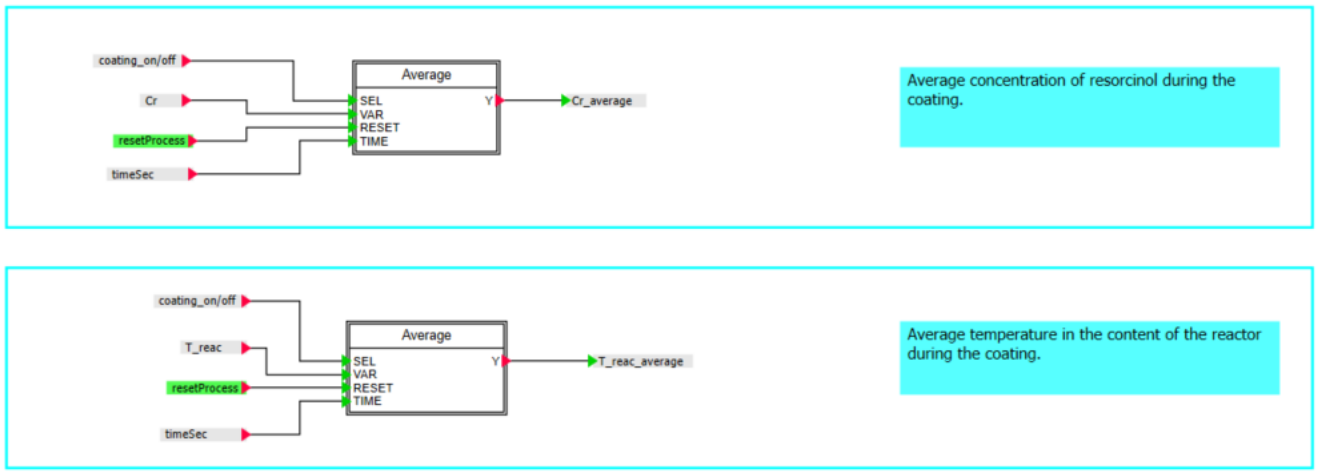


Figure 5.5.3: The third part of the yield chart.

Figure 5.5.3 shows the calculation of both the average concentration and the average temperature for the duration of the coating step. The top cyan rectangle contains the average concentration, while the bottom rectangle contains the temperature. They are both included in this figure because the method for finding the average is the same, and that is also why the method has been made in a macro that is used for both concentration and temperature. The macro, *Average*, has four inputs and one output. The purpose of the first input is to specify the duration for which the average value of the second input should be calculated. The third input will reset the calculation. This is done after the yield is calculated to be able to calculate the yield of the process for the next run-through. The signal *coating_on/off* is connected to the first input of both instances of the macro, meaning that the average concentration and temperature are only calculated during the coating step, as desired.

5.6 Supporting Charts

In order to simulate the behavior of the crucial parameters described in the previous sections, there is a need to know the behavior of some other variables affecting these parameters. The essential supporting variables are the total mass, the volume accumulated in the reactor, the combined density, the total specific heat capacity, and the solution's level in the reactor. Three of the mentioned supporting variables are calculated in three separate charts shown in the Figures 5.6.1 - 5.6.3.

Figure 5.6.1 shows the calculations of accumulated mass. The essence is the integrator blocks that take the flow rates as inputs. Notice how all the integrators' T inputs are set to 60. This is because the flow rates are defined as mass per minute. Consequently, the integral of the flow rate would be too high by a factor of 60 because the integrator block operates based on seconds. The following equation describes the math behind the integrator block, where the input T's functionality is apparent: $y = \frac{1}{T}\int xdt$.

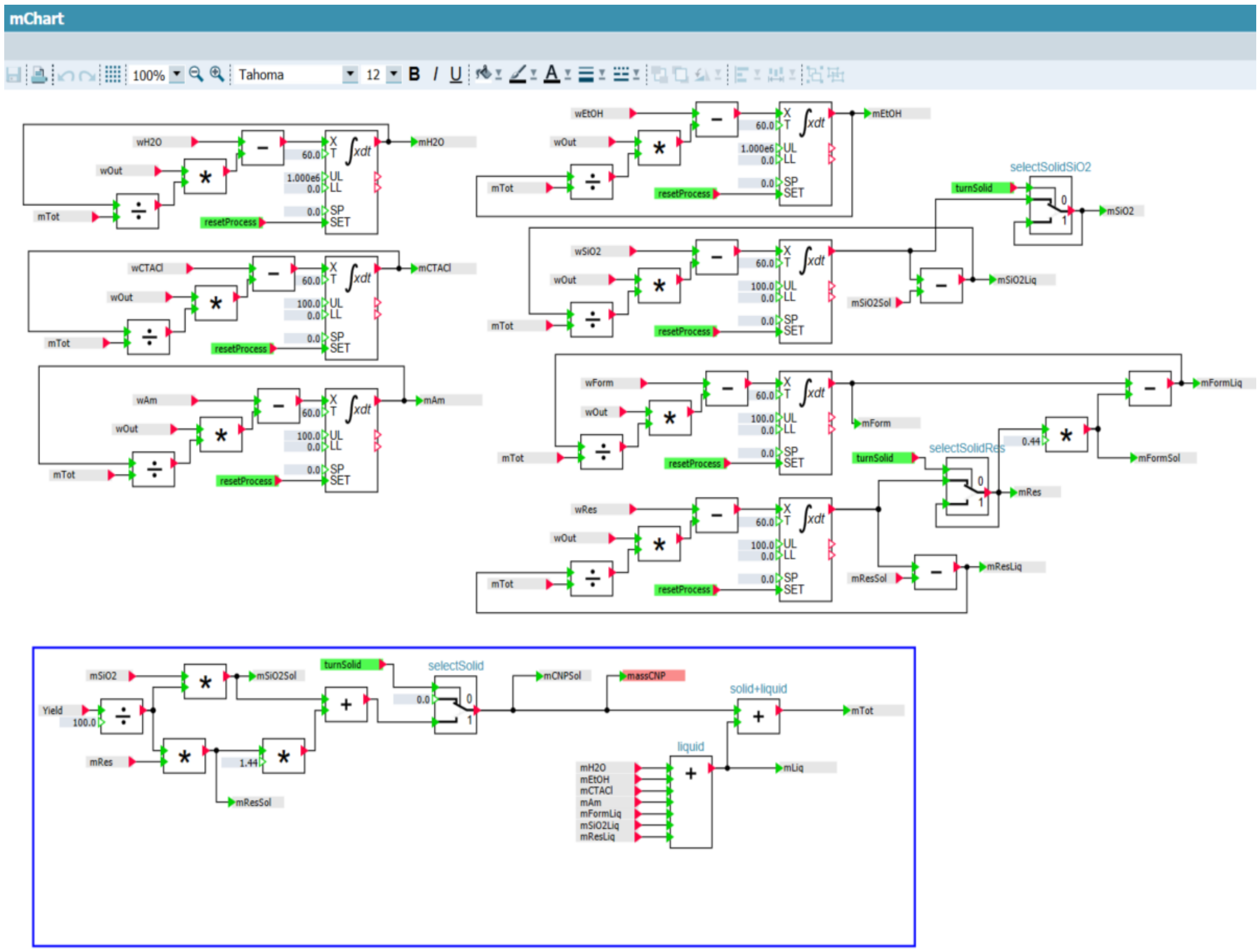


Figure 5.6.1: The accumulated total mass and mass of individual compounds.

The four integrator blocks corresponding to $mH2O$, $mCTACl$, mAm, and $mEtOH$ in the figure only have the change in mass as input, current mass as output, and the signal $resetProcess$ to reset the integrator's value. As defined in Equation 3.27, the change in mass is the difference between the inlet flow rate and the compound's weighted portion of the outlet flow rate. That is why the calculated mass of the compound is sent in a feedback loop where it is divided by the total mass and multiplied with the outlet flow rate. This value represents how much of the single compound is drained from the reactor, assuming perfect mixing. An example from the chart is the top left integrator in the figure. Here, the flow $wOut$ is multiplied with $mH2O$ divided by $mTot$, which is then subtracted from the inlet flow $wH2O$, yielding the change in H_2O mass that is finally integrated to get the current mass.

The other three integrators have the same functionality, but their outputs are also used to calculate the relation between liquid and solid mass for these individual compounds. This relation is based on the yield as well as the general relation shown in Equation 5.1. $mSiO2Liq$ is the amount of silicon nanoparticles that ends up uncoated in the liquid that is considered waste at the end of the process. It is set by the output of the subtraction block that takes the difference between the total silicon mass, $mSiO2$, and the solid silicon mass, $mSiO2Sol$. $mSiO2Liq$ is used in a feedback loop to account for the amount of silicon nanoparticles that flows out of the reactor.

The solid part is calculated inside the blue rectangle, and is described after the next paragraph. The selection block *selectSolidSiO*2 takes the output of the integrator block as input, which is the total silicon mass. Based on the PLC signal *turnSolid*, the input of *selectSolidSiO*2 connected through to the output will switch from input 0 to input 1. The output is fed into input 1, which effectively freezes the value received at input 0 when *turnSolid* turns active. The exact same logic is used for calculating the mass of resorcinol, but the formaldehyde is calculated in a slightly different way. This is because the amount of formaldehyde that turns solid is related to the resorcinol mass. As seen in the right side of the chart, *mRes* is multiplied by 0.44, which results in the solid mass of formaldehyde, *mFormSol*. This value is then subtracted from *mForm* to get the liquid portion of formaldehyde, *mFormLiq*, which is the value used in the feedback loop.

The main idea of the calculations performed by the blocks outlined in blue is to know how much mass has accumulated in the reactor in total, and also the amount of liquid mass compared to solid mass. This relation is based on the following equation:

$$m_{CNP} = m_{SiO2} + 1.44 \cdot m_{res} \tag{5.1}$$

where m_{CNP} is the mass of the coated nanoparticles at 100% yield, m_{SiO2} is the mass of the silicon nanoparticles, and m_{res} is the mass of resorcinol. The resorcinol mass is multiplied by 1.44 because it includes the mass of formaldehyde formed in the coating, which is equal to 44% of the resorcinol mass.

Starting from the left inside the blue outline, *mSiO*2 and *mRes* are each multiplied with the normalized value of *Yield* individually, resulting in *mSiO*2*Sol* and *mResSol*, respectively. *mResSol* is then multiplied by 1.44 and added with *mSiO*2*Sol*, which gives the value of *mCNPSol* as seen in Equation 5.1. However, *mCNPSol* will not receive this value before the settling step of the process because that is when *turnSolid* is set active by the PLC, which will switch the value of *selectSolid* from 0 to the calculated solid mass. The reason for this logic is to make the simulation model valid for the entirety of the process, because the calculated solid mass would actually slowly turn solid during this step of the physical process.

The addition block called *liquid* has seven inputs, which are the liquid compounds and the calculated liquid portions of the other compounds. The output, *mLiq*, is added together with the solid mass in the addition block *solid* + *liquid*, which results in the final *mTot* containing the varying total mass inside the reactor.

The chart made for calculating the specific heat capacity of the content in the reactor, shown in Figure 5.6.2, is based on the same principle as the mass chart, which is to separate the physical properties of the liquid and solid portion of the total reactor content. Shortly explained, it is based on the specific heat capacities of individual compounds and their weighting factor in the whole solution. The blocks inside the red outline are the specific heat capacities and the weighting factor for each compound known to stay liquid throughout the entire process, while the blocks inside the green outline are the specific heat capacities and the weighting factor for the solid compounds. The specific heat capacity of the liquid and the solid are added together and divided by the total mass to get *cpTot*, the specific heat capacity of the solution in its entirety.

The chart calculating the total density of the solution is almost an exact replica of the chart calculating specific heat capacity, which is why it will not be shown in this chapter. The interested reader can examine all charts in Appendix E

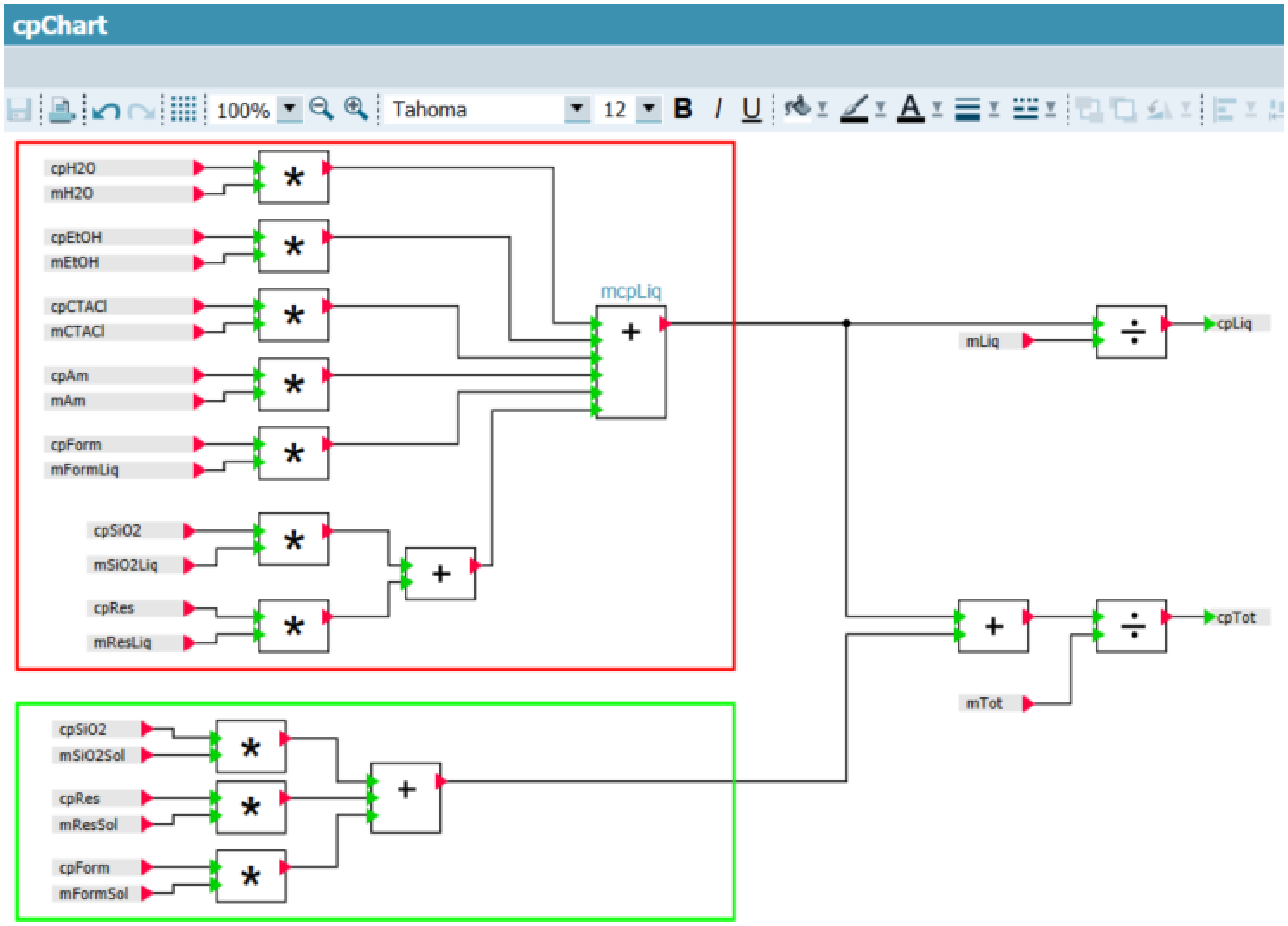


Figure 5.6.2: The calculated specific heat capacity of the total content in the reactor.

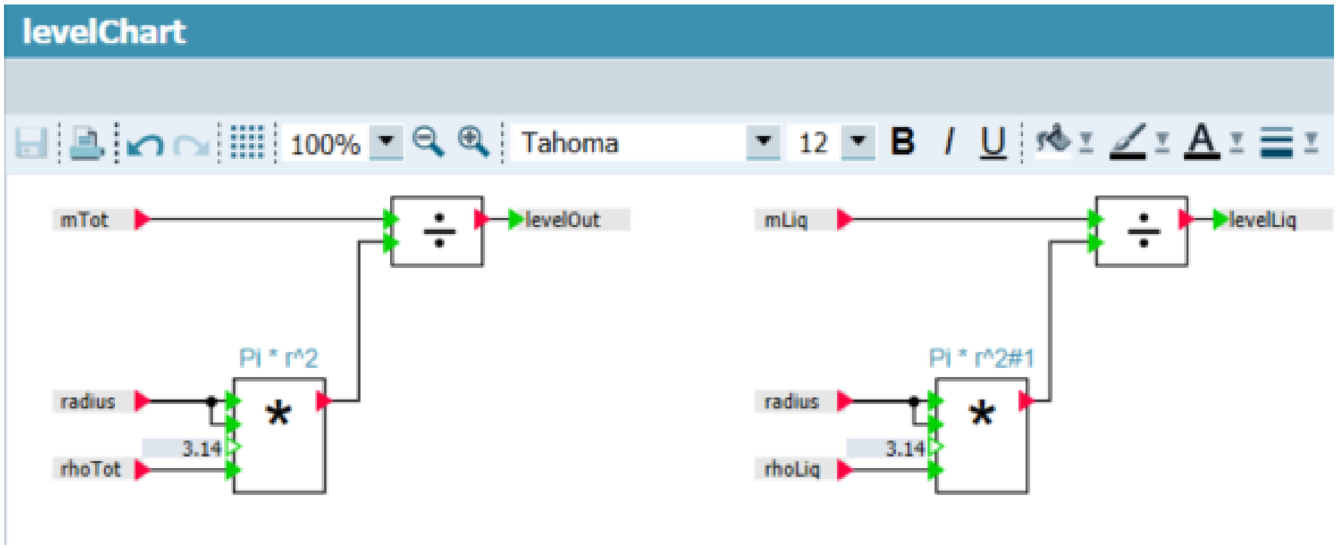


Figure 5.6.3: The calculated level of the solution in the reactor.

The solution's level in the reactor is based on the solution's volume relative to the dimensions of the reactor. Starting from the left of the chart shown in Figure 5.6.3, the total mass inside the reactor, $mTot$, is divided by the product calculated in $Pi * r\hat{}2$. This multiplication block calculates the base area of the reactor, and it also takes the total density, $rhoTot$, as an input. The level is the volume divided by the base area, which is why $mTot$ needs to be divided by $rhoTot$ in order to get the volume. The level is sent to a connector called $levelOut$. The exact same calculation is done again to the right of $levelOut$, with the variables $mTot$, $rhoTot$, and $levelOut$ switched out for the corresponding liquid variables. The result, $levelLiq$, is used to calculate the flow rate out of the reactor.

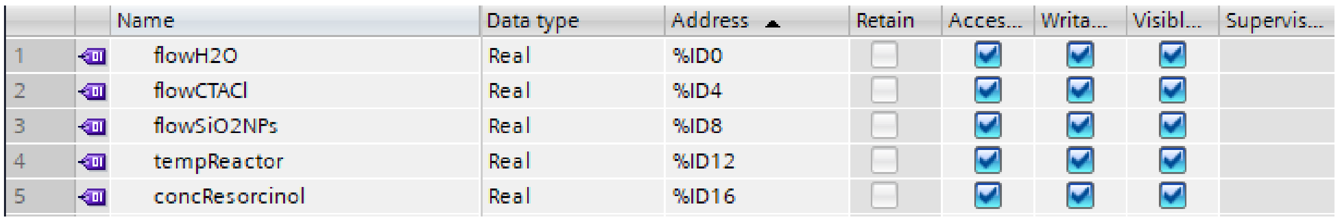
Chapter 6

Control System

In this chapter, the development of the PLC control system and the HMI (Human Machine Interface) screens are presented. In addition an explanation of the general programming style and programming methodology.

6.1 Programmable Logic Controller

The main intention for the PLC (Programmable Logic Controller) control system is to make the correct control action, at the correct time, in the correct phase of the process, based on different parameters collected, on the various inputs on the PLC, from the process. These parameters are generated, in this project, by the SIMIT model. The action is then performed to regulate or control some variable of interest in the process. In this project, these variables are mainly the reactor temperature, the different flow rates of compounds, and the resorcinol concentration. These inputs or outputs variables are called *Tags*, each tag has its corresponding variable name, which can be used in the program code, connected to an input or output on the PLC. A snippet of the tag table for the PLC used in this project can be seen in Figure 6.1.1. The conversion from analog to digital signals between the models, "Sensor" values, and the PLC is not taken account for in this project, but should this control system be implemented on the real process in the future is this something that needs to be taken into account.



		Name	Data type	Address ▲	Retain	Acces...	Writa...	Visibl...	Supervis...
1		flowH2O	Real	%ID0	☐	✓	✓	✓	
2		flowCTACl	Real	%ID4	☐	✓	✓	✓	
3		flowSiO2NPs	Real	%ID8	☐	✓	✓	✓	
4		tempReactor	Real	%ID12	☐	✓	✓	✓	
5		concResorcinol	Real	%ID16	☐	✓	✓	✓	

Figure 6.1.1: Overview of the PLCs tags

6.1.1 Programming Style

In the TIA (Totally integrated Automation) Portal's PLC development environment STEP7, there are a number of different ways to program a PLC system. The three considered ways are: SCL (Structured Control Language), Ladder, and Graph. Each of the three methods has a recommended task according to recommendation DA002 in [17]. SCL should be used in the standard blocks due to its compact form and good readability. Ladder should be used for calling environments, as this allows for a faster overview and is easier to diagnose. The last one, Graph, is preferred when having sequential steps as it is easy to develop the code and it can be followed easily. In this project, in accordance with the recommendations, it was decided to use SCL to develop all the standard blocks like the different FBs (function blocks), and the call environment was developed within an organization block called *Main* in Ladder code. It was decided to use SCL for the sequential steps due to the fact that it was relatively easy to develop a switch case programming structure that made it easy to get an overview of the input and output of the FBs in the different steps.

One of the goals of the PLC system was that it should be easily maintainable and understandable. Therefore, it was decided that it should be developed based on Siemens Programming style guide for SIMATIC S7-1200/1500 [17] and Programming Guideline for S7-1200/1500 [16]. The style guide [17] states the following in its goals section: “The rules and recommendations described in the following chapters [are supposed to/ shall] help you create a uniform program code which is maintainable and reusable.” Hence, by following both guidelines, the goal for the PLC system will be accomplished. In this project, the main focus was on functionality, performance, and clarity when the code was developed. The rules and recommendations mentioned below have been chosen due to this focus.

Rules and recommendations complied with from [17]:

- **Nomenclature and Formatting:**
  - NF001 Rule: Unique and consistent English identifiers
  - NF002 Rule: Use meaningful comments and properties
  - NF004 Rule: Comply with prefixes and structures for libraries
  - NF005 Rule: Use PascalCasing for objects.
  - NF006 Rule: Use camelCasing for code elements
  - NF007 Rule: Use prefixes
  - NF008 Rule: Write identifiers of constants in CAPITALS
  - NF009 Rule: Limit the character set for identifiers
  - NF014 Rule: Format SCL code meaningfully
- **Design guidelines/architecture:**
  - DA001 Rule: Structure and group a project/library
  - DA002 Recommendation: Use appropriate programming language
  - DA006 Rule: Access static variables from within the block only
  - DA008 Rule: Write output variables only once
  - DA009 Rule: Keep used code only
  - DA011 Rule: Continuous asynchronous execution with “enable”
- **Performance**
  - PE006 Recommendation: Prefer temporary variables
  - PE007 Recommendation: Declare important test variables as static
  - PE008 Recommendation: Declare control/index variables as “DINT”
  - PE011 Recommendation: Simplify IF/ELSE instructions
  - PE012 Recommendation: Sort IF/ELSIF branches according to expectation
  - PE013 Recommendation: Avoid memory intense instructions

6.1.2 Structure

The PLC program is divided into the different folders *MiscDBs*, *System Control*, *Preparation of Solution*, *Coating of Particles*, *Washing and Waste Removal*, and *Supporting Functions*. This is to make the overview of the structure of the PLC control in the TIA Portal tidy. Each folder contains FBs executing specific tasks in that phase and/or DBs (data blocks) that store the values of the different parameters.

The folder *MiscDBs* contains the data blocks that save miscellaneous values such as the values for the HMI/PLC interface, the mass/volume calculations, and the HMI coating time conversion. The other DBs are placed in the folders with their corresponding FBs. The function of the FBs in the folders *System Control*, *Supporting Functions*, *Preparation of Solution*, *Washing and Waste Removal*, and *Coating of Particles* will be described in the following five subsections.

6.1.3 System Control

The folder *SystemControl* contains four components, one FB, *ProcessSequence*, responsible for the overall PLC system structure, its DB, and two OBs (Organization Blocks) called *Main*, and *Cyclic interrupt* (*PID*).

"Main":
The OB, *Main*, is an organization block that is the interface between the operating system and the user program. It initializes the entire program and processes the program cyclically. The program runs the *ProcessSequence* FB.

"Cyclic interrupt (PID)":
In this OB, all of the PIDs are situated. There are three PIDs, one for resorcinol concentration regulation and two for temperature regulation. The reason they need their own OB is due to the way the different OB blocks are compiled. The cyclic interrupt has a set execution time for every compilation. The other OBs, as *Main*, does not execute at a set rate, it depends on the severity of the calculations made, some take longer time than others. The execution time is important for the PID as it calculates the correct control action based on the input value. Meaning if the PIDs were present in *Main*, and did not have its own Cyclic OB, the calculated control action would probably be wrong according to the input due to delays or would miss out on changes in the process to react on, due to slow sampling from the process. The PID for the regulation of resorcinol concentration can be seen in Figure 6.1.2. Here the tags, "*startConcReg*", "*concResorcinol*", *and*"*positionResValve*" can be seen. Their corresponding I/O address is shown above the variable name, where *M* i an internal address, *ID* is a digital input, and *QD* is a digital output. The setpoint value selected on the HMI, collected from the PLC-HMI interface DB, "*PLC/HMIInterface*".*setConcPID*, is also shown.

The PID collect the value of the resorcinol concentration and the PID calculates the correct control action based on the PID parameters defined in the block, to get the resorcinol value to the setpoint value defined on the HMI. The *startConcReg* signal closes the NO (normally open) switch when the bit goes 1, to enable the regulation of the resorcinol concentration. How the PIDs work, and how the PIDs parameters were decided is described in detail in Chapter 7.

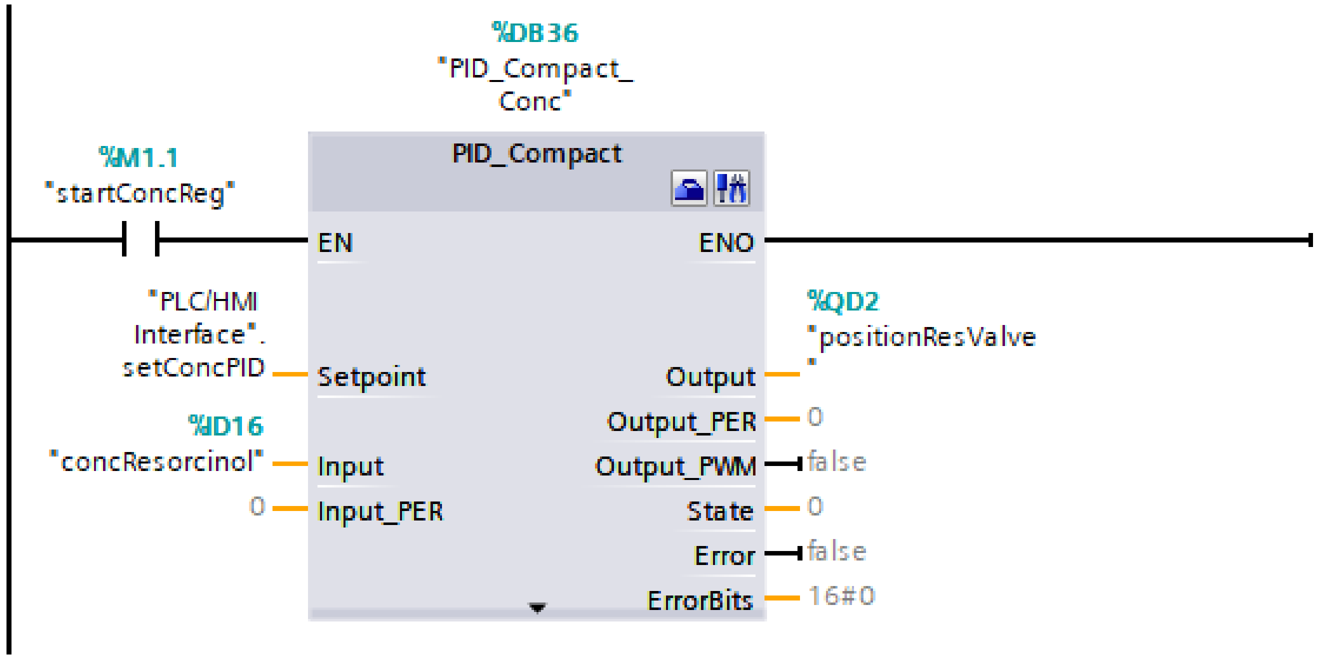
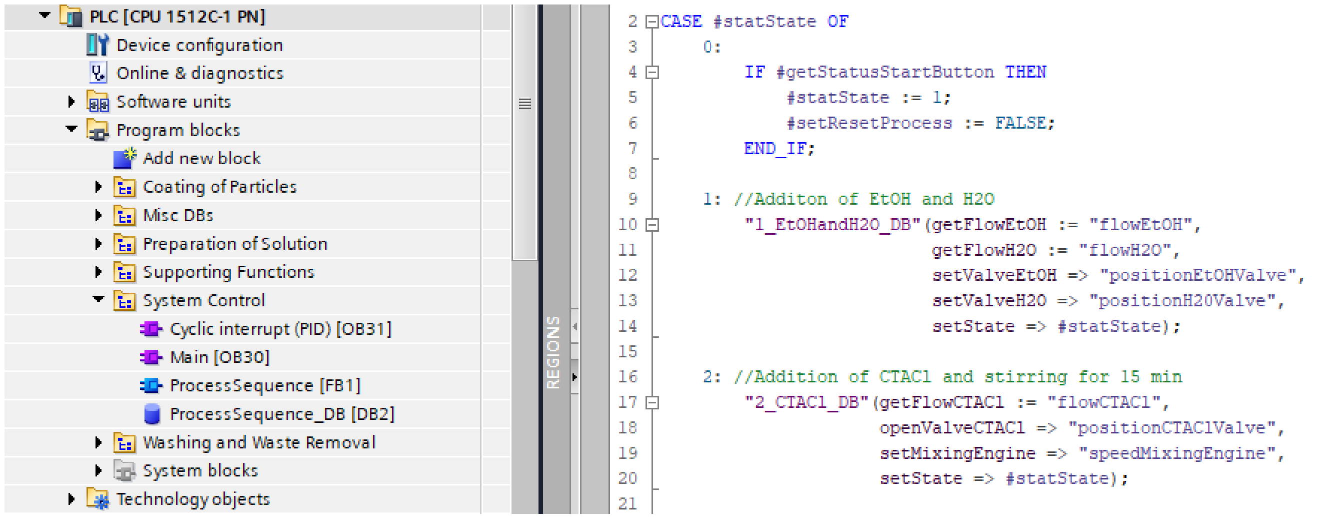


Figure 6.1.2: PID in the Cyclic interrupt OB

"ProcessSequence":
The process follows the sequence that was decided by the step diagram seen in Figure 2.3.1, this was to make the PLC code organized. This functionality is handled in an overall FB called *ProcessSequence*. This FB has a variable called *statState* that decides which step that is executed. This is done by using a switch case, where the corresponding FB to each step from 0-12 is executed. To start the process, a start button on the HMI screen is pushed. This button changes the input *getStatusStartButton* on the *ProcessSequence* FB from *FALSE* to *TRUE*, thus changing the variable *statState* from 0 to 1, initializing step 1, called 1_*EtOHandH2O*, and setting *setResetProcess* to *FALSE*, as shown in Figure 6.1.3 below. The figure also shows how the FBs in steps 1 and 2 are initialized with their corresponding inputs and outputs, in addition to the folder structure that can be seen on the far left side of the figure.



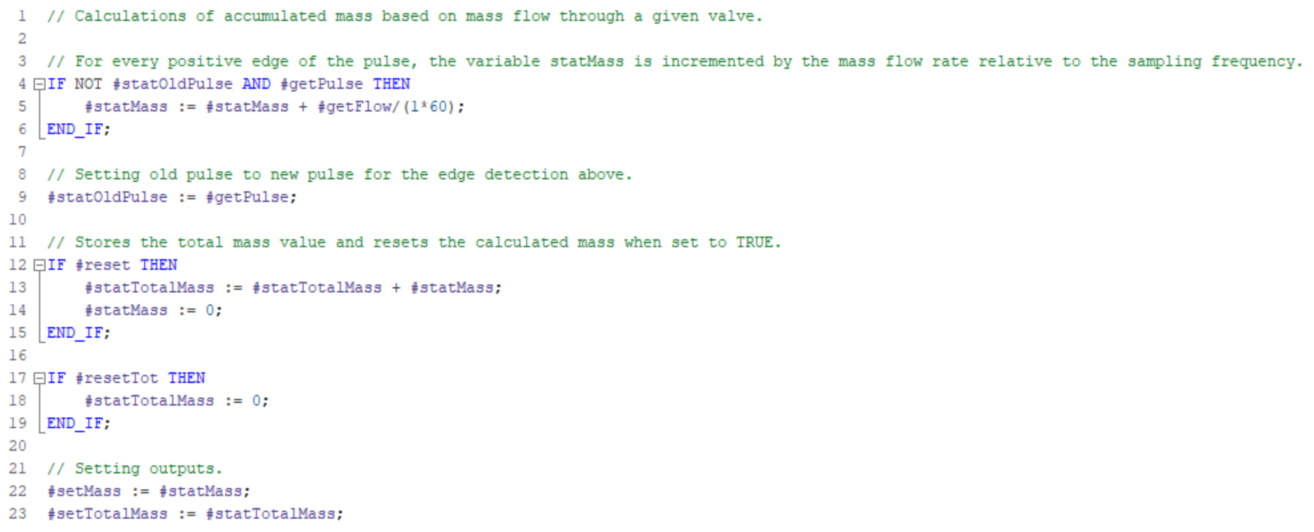
```
CASE #statState OF
    0:
        IF #getStatusStartButton THEN
            #statState := 1;
            #setResetProcess := FALSE;
        END_IF;

    1: //Additon of EtOH and H2O
        "1_EtOHandH2O_DB"(getFlowEtOH := "flowEtOH",
                          getFlowH2O := "flowH2O",
                          setValveEtOH => "positionEtOHValve",
                          setValveH2O => "positionH2OValve",
                          setState => #statState);

    2: //Addition of CTAC1 and stirring for 15 min
        "2_CTACl_DB"(getFlowCTACl := "flowCTACl",
                     openValveCTACl => "positionCTAClValve",
                     setMixingEngine => "speedMixingEngine",
                     setState => #statState);

```

Figure 6.1.3: The structure of the PLC program and the first three steps in the *ProcessSequence* FB.

6.1.4 Supporting Functions

The folder *Supporting Functions* contains two FBs, *MassCalculation*, responsible for calculating the mass added to the reactor, and *CoatingTime*, responsible for converting the coating time values from the HMI.

"MassCalculation":
The first FB in this folder is the *MassCalculation* FB, its task is to calculate the mass of a given compound that has accumulated in the reactor. This value is used as a feedback signal that decides when to close the different valves. For example, should only 30 kg of *CTACl* be added in step 2 (2_*CTACl*). Therefore, the accumulated mass needs to be known to close the valve at the correct time. In cases where volume is desired instead of mass, this is corrected for in the given FBs where *MassCalculation* is executed.



```
 1  // Calculations of accumulated mass based on mass flow through a given valve.
 2
 3  // For every positive edge of the pulse, the variable statMass is incremented by the mass flow rate relative to the sampling frequency.
 4  IF NOT #statOldPulse AND #getPulse THEN
 5      #statMass := #statMass + #getFlow/(1*60);
 6  END_IF;
 7
 8  // Setting old pulse to new pulse for the edge detection above.
 9  #statOldPulse := #getPulse;
10
11  // Stores the total mass value and resets the calculated mass when set to TRUE.
12  IF #reset THEN
13      #statTotalMass := #statTotalMass + #statMass;
14      #statMass := 0;
15  END_IF;
16
17  IF #resetTot THEN
18      #statTotalMass := 0;
19  END_IF;
20
21  // Setting outputs.
22  #setMass := #statMass;
23  #setTotalMass := #statTotalMass;
```

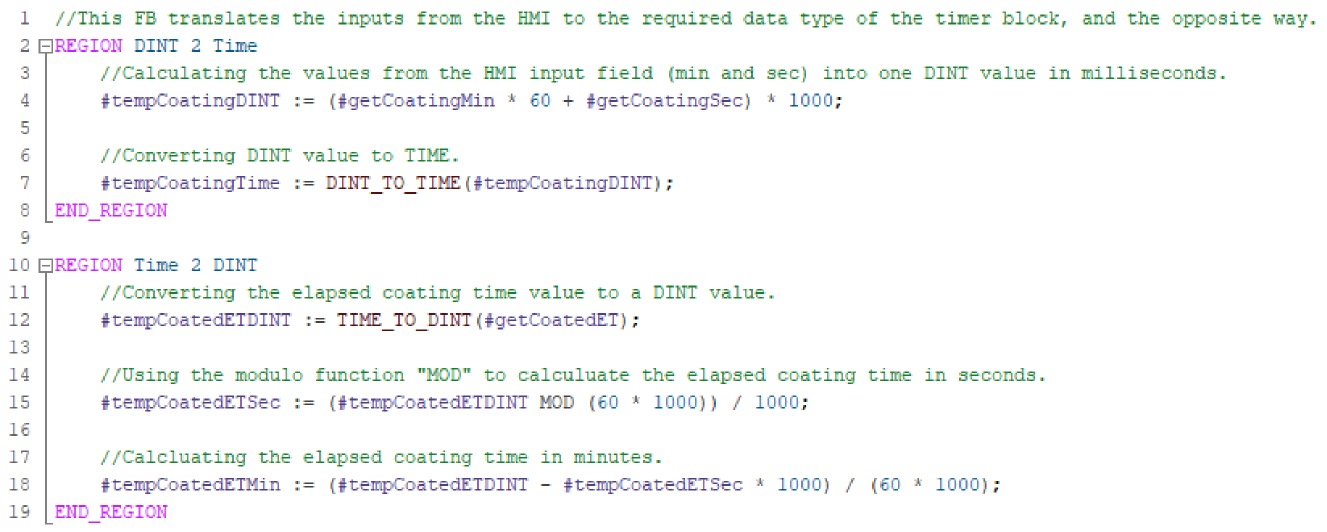
Figure 6.1.4: Overview of the *MassCalculation* FB

The first IF statement in Figure 6.1.4 declares when it is desired to add the input *getFlow* to the static variable *statMass* at a given sampling rate defined by the FBs input *getPulse*. It only adds the flow when the *getPulse* has a positive edge. The sampling rate is set to 1 Hz, equaling one pulse per second. Since the value, *getFlow*, is defined as kg/min, it is divided by 60 multiplied with 1 to covert it to flow per second per pulse. The value *getPulse* is then made equal to the static variable *statOldPulse* as a part of the rising edge detection. Since some compounds are added in multiple steps, a functionality to reset the mass value was created. The second IF statement saves the accumulated mass, *statMass*, as *statTotalMass*, and sets *statMass* thereafter to 0. This happens when the input *reset* is set high. *statTotalMass* is the value of total mass of a given compound added to the reactor during the entire process, not only in an individual step. The final IF statement sets the total mass value to 0 when the input, *resetTot*, is set high. Beneath line 22 in the figure, the static variable *statMass* is set on the output *setMass* which outputs the calculated mass to the given FB that *MassCalculation* has been called in. Also the variable, *statTotalMass*, is saved in the DB to the given compound.

"CoatingTime":
The other FB in this folder is the *CoatingTime* FB. The FB has two functionalities. Converting the inputed coating time from the two input field on the HMI, to one Time data type variable, and the opposite. Converting the elapsed coating time from the timer with data type Time, into two variables describing the elapsed coating time as minutes and seconds with a DINT data type, to be shown on two digital displays on the HMI. The FB is separated into two regions each representing their corresponding

conversion, named *DINT* 2 *Time* and *Time* 2 *DINT*, respectively. The conversion from DINT to Time is needed due to the fact that the coating time is used in a timer block to control the length of the coating phase in step 6_*Form*. The timer block has an input, PT, that decides the timer length, and this input needs a variable with a Time data type, and the input from the HMI has a DINT variable. Consequently, this FB has been developed to alleviate this issue. The same issue appears the opposite direction as well. The digital display can not display a Time data type value. Therefore, the elapsed time from the timer block, needs to be converted to DINT and splited into the equivalent time in minutes and seconds.

The *CoatingTime* FB has three inputs, one that contains the value for coating time in minutes, named *getCoatingMin*, one that contains the value for coating time in seconds, named *getCoatingSec*, and one that contains the elapsed coating time, named *getCoatedET*. Firstly, the conversion from DINT to Time will be explained. A DINT value of one represents one millisecond, therefore, the expression on line 4 in Figure 6.1.5 is needed. In this expression, the input *getCoatingMin* is multiplied by 60 to get the value in seconds, and then it is added with the input *getCoatingSec* to get the total time value defined in seconds. This value is multiplied by 1000 to get the required DINT value in milliseconds called *tempCoatingDINT*. On line 7, the function *DINT_TO_TIME*(...), is used to convert the *tempCoatingDINT* to a time data type value called *tempCoatingTime*. The time value *tempCoatingTime* is then set on the output *setCoatingTime*.



```
 1  //This FB translates the inputs from the HMI to the required data type of the timer block, and the opposite way.
 2  REGION DINT 2 Time
 3      //Calculating the values from the HMI input field (min and sec) into one DINT value in milliseconds.
 4      #tempCoatingDINT := (#getCoatingMin * 60 + #getCoatingSec) * 1000;
 5  
 6      //Converting DINT value to TIME.
 7      #tempCoatingTime := DINT_TO_TIME(#tempCoatingDINT);
 8  END_REGION
 9  
10  REGION Time 2 DINT
11      //Converting the elapsed coating time value to a DINT value.
12      #tempCoatedETDINT := TIME_TO_DINT(#getCoatedET);
13  
14      //Using the modulo function "MOD" to calculuate the elapsed coating time in seconds.
15      #tempCoatedETSec := (#tempCoatedETDINT MOD (60 * 1000)) / 1000;
16  
17      //Calcluating the elapsed coating time in minutes.
18      #tempCoatedETMin := (#tempCoatedETDINT - #tempCoatedETSec * 1000) / (60 * 1000);
19  END_REGION
```

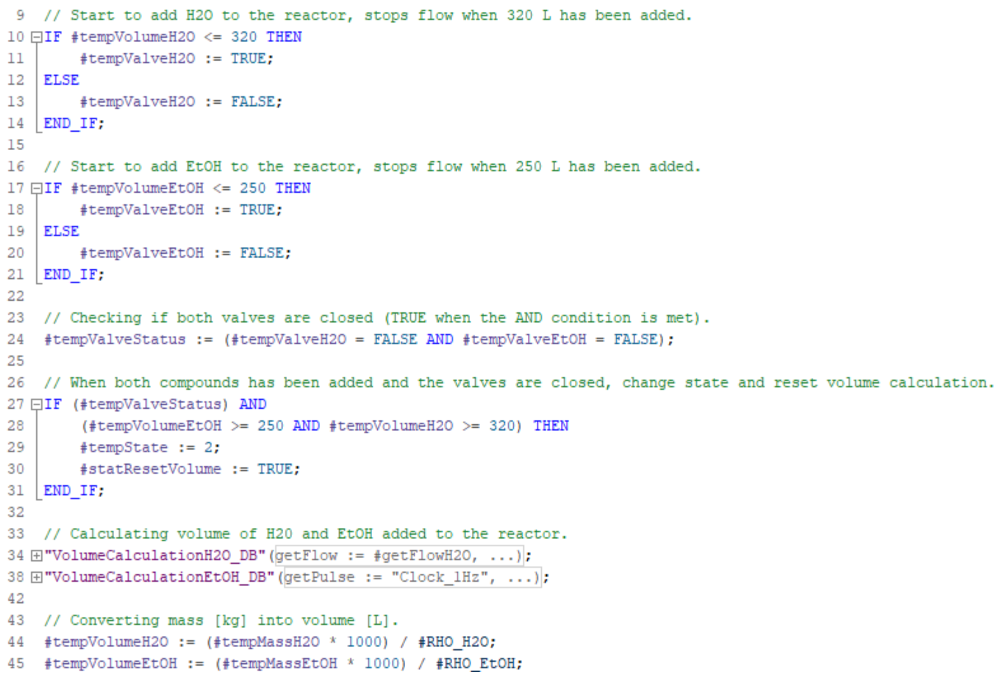
Figure 6.1.5: Overview of the *CoatingTime* FB

The conversion the other direction, apparent in region *Time* 2 *DINT*, works very similarly. Firstly, the *TIME_TO_DINT* function is used to convert the elapsed coating time to a DINT data type variable, called *tempCoatedET*. This one variable now needs to be split into to, one representing the time in minutes, and the other in second. The sum of them should be the same value as *tempCoatedET*. The time, with data type DINT in second, is calculated in line 15. This is done by completing the modulo of the variable *tempCoatedET* and 60 multiplied with 1000. The modulo gives the remainder of this division, where 60 multiplied with 1000 represents minutes. Meaning the remainder is the DINT values in milliseconds. Therefore, it is divided by 1000 to get the value in seconds. This value is saved as *tempCoatedETSec*. On line 18, the DINT time representation in minutes is calculated. This is done by taking the original DINT value for the time, *tempCoatedET*, minus the seconds value, *tempCoatedETSec*, resulting in the DINT value for minutes in millisecond. This value is therefore divided with 60000 to get the value in minutes. The end result is a DINT integer value of the time representation in minutes, named *tempCoatedETMin*. Both *tempCoatedETMin*, and *tempCoatedETSec* are set to their corresponding outputs at the end of the script.

6.1.5 Preparation of Solution

This section contains the four steps that together represents the preparation of the solution in this process. The four steps in this stage are: 1_$EtOHandH2O$, 2_$CTACl$, 3_$Silicon$, and 4_$Heating70$.

"1_EtOHandH2O":
The entire process is started with this step. The goal of this step is to add 320 L of H_2O and 250 L of EtOH. This FB controls this sequence. A snippet of the script is shown in Figure 6.1.6. The two first IF statements, between line 10 and 21, decides if the valves should be opened or closed. If $tempVolumeH2O$ is lower than or equal to 320 L, the valve should open, and when $tempVolumeH2O$ becomes greater than 320 L the IF statement goes $FALSE$ and consequently, $tempValveH2O$. The functionality is exactly the same for EtOH. Line 23 checks if both of the valves are closed or not, if both are closed $tempValveStatus$ goes $TRUE$ to indicate this. This signal is then used, in combination with checking if the volumes are over their desired value, to change to the next step, by setting $tempState$ to 2, and resetting the calculated masses, by setting $statResetVolume$ to $TRUE$. The combination of both signals are used to ensure that the valves are closed before switching the state, a fault in the physical components could leave the valves open. If this happens it is better that the system does not continue the process, as this could possibly destroy the batch. In line 34 and 38 the $MassCalculation$ FB is executed, and converted to volume in line 44 and 46. The outputs, $setValveEtOH$, $setValveH2O$, and $setState$, are set in the last three lines of the FB. The execution of $MassCalculation$ and setting of the output will only be showed in Figure 6.1.6 as this is done equally for each FB. The interested reader can see the entirety of each FB in Appendix D.

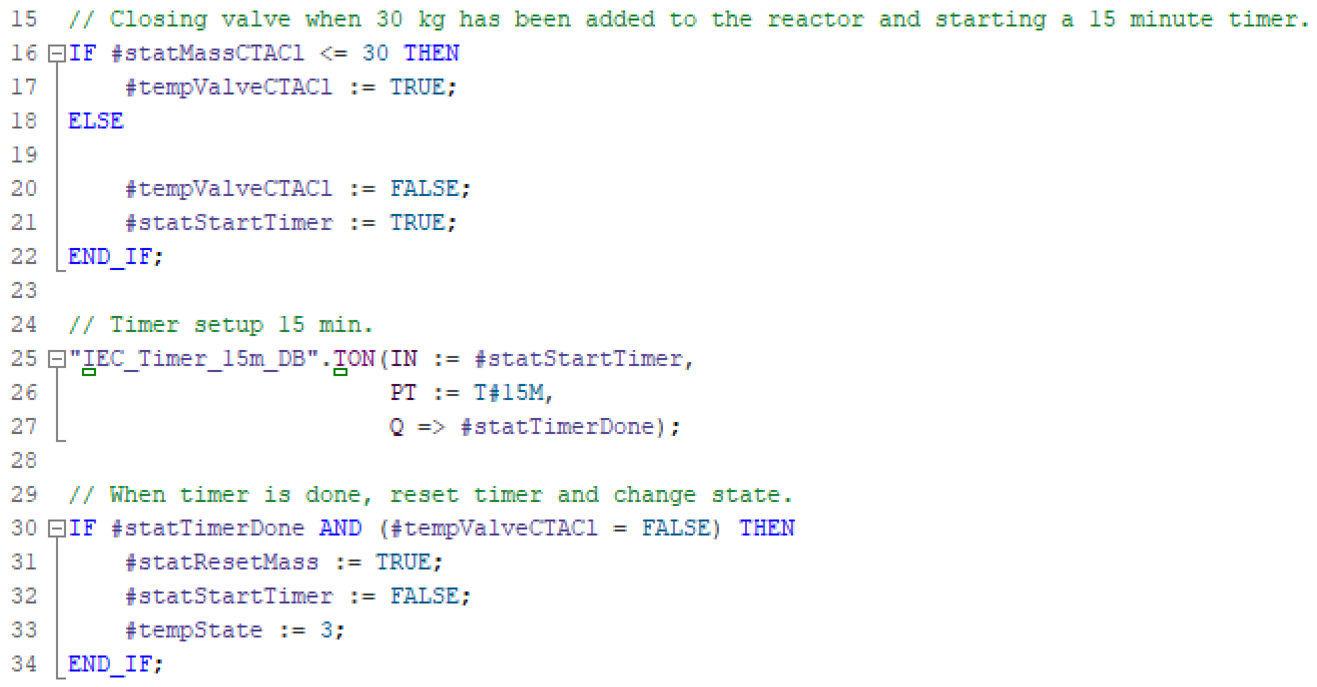


```
 9   // Start to add H2O to the reactor, stops flow when 320 L has been added.
10   IF #tempVolumeH2O <= 320 THEN
11       #tempValveH2O := TRUE;
12   ELSE
13       #tempValveH2O := FALSE;
14   END_IF;
15
16   // Start to add EtOH to the reactor, stops flow when 250 L has been added.
17   IF #tempVolumeEtOH <= 250 THEN
18       #tempValveEtOH := TRUE;
19   ELSE
20       #tempValveEtOH := FALSE;
21   END_IF;
22
23   // Checking if both valves are closed (TRUE when the AND condition is met).
24   #tempValveStatus := (#tempValveH2O = FALSE AND #tempValveEtOH = FALSE);
25
26   // When both compounds has been added and the valves are closed, change state and reset volume calculation.
27   IF (#tempValveStatus) AND
28       (#tempVolumeEtOH >= 250 AND #tempVolumeH2O >= 320) THEN
29       #tempState := 2;
30       #statResetVolume := TRUE;
31   END_IF;
32
33   // Calculating volume of H20 and EtOH added to the reactor.
34   "VolumeCalculationH2O_DB"(getFlow := #getFlowH2O, ...);
38   "VolumeCalculationEtOH_DB"(getPulse := "Clock_1Hz", ...);
42
43   // Converting mass [kg] into volume [L].
44   #tempVolumeH2O := (#tempMassH2O * 1000) / #RHO_H2O;
45   #tempVolumeEtOH := (#tempMassEtOH * 1000) / #RHO_EtOH;
```

Figure 6.1.6: Overview of the 1_$EtOHandH2O$ FB

2_CTACl:
This FBs goal is to add the correct amount of $CTACl$ and wait 15 minutes until the next step is initiated. Figure 6.1.7, shows a snippet of the script for this FB. The first IF statement decides if the valve should be opened or not. This is based on the value of $statMassCTACl$, calculated from the $MassCalculationCTACl\_DB$ instance in this FB. If $statMassCTACl$ is equal to or lower than 30 kg, will $tempValveCTACl$ be set to $TRUE$, but if $statMassCTACl$ is greater than 30 kg, will $tempValveCTACl$ be set to $FALSE$, closing the valve, and $statStartTimer$ will be set to $TRUE$, starting the 15 minute

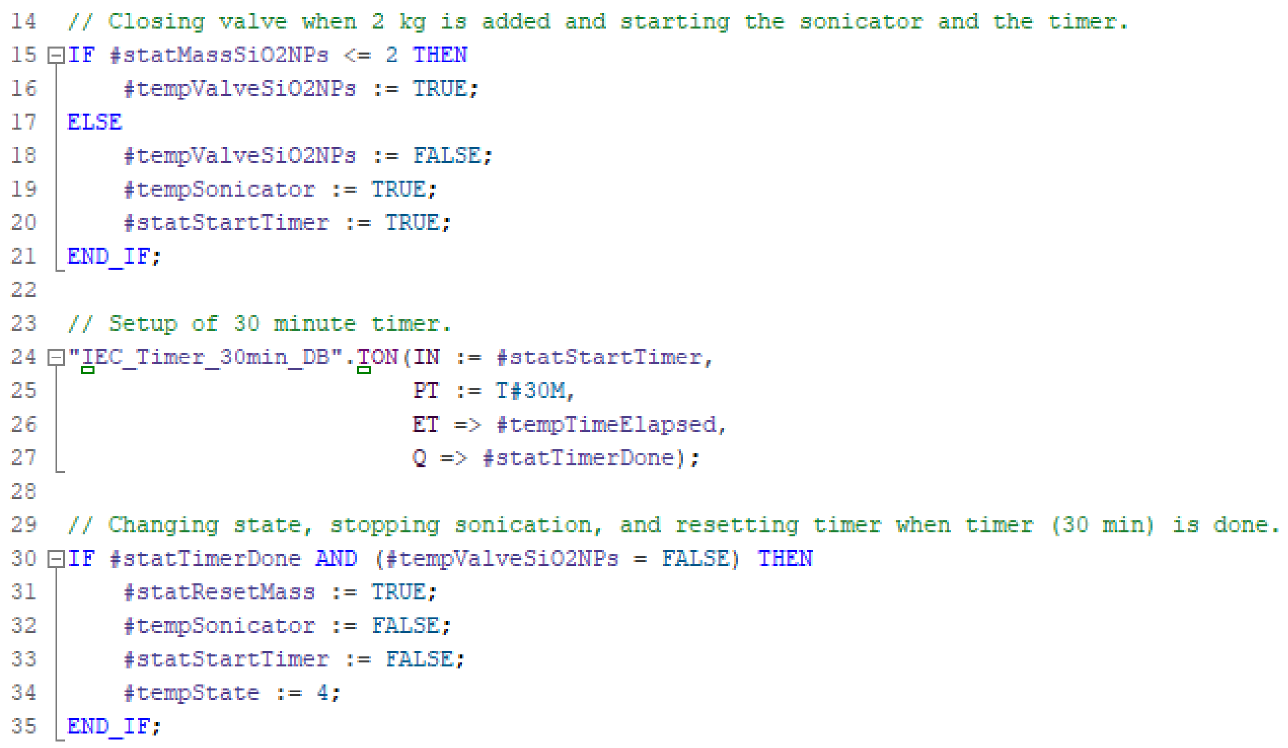
timer. When the IN input on the $IEC\_Timer\_15m\_DB$ goes high it will count the time specified on the input PT, 15 minutes in this case. $statStartTimer$, which is placed on the input IN, needs to be $TRUE$ during the entire countdown or else the timer will stop. When the time is up, the output Q goes $TRUE$, setting $statTimerDone$ $TRUE$. The last IF statement goes $TRUE$ when $statTimerDone$ goes $TRUE$ and $tempValveCTACl$ is $FALSE$, the position of the valve is included for the reason explained in the section about the FB, 1_$EtOHandH2O$. When this happens, the mass and the timer is reset, by setting $statResetMass$ $TRUE$, and $statStartTimer$ $FALSE$, and $tempState$ is set to 3. The last three lines set the $tempValveCTACl$ and the $tempState$ values decided by the script to the outputs. The output $setMixingEngine$ is always set $TRUE$ in this state, independently on the decisions made in this step.



```
// Closing valve when 30 kg has been added to the reactor and starting a 15 minute timer.
IF #statMassCTACl <= 30 THEN
    #tempValveCTACl := TRUE;
ELSE

    #tempValveCTACl := FALSE;
    #statStartTimer := TRUE;
END_IF;

// Timer setup 15 min.
"IEC_Timer_15m_DB".TON(IN := #statStartTimer,
                       PT := T#15M,
                       Q => #statTimerDone);

// When timer is done, reset timer and change state.
IF #statTimerDone AND (#tempValveCTACl = FALSE) THEN
    #statResetMass := TRUE;
    #statStartTimer := FALSE;
    #tempState := 3;
END_IF;
```

Figure 6.1.7: Overview of the 2_$CTACl$ FB

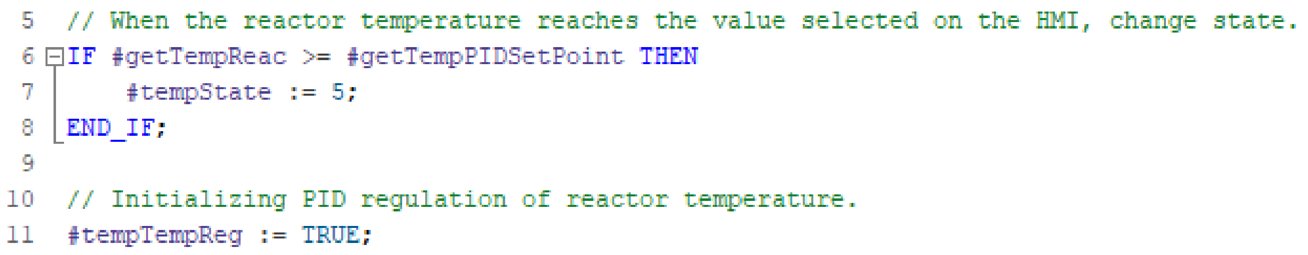
3_Silicon:
This FBs goal is to add the correct amount of SiO_2 nanoparticles to the reactor and to run the sonicator for 30 minutes to agitate the particles. A snippet of the script, for this FB, is shown in Figure 6.1.8. The IF statement, between line 15 and 21 decides whether to open or close the valve. When $statMassSiO2NPs$ is equal to or lower than 2 kg, is $tempValveSiO2NPs$ $TRUE$, meaning the valve is open. When the value of $statMassSiO2NPs$ exceeds 2 kg, $tempValveSiO2NPs$ is set to $FALSE$, closing the valve, $tempSonicator$ is set to $TRUE$, starting the sonicator, and $statStartTimer$ is set $TRUE$, starting the 30 minute timer. The timer, $IEC\_Timer\_30min\_DB$, starts counting when $statStartTimer$ is $TRUE$, and keeps counting while $statStartTimer$ is $TRUE$, if it goes $FALSE$ the timer resets. After 30 minutes the output Q sets $statTimerDone$ to $TRUE$. The last IF statements conditions is satisfied when $statTimerDone$ is $TRUE$, and $tempValveSiO2$ is $FALSE$. This resets the mass and the timer, by setting $statResetMass$ $TRUE$ and $statStartTimer$ $FALSE$, stops the sonicator, by setting $tempSonicator$ $FALSE$, and sets the new step, by setting $tempState$ to 4. The variable, $tempValveSiO2$, $tempSonication$, and $tempState$, are set to their output counterpart on the last three lines in the script.



```
// Closing valve when 2 kg is added and starting the sonicator and the timer.
IF #statMassSiO2NPs <= 2 THEN
    #tempValveSiO2NPs := TRUE;
ELSE
    #tempValveSiO2NPs := FALSE;
    #tempSonicator := TRUE;
    #statStartTimer := TRUE;
END_IF;

// Setup of 30 minute timer.
"IEC_Timer_30min_DB".TON(IN := #statStartTimer,
                         PT := T#30M,
                         ET => #tempTimeElapsed,
                         Q => #statTimerDone);

// Changing state, stopping sonication, and resetting timer when timer (30 min) is done.
IF #statTimerDone AND (#tempValveSiO2NPs = FALSE) THEN
    #statResetMass := TRUE;
    #tempSonicator := FALSE;
    #statStartTimer := FALSE;
    #tempState := 4;
END_IF;
```

Figure 6.1.8: Overview of the 3_*Silicon* FB

4_Heating70:
This FB initiates the regulation of the reactor temperature to seventy degrees Celsius. The script of this FB is shown in Figure 6.1.9 Line 6, sets *tempTempReg TRUE*, initiating the regulation of the reactor temperature. The IF statement sets *tempState* to 5, when the reactor temperature reaches the temperature setpoint selected on the HMI, *getTempPIDSetPoint*. The values of *tempTempReg* and *tempState* are at the end of the script set to their corresponding outputs.



```
 5  // When the reactor temperature reaches the value selected on the HMI, change state.
 6  IF #getTempReac >= #getTempPIDSetPoint THEN
 7      #tempState := 5;
 8  END_IF;
 9
10  // Initializing PID regulation of reactor temperature.
11  #tempTempReg := TRUE;
```

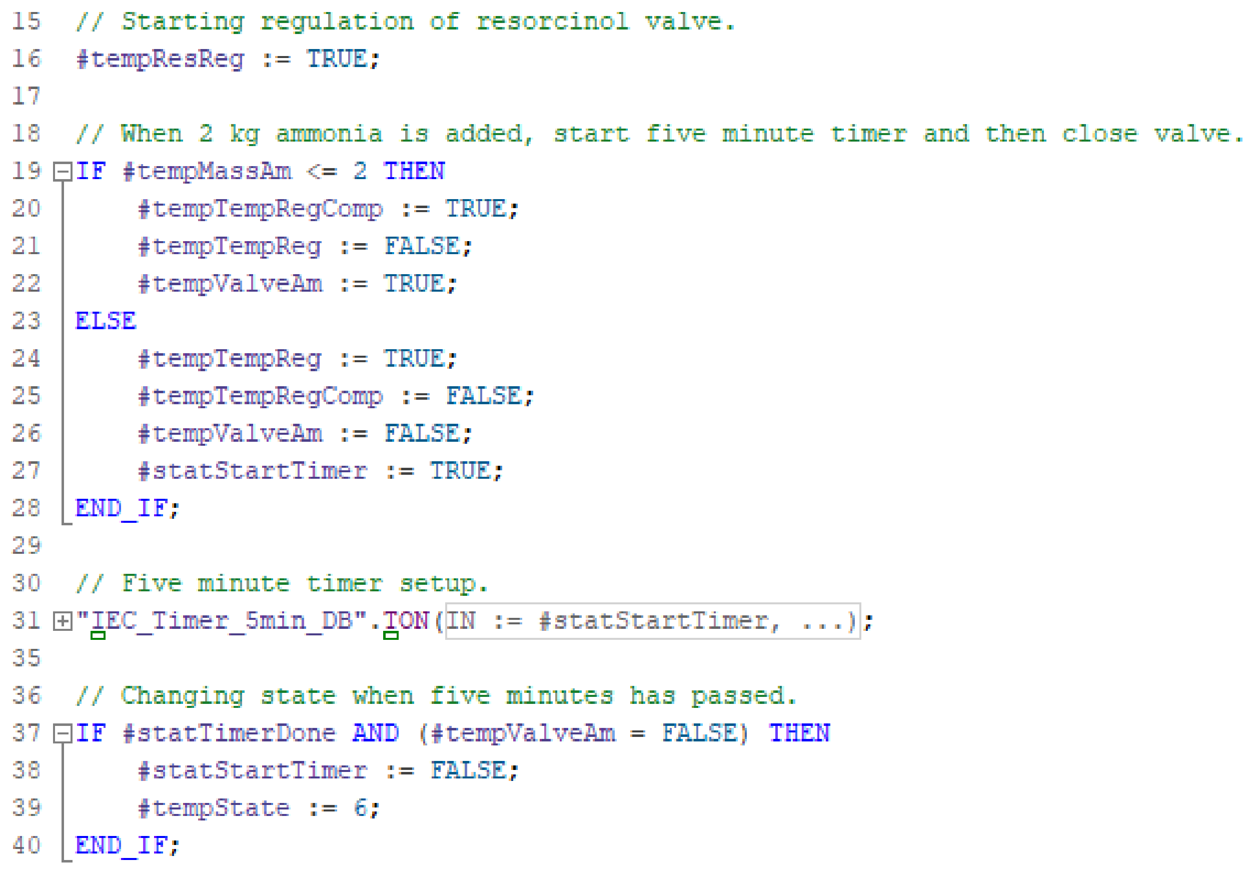
Figure 6.1.9: Overview of the 4_*Heating*70 FB

6.1.6 Coating of Silicon Particles

This section contains the three steps that together represents the stage of the coating of the silicon particles in this process. The three steps in this stage are: 5_*ResandAm*, 6_*Form*, and 7_*CoatingThickness*.

5_ResandAm:
This FB starts the regulation of the resorcinol concentration, and controls the addition of ammonia. The step is switched after 2 kg of ammonia has been added, and 5 minutes has passed. A snippet from the script of this FB is shown in Figure 6.1.10. In line 15, *tempResReg* is set *TRUE*, initiating the PID regulating the resorcinol concentration. The IF statement, between line 19 and 28, decides whether the valve

should be closed or opened. If *tempMassAm* is equal to or lower than 2 kg, *tempValveAm*, goes *TRUE*, opening the valve. The temperature regulation also switches from *tempTempReg* to *tempTempRegComp*, which switches to the PID that is designed to better compensate against a flow of compounds other than resorcinol. If *tempMassAm* is greater than 2 kg, *tempValveAm* goes *FALSE*, closing the valve, and *statStartTimer* goes *TRUE*, starting the 5 minute timer, and *tempTempReg* and *tempTempRegComp* switches again, enabling the ordinary PID. The timer, *IEC_Timer_5min_DB*, counts as long as *statStartTimer* is *TRUE*, if *statStartTimer* goes *FALSE* before the timer is completed, the timer resets. When the timer is completed, *statTimerDone* goes *TRUE*. The last IF statement decides when to change to the next step. This IF statements condition is satisfied when *statTimerDone* is *TRUE* and *tempValveAm* is *FALSE*. This makes *statStartTimer* go *FALSE*, resetting the timer, and sets *setState* to 6. The values of *tempValveAm*, *tempResReg*, and *tempState*, are set on their corresponding outputs on the last three lines in the script.



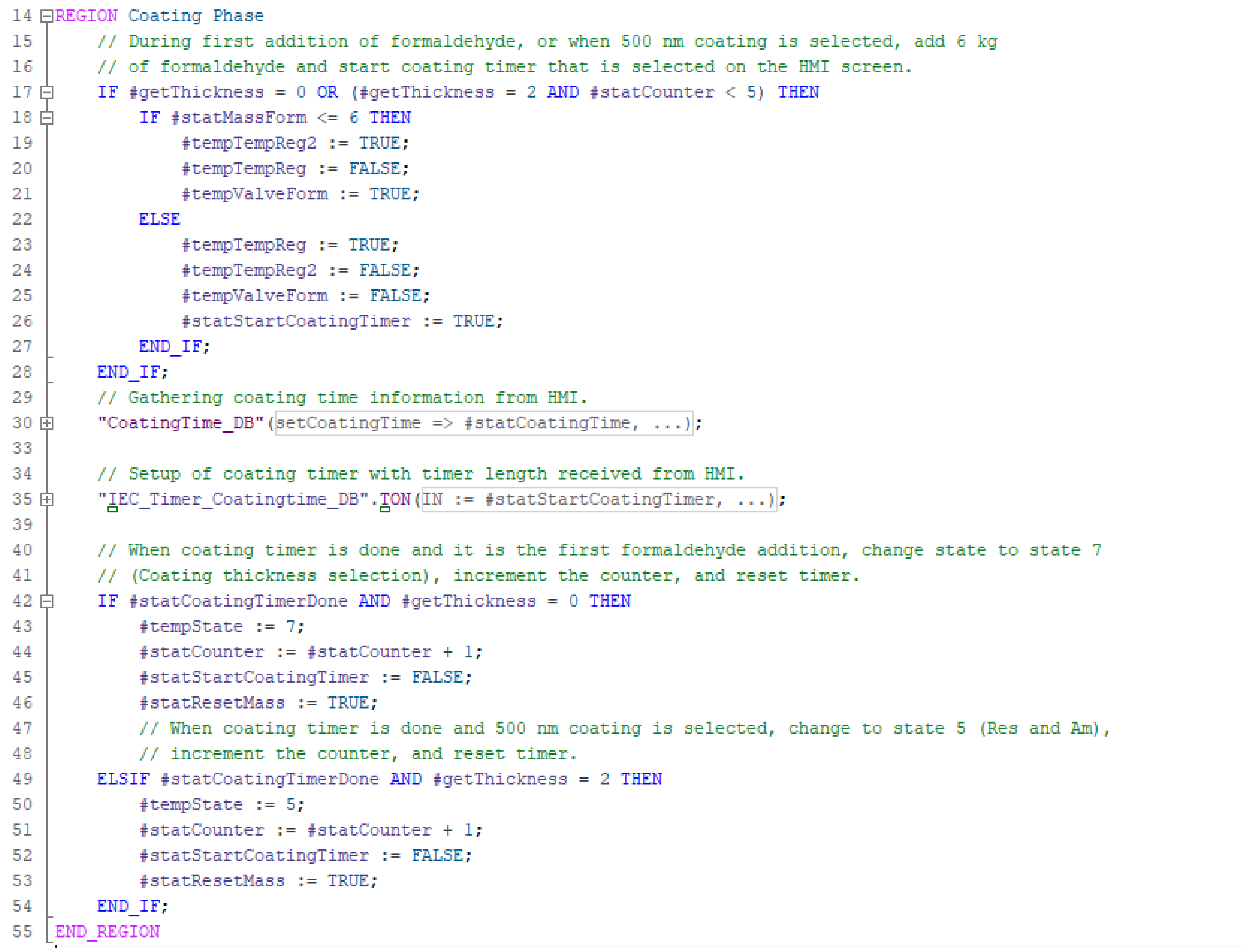
```
// Starting regulation of resorcinol valve.
#tempResReg := TRUE;

// When 2 kg ammonia is added, start five minute timer and then close valve.
IF #tempMassAm <= 2 THEN
    #tempTempRegComp := TRUE;
    #tempTempReg := FALSE;
    #tempValveAm := TRUE;
ELSE
    #tempTempReg := TRUE;
    #tempTempRegComp := FALSE;
    #tempValveAm := FALSE;
    #statStartTimer := TRUE;
END_IF;

// Five minute timer setup.
"IEC_Timer_5min_DB".TON(IN := #statStartTimer, ...);

// Changing state when five minutes has passed.
IF #statTimerDone AND (#tempValveAm = FALSE) THEN
    #statStartTimer := FALSE;
    #tempState := 6;
END_IF;
```

Figure 6.1.10: Overview of the 5_*ResandAm* FB

6_Form:
This FB controls the addition of formaldehyde, this happens in two sequences. Firstly during the coating phase and secondly after the coating phase. These two phases are separated in the script, into two regions, *Coating Phase*, and *Final Form Layer*. Firstly the *Coating Phase* region will be explained, the script for this region is shown in Figure 6.1.11. The first IF statement decides if the process should enter the coating phase region or not. It enters the region if *getThickness* is equal to 0, meaning it is the first coating layer, or *getThickness* is equal to 2, meaning acoating layer of 500 *nm* has been selected on the HMI, and *statCounter* is less than 5, meaning formaldehyde has been added less than five times. If one of these conditions are true, then it will either open the valve or close it, depending on *statMassForm*. If *statMassForm* is equal to or lower than 6 kg, *tempValveForm* goes *TRUE*, opening the valve, and as

with the addition of ammonia in step $5\_ResanAm$, $tempTempReg$ and $tempTempRegComp$ switches, enabling the compensation PID to be used while formaldehyde is added. If $statMassForm$ is greater than 6 kg, $tempValveForm$ goes $TRUE$, closing the valve, $statStartCoatingTimer$ goes $TRUE$, starting the timer, and $tempTempReg$ and $tempTempRegComp$ switches back.

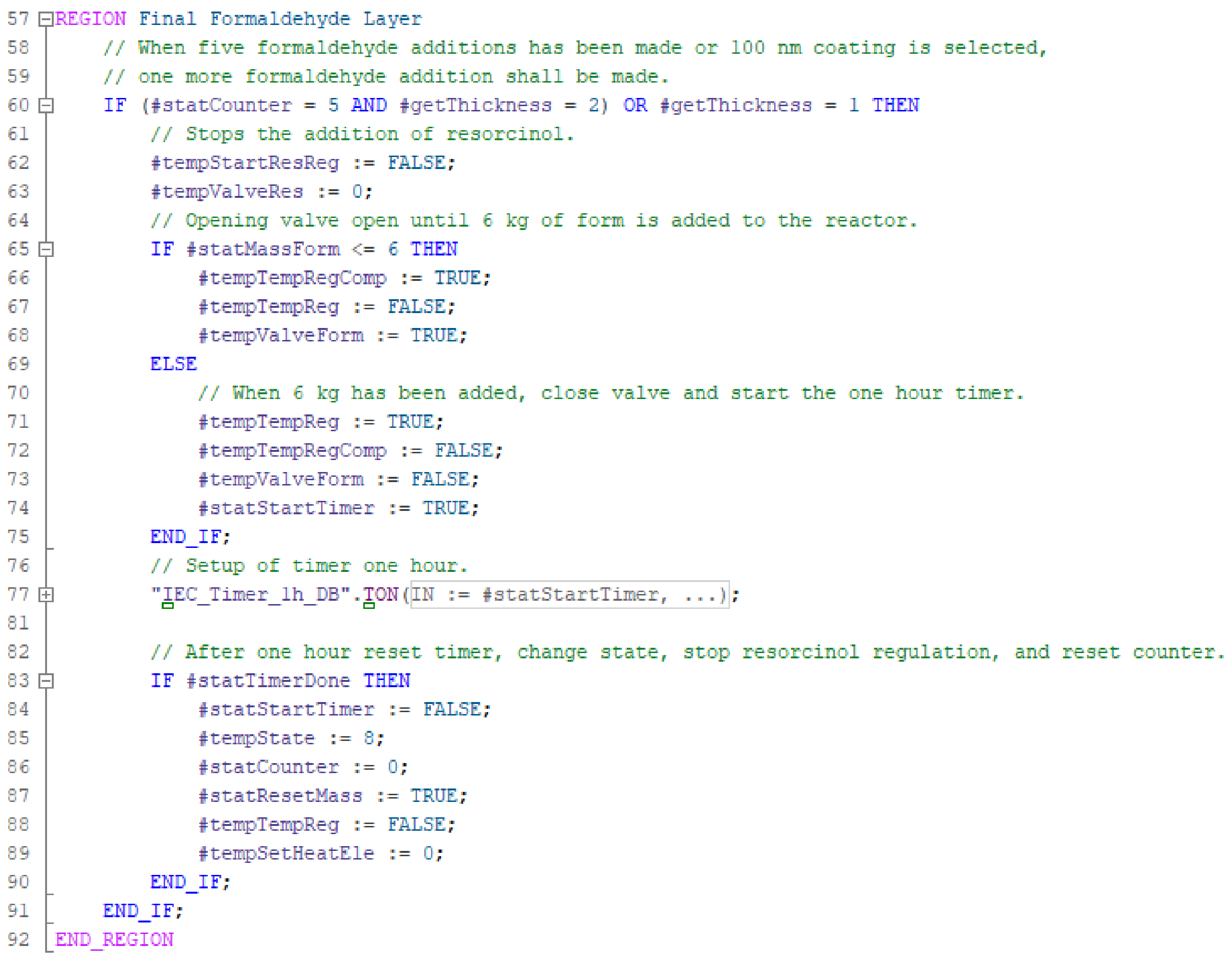
The timer $IEC\_Timer\_Coatingtime\_DB$ starts coating when $statStartCoatingTimer$ goes $TRUE$. the timers length is decided by $statCoatingTime$, which is converted to a time variable in $CoatingTime_DB$, which is based on the input from the HMI. The timer is done when $statCoatingTimerDone$ goes $TRUE$. The FB in this phase changes state based on two values $statCoatingTimerDone$, which indicates whether the timer is done or not, and $getThickness$, which indicates which coating thickness has been chosen. If $getThickness$ is 0, meaning it is the first coating layer run-through, then $tempState$ goes to 7, which is the step that lets the operator decide the coating thickness, $statCounter$ increments one, $statStartCoatingTimer$ goes $FALSE$, resetting the timer, and $statResetMass$ goes $TRUE$, resetting the mass added in this phase. the only difference if $getThickness$ is 2, is that $tempState$ is set to 5, starting a new coating layer. This will repeat itself until $statCounter$ is five. When $statCounter$ is greater than 4, the script will no longer go into the $Coating\ Phase$ region, and the $Final\ Formaldehyde\ Layer$ region is entered.



```
REGION Coating Phase
    // During first addition of formaldehyde, or when 500 nm coating is selected, add 6 kg
    // of formaldehyde and start coating timer that is selected on the HMI screen.
    IF #getThickness = 0 OR (#getThickness = 2 AND #statCounter < 5) THEN
        IF #statMassForm <= 6 THEN
            #tempTempReg2 := TRUE;
            #tempTempReg := FALSE;
            #tempValveForm := TRUE;
        ELSE
            #tempTempReg := TRUE;
            #tempTempReg2 := FALSE;
            #tempValveForm := FALSE;
            #statStartCoatingTimer := TRUE;
        END_IF;
    END_IF;
    // Gathering coating time information from HMI.
    "CoatingTime_DB"(setCoatingTime => #statCoatingTime, ...);

    // Setup of coating timer with timer length received from HMI.
    "IEC_Timer_Coatingtime_DB".TON(IN := #statStartCoatingTimer, ...);

    // When coating timer is done and it is the first formaldehyde addition, change state to state 7
    // (Coating thickness selection), increment the counter, and reset timer.
    IF #statCoatingTimerDone AND #getThickness = 0 THEN
        #tempState := 7;
        #statCounter := #statCounter + 1;
        #statStartCoatingTimer := FALSE;
        #statResetMass := TRUE;
        // When coating timer is done and 500 nm coating is selected, change to state 5 (Res and Am),
        // increment the counter, and reset timer.
    ELSIF #statCoatingTimerDone AND #getThickness = 2 THEN
        #tempState := 5;
        #statCounter := #statCounter + 1;
        #statStartCoatingTimer := FALSE;
        #statResetMass := TRUE;
    END_IF;
END_REGION
```

Figure 6.1.11: Overview of the 6_Form FB

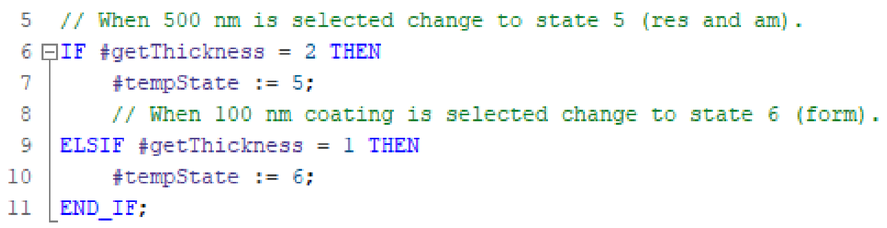
The second region of this FB, the *Final Form Layer* region, is described here. A snippet of this region in the FB can be seen in Figure 6.1.12. This region is only entered if *statCounter* is equal to 5 and *getThickness* is 2, meaning 500 *nm* coating thickness has been selected in step 7_*CoatingThickness*, or *getThickness* is 1, meaning 100 *nm* thickness has been selected in step 7_*CoatingThickness*. If one of these conditions has been complied with, *tempStartResReg* is set to *FALSE*, stopping the PID regulation of the resorcinol concentration, and *tempValveRes* is set to 0, stopping the resorcinol flow. Depending on *statMassForm*, the IF statement between line 62 and 68, will either open or close the formaldehyde valve. If *statMassForm* is equal to or lower than 6 kg, *tempValveRes* goes *TRUE*, opening the valve, and *tempTempReg* and *tempTempRegComp* are switched, enabling the PID design for better compensation against the formaldehyde flow. If it is greater than 6 kg, *tempValveRes* goes *FALSE*, closing the valve, *statStartTimer* goes *TRUE*, starting the timer, and *tempTempReg* and *tempTempRegComp* switches, enabling the normal PID. The timer *IEC*_*Timer*_1*h*_*DB* is started when *statStartTimer* goes *TRUE*. It counts for 60 minutes, as long as *statStartTimer* stays *TRUE*. After 60 minutes, when the timer is completed, *statTimerDone* goes *TRUE*. The last IF statement decides when to change to the next state, this happens when *statTimerDone* is *TRUE*. When this happens is *statStartTimer* set *FALSE*, *statCounter* set to 0, *statResetMass* set to *TRUE*, and *tempState* set to 8. This resets the timer, counter and the mass added, and switches to step 8. On the last five lines is *tempValveRes*, *tempValveForm*, *setStartReg*, *tempState*, and *elapsedCoatingTime* set to their corresponding outputs.



```
57 REGION Final Formaldehyde Layer
58     // When five formaldehyde additions has been made or 100 nm coating is selected,
59     // one more formaldehyde addition shall be made.
60     IF (#statCounter = 5 AND #getThickness = 2) OR #getThickness = 1 THEN
61         // Stops the addition of resorcinol.
62         #tempStartResReg := FALSE;
63         #tempValveRes := 0;
64         // Opening valve open until 6 kg of form is added to the reactor.
65         IF #statMassForm <= 6 THEN
66             #tempTempRegComp := TRUE;
67             #tempTempReg := FALSE;
68             #tempValveForm := TRUE;
69         ELSE
70             // When 6 kg has been added, close valve and start the one hour timer.
71             #tempTempReg := TRUE;
72             #tempTempRegComp := FALSE;
73             #tempValveForm := FALSE;
74             #statStartTimer := TRUE;
75         END_IF;
76         // Setup of timer one hour.
77         "IEC_Timer_1h_DB".TON(IN := #statStartTimer, ...);
81
82         // After one hour reset timer, change state, stop resorcinol regulation, and reset counter.
83         IF #statTimerDone THEN
84             #statStartTimer := FALSE;
85             #tempState := 8;
86             #statCounter := 0;
87             #statResetMass := TRUE;
88             #tempTempReg := FALSE;
89             #tempSetHeatEle := 0;
90         END_IF;
91     END_IF;
92 END_REGION
```

Figure 6.1.12: Overview of the 6_*Form* FB

7_CoatingThickness:
This FB enables the process operator to choose the desired coating thickness on the silicon nanoparticles. The FB is an idle step, meaning the process will remain in this step until a process operator decides the thickness. A snippet form the FB script is shown in Figure 6.1.13. The IF statement decides which state to switch to depending on the coating thickness decided on the HMI. If 500 *nm* is selected, then *getThickness* goes to 2, setting *tempState* to 5. IF 100 *nm* is selected, then *getThickness* goes to 1, setting *tempState* to 6. On the last line of the script the value of *tempState*, is set on the output *setState*, initiating the step switch.



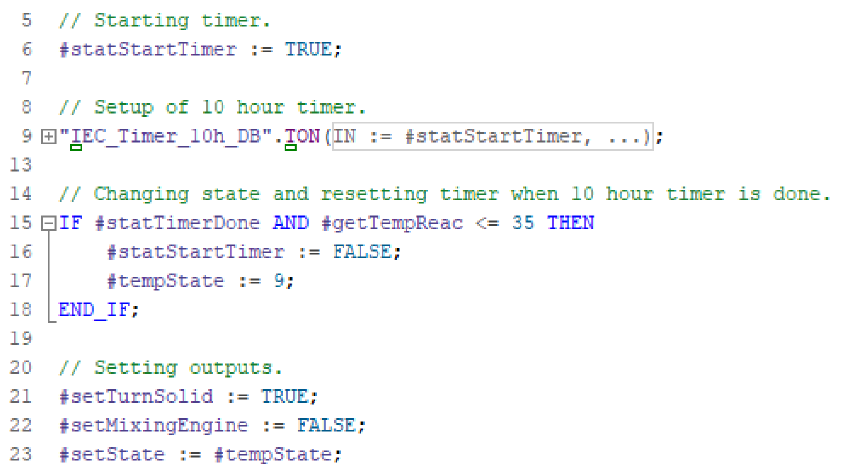
```
// When 500 nm is selected change to state 5 (res and am).
IF #getThickness = 2 THEN
    #tempState := 5;
    // When 100 nm coating is selected change to state 6 (form).
ELSIF #getThickness = 1 THEN
    #tempState := 6;
END_IF;
```

Figure 6.1.13: Overview of the 7_*CoatingThickness* FB

6.1.7 Cleaning and Waste Removal

This section contains the five steps that together represents the stage of the cleaning and waste removal in this process. The five steps in this stage are: 8_*Settling*, 9_*EmptyReactor*, 10_*Washing*, 11_*Drying*, and 12_*ProductRemoval*.

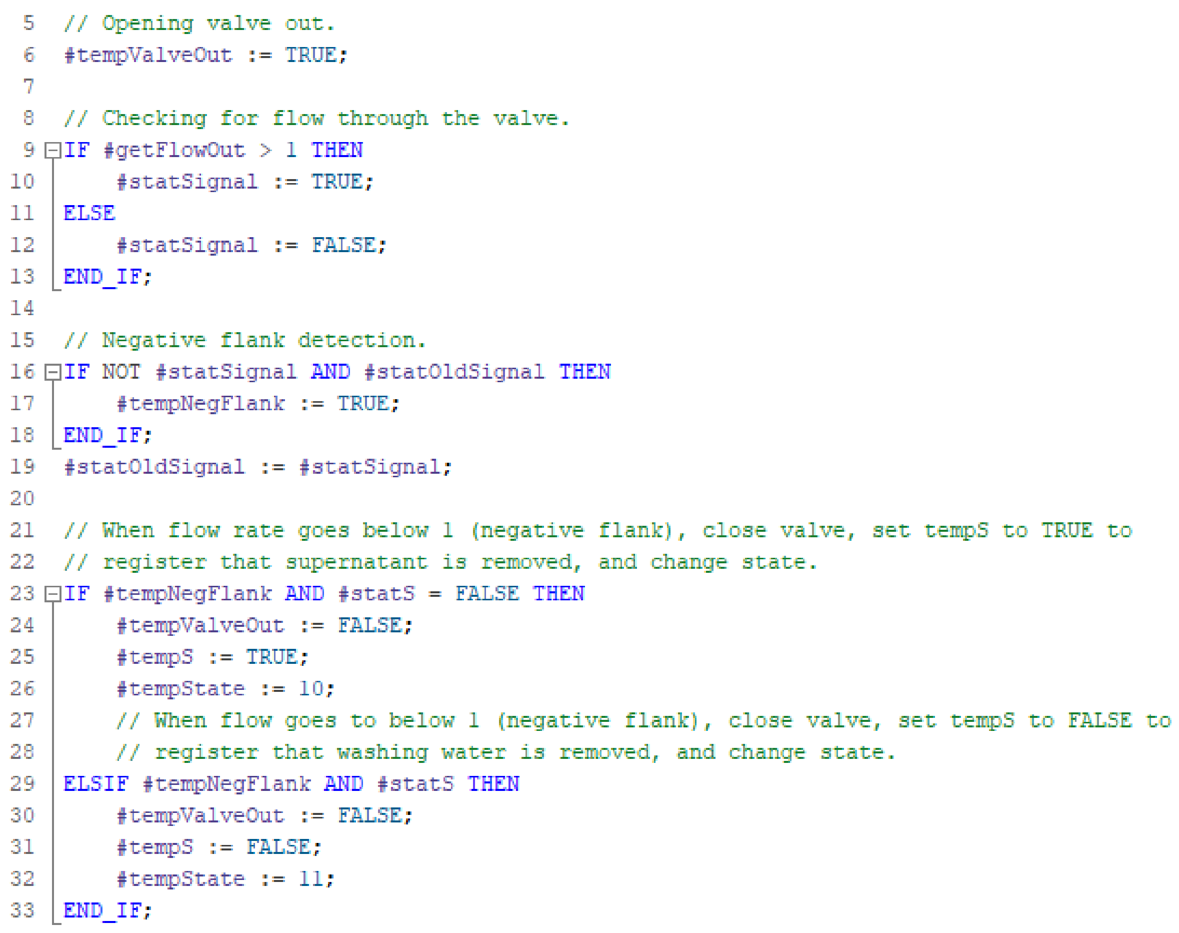
8_Settling:
This FB represents the settling step, in this step the contents of the reactor should be allowed to settle, meaning no stirring or heating is enabled.A snippet form the script for this FB can be seen in Figure 6.1.14. The FB starts by setting *statStartTimer* to *TRUE*, starting the timer. The timer, *IEC_Timer_10h_DB*, start counting when *statStartTimer* goes *TRUE*. The timer counts 10 hours, before setting the *statTimerDone TRUE*. The IF statement in this FB, decides when to switch to the next step. This happens when *statTimerDone* is *TRUE*, and *getTempReac* is lower than 35 °C. When this happens is *statStartTimer* set to *FALSE*, resetting the timer, and *tempState* is set to 9. On the last three lines of this FB the outputs are set to the following: *setTurnSolid* to *TRUE*, acting as a signal for SIMIT to communicate that the contents of the reactor has settled, *setMixingEngine* to *FALSE*, and *setState* to *tempState*.



```
// Starting timer.
#statStartTimer := TRUE;

// Setup of 10 hour timer.
"IEC_Timer_10h_DB".TON(IN := #statStartTimer, ...);

// Changing state and resetting timer when 10 hour timer is done.
IF #statTimerDone AND #getTempReac <= 35 THEN
    #statStartTimer := FALSE;
    #tempState := 9;
END_IF;

// Setting outputs.
#setTurnSolid := TRUE;
#setMixingEngine := FALSE;
#setState := #tempState;
```

Figure 6.1.14: Overview of the 8_*Settling* FB

9_EmptyReactor:
This FB controls the emptying of the waste from the reactor. This FB is activated twice, once after the settling step and once after the washing step. The FBs script can be seen in Figure 6.1.15. Firstly, the outlet valve is opened by setting $tempValveOut$ to $TRUE$. The first IF statement coverts the $getFlowOut$ signal to a Boolean, "is flowing" signal, meaning $statSignal$ is $TRUE$ when there is a flow, and $FALSE$ when there is a flow lower than 1. The reactor can be considered empty at a flow of 1 because the there is no longer any pressure above the valve, meaning the residual liquids in the reactor are so small that they can be neglected, they will nevertheless disappear in the drying stage. The second IF statement is an edge-detector. It checks $statSignal$ for a negative edge. IF $statSignal$ has a negative edge, $tempNegFlank$ goes $TRUE$. The IF statement between line 23 and 33 decides which step to change to depending on two factor, firstly if $tempNegFlank$ is $TRUE$, meaning the reactor is empty for liquids, and secondly if $statS$ is $TRUE$ or $FALSE$. IF $statS$ is $FALSE$, this means that the previous step was the settling step, if $statS$ is $TRUE$, this means that the previous step was the washing step. IF $statS$ is $FALSE$, and $tempNegFlank$ is $TRUE$, then $tempValveOut$ is set $FALSE$, closing the valve, $tempS$ is set to $TRUE$, indicating that the next time this step is run, the previous step was the washing step, and $tempState$ is set to 10. IF $statS$ and $tempNegFlank$ is $TRUE$, then $tempValveOut$ is set $FALSE$, closing the valve, $tempS$ is set to $FALSE$, indicating that the next time this step is run, the previous step was the settling step, and $tempState$ is set to 11. The outputs are set on the last two lines of the script. $tempS$ is set to $statS$ to save its value when the step changes.



```
 5  // Opening valve out.
 6  #tempValveOut := TRUE;
 7
 8  // Checking for flow through the valve.
 9  IF #getFlowOut > 1 THEN
10      #statSignal := TRUE;
11  ELSE
12      #statSignal := FALSE;
13  END_IF;
14
15  // Negative flank detection.
16  IF NOT #statSignal AND #statOldSignal THEN
17      #tempNegFlank := TRUE;
18  END_IF;
19  #statOldSignal := #statSignal;
20
21  // When flow rate goes below 1 (negative flank), close valve, set tempS to TRUE to
22  // register that supernatant is removed, and change state.
23  IF #tempNegFlank AND #statS = FALSE THEN
24      #tempValveOut := FALSE;
25      #tempS := TRUE;
26      #tempState := 10;
27      // When flow goes to below 1 (negative flank), close valve, set tempS to FALSE to
28      // register that washing water is removed, and change state.
29  ELSIF #tempNegFlank AND #statS THEN
30      #tempValveOut := FALSE;
31      #tempS := FALSE;
32      #tempState := 11;
33  END_IF;
```

Figure 6.1.15: Overview of the 9_*EmptyReactor* FB

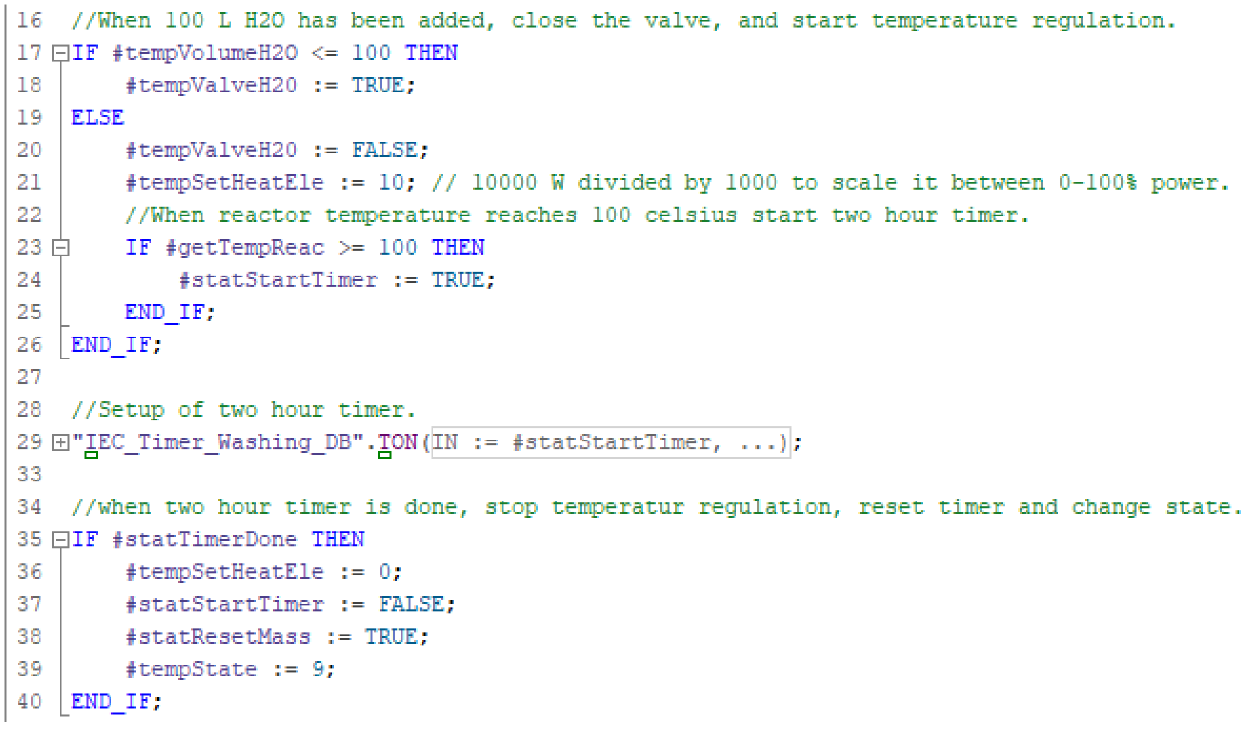
10_Washing:
This FB, controls the washing of the coated silicon nanoparticles. The particles are boil washed for two hours at around 100 °C. A snippet from the FBs script can be seen in Figure 6.1.16. The region $Calculation\ of\ Mass$ calculates the mass of H_2O accumulated in the reactor. The IF statement start the addition of H_2O. If the H_2O added in this step, $tempVolumeH2O$, is equal to lower than 100 L, then $tempValveH2O$ goes $TRUE$, opening the valve. If $tempVolumeH2O$ exceeds 100 L, then $tempValveH2O$ goes $FALSE$, closing the valve, and the heating element is set to 8100 W. This was calculated as follows:

$$\begin{aligned} T_h &= \frac{h_{reac} \cdot A_{reac} \cdot (T_{reac} - T_{Ambient})}{h_h \cdot A_h} + T_{reac} = \frac{h_{reac} \cdot \frac{2 \cdot m_{CNP} + m_{H2O}}{r \cdot \rho_{H2O}} \cdot (T_{reac} - T_{Ambient})}{h_h \cdot A_h} + T_{reac} \\ &= \frac{7176 \cdot 2 \cdot \frac{110}{0.4 \cdot 1000} \cdot (100 - 22.5)}{72000 \cdot 1.667} + 100 = 104.028 \quad [°C] \end{aligned} \tag{6.1}$$

the value for $m_{CNP} + m_{H2O}$ is set to 110 to add some calculation margins to ensure that the temperature will exceed 100 °C. ρ_{H2O} is used as the majority of the mass is water, making the solution having a density a little over water density.

$$P = \frac{h_h \cdot A_h \cdot (T_h - T_{reac})}{60} = \frac{72000 \cdot 1.667 \cdot (104.028 - 100)}{60} = 8057.6 \approx 8100 \quad [W] \tag{6.2}$$

It is rounded up to 10000, to ensure that it goes to a boiling point quick enough. When the real process is built this step should be regulated in order to reach boiling point quicker, but in this project the focus has been on the coating phase. This needs to be scaled down from [0, 100000] to [0, 100], due to the range of the PID having this new scale, and therefore the SIMIT model compensates for this. That is why line 21, in the script, sets $tempSetHeatingEle$ to 10.



```
16   //When 100 L H2O has been added, close the valve, and start temperature regulation.
17   IF #tempVolumeH2O <= 100 THEN
18       #tempValveH2O := TRUE;
19   ELSE
20       #tempValveH2O := FALSE;
21       #tempSetHeatEle := 10; // 10000 W divided by 1000 to scale it between 0-100% power.
22       //When reactor temperature reaches 100 celsius start two hour timer.
23       IF #getTempReac >= 100 THEN
24           #statStartTimer := TRUE;
25       END_IF;
26   END_IF;
27
28   //Setup of two hour timer.
29   "IEC_Timer_Washing_DB".TON(IN := #statStartTimer, ...);
33
34   //when two hour timer is done, stop temperatur regulation, reset timer and change state.
35   IF #statTimerDone THEN
36       #tempSetHeatEle := 0;
37       #statStartTimer := FALSE;
38       #statResetMass := TRUE;
39       #tempState := 9;
40   END_IF;
```

Figure 6.1.16: Overview of the 10_$Washing$ FB

When the temperature reaches 100 °C, *statStartTimer* goes $TRUE$, starting the timer. The timer, $IEC\_Timer\_Washing\_DB$, starts when *statStartTimer* goes $TRUE$, and counts for two hours as long as *statStartTimer* stays $TRUE$. When the timer is completed, *statTimerDone* goes $TRUE$. The last IF statement decides when to change to the next step. The IF statement condition is when *statTimerDone* is $TRUE$. When this is complied with, are the following variables set: *tempSetHeatingEle* = 0, stopping the heating, *statStartTimer* = $FALSE$, resetting the timer, $statResetMass = TRUE$, resetting the mass, and *tempState* = 9. $tempValveH2O$, *tempSetHeatingEle*, and *tempState* are all set to their corresponding outputs on the last three lines of the script.

11_Drying:
This FB controls the drying of the coated silicon nanoparticles. A snippet from the FB can be seen in Figure 6.1.17. The FB starts by setting the heating element to 150 W, this value is calculated from the following equation:

$$\begin{aligned} T_h &= \frac{h_{reac} \cdot A_{reac} \cdot (T_{reac} - T_{Ambient})}{h_h \cdot A_h} + T_{reac} = \frac{h_{reac} \cdot \frac{2 \cdot m_{CNP}}{r \cdot \rho_{CNP}} \cdot (T_{reac} - T_{Ambient})}{h_h \cdot A_h} + T_{reac} \\ &= \frac{7176 \cdot \frac{2+1.44 \cdot 2}{0.4 \cdot 950} \cdot (70 - 22.5)}{72000 \cdot 1.667} + 70 = 70.073 \quad [°C] \end{aligned} \tag{6.3}$$

where m_{CNP} is the coated nanoparticle mass, and ρ_{tot} is an estimation of the density. With this heating element temperature calculated the corresponding power on heating element to reach this temperature can also be calculated.

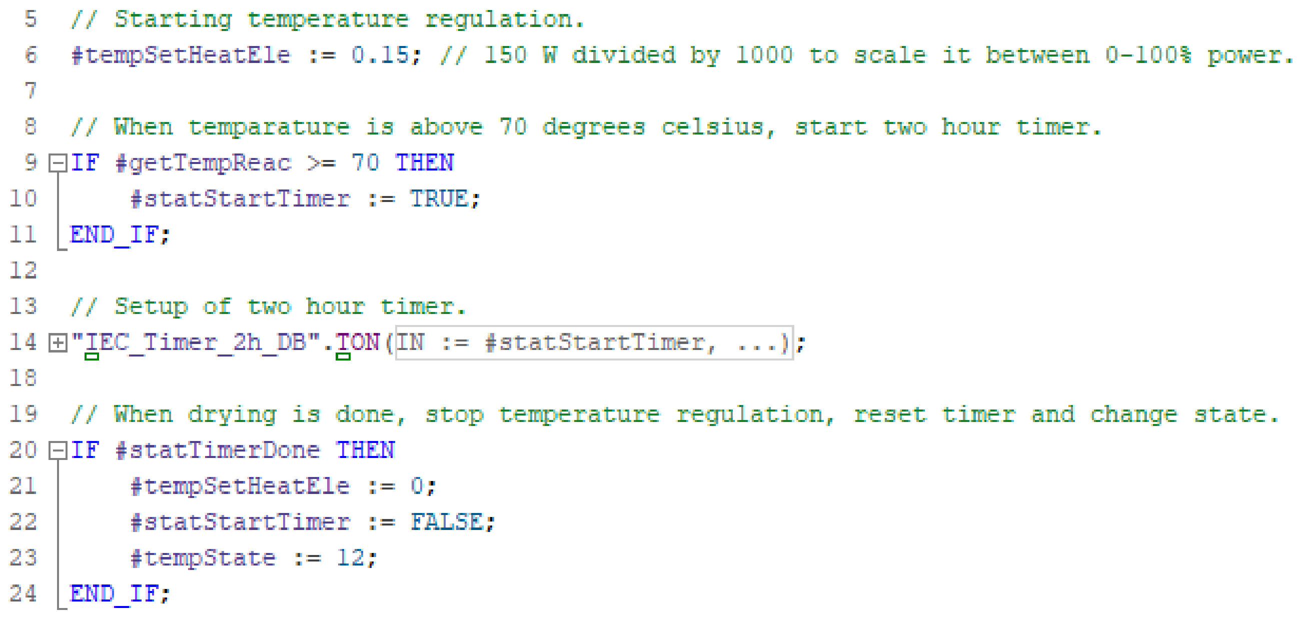
$$P = \frac{h_h \cdot A_h \cdot (T_h - T_{reac})}{60} = \frac{72000 \cdot 1.667 \cdot (70.073 - 70)}{60} = 146 \approx 150 \quad [W] \tag{6.4}$$

As in the previous step, this needs to be scaled down from [0, 100000] to [0, 100], due to the range of the PID having this scale, and therefore the SIMIT model compensates for this. That is why line 6 in the script sets *tempSetHeatingEle* to 0.15. It was decided that the temperature does not need regulating as the step is time consuming and no disturbances are present. Therefore, the power on the element is set to a constant value.

When the temperature, *getTempReac*, is equal to or greater than 70 °C, *statStartTimer* will go $TRUE$, starting the timer. The timer, $IEC\_Timer\_2h\_DB$, starts when *statStartTimer* goes $TRUE$, after the two hours has passed, will *statTimerDone* go $TRUE$ indicating that the timer is completed. When the timer is completed, with *statTimerDone* $TRUE$, the conditions in the last IF statement has been complied with, and the following variables are set: *tempSetHeatingEle* to 0, stopping the drying, *statStartTimer* to $FALSE$, resetting the timer, and *tempState* to 12. *tempState* and *tempSetHeatingEle*, are set to their corresponding outputs on the last to lines of the FB.

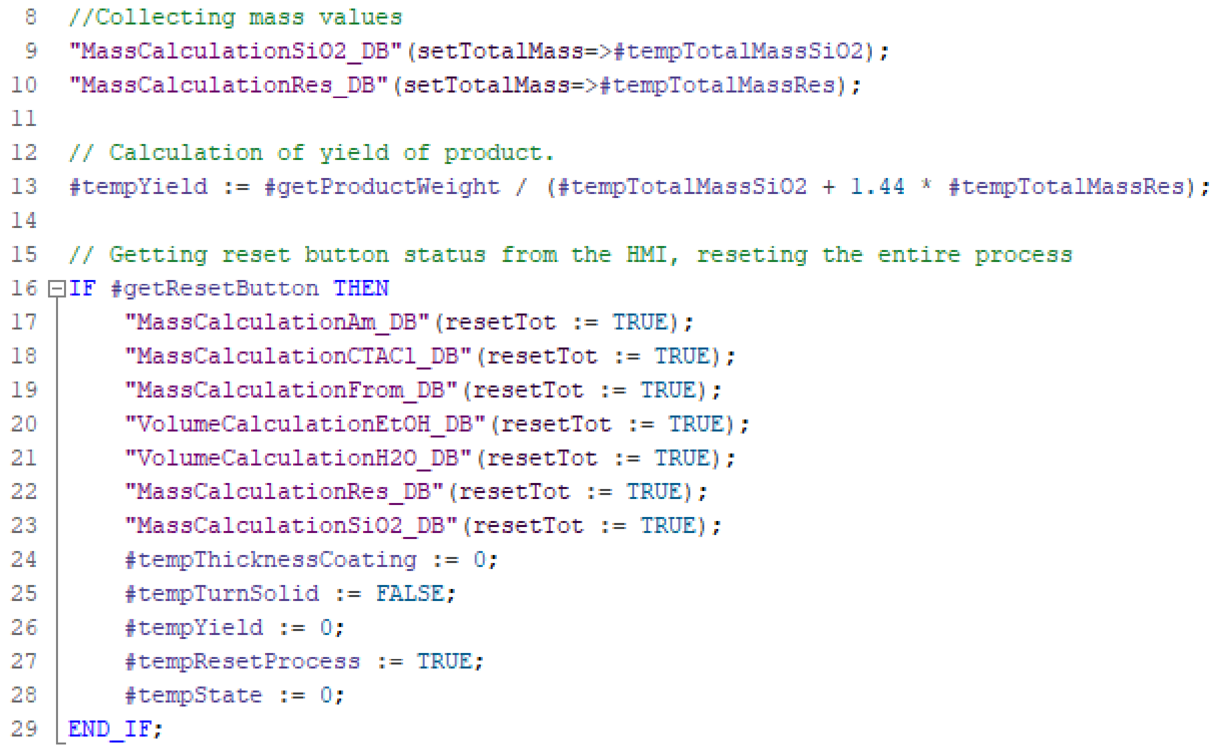
12_ProductRemoval:
This FBs task is to calculated the yield of the process and act as an idle state until the coated nanoparticles has been removed. A snippet from this FBs script is shown in Figure 6.1.18. In this step the yield value that is shown on the HMI is calculated. Firstly, the total value of mass of SiO_2 and resorcinol added to the reactor is extracted form their corresponding mass calculation DB on line 6 and 7. On line 10, *tempYield* is the percent value of yield calculated form the total weight of the coated nanoparticles, *getProductWeight*, divided by the weight of silicon added to the reactor plus the wight of resorcinol multiplied by 1.44. The IF statement decides when to reset the entire process. This happens when a button is pushed on the HMI. When the button is pushed, goes *getResetButton* $TRUE$, satisfying the condition of the IF statement. This resets the following variables: All the individual masses, gets *resetTot* set to $TRUE$, *tempThicknessCoating* is set to 0, *tempYield* is set to 0, *tempResetProcess* is set to $TRUE$,

tempTurnSolid is set to *FALSE* and *tempState* is set to 0. *tempResetProcess*, *tempThicknessCoating*, *tempTurnSolid*, *tempYield*, and *tempState* are set to their corresponding output on the last three lines of the script.



```
// Starting temperature regulation.
#tempSetHeatEle := 0.15; // 150 W divided by 1000 to scale it between 0-100% power.

// When temparature is above 70 degrees celsius, start two hour timer.
IF #getTempReac >= 70 THEN
    #statStartTimer := TRUE;
END_IF;

// Setup of two hour timer.
"IEC_Timer_2h_DB".TON(IN := #statStartTimer, ...);

// When drying is done, stop temperature regulation, reset timer and change state.
IF #statTimerDone THEN
    #tempSetHeatEle := 0;
    #statStartTimer := FALSE;
    #tempState := 12;
END_IF;
```

Figure 6.1.17: Overview of the 11_*Drying* FB



```
//Collecting mass values
"MassCalculationSiO2_DB"(setTotalMass=>#tempTotalMassSiO2);
"MassCalculationRes_DB"(setTotalMass=>#tempTotalMassRes);

// Calculation of yield of product.
#tempYield := #getProductWeight / (#tempTotalMassSiO2 + 1.44 * #tempTotalMassRes);

// Getting reset button status from the HMI, reseting the entire process
IF #getResetButton THEN
    "MassCalculationAm_DB"(resetTot := TRUE);
    "MassCalculationCTACl_DB"(resetTot := TRUE);
    "MassCalculationFrom_DB"(resetTot := TRUE);
    "VolumeCalculationEtOH_DB"(resetTot := TRUE);
    "VolumeCalculationH2O_DB"(resetTot := TRUE);
    "MassCalculationRes_DB"(resetTot := TRUE);
    "MassCalculationSiO2_DB"(resetTot := TRUE);
    #tempThicknessCoating := 0;
    #tempTurnSolid := FALSE;
    #tempYield := 0;
    #tempResetProcess := TRUE;
    #tempState := 0;
END_IF;
```

Figure 6.1.18: Overview of the 12_*ProductRemoval* FB

6.2 Human Machine Interface

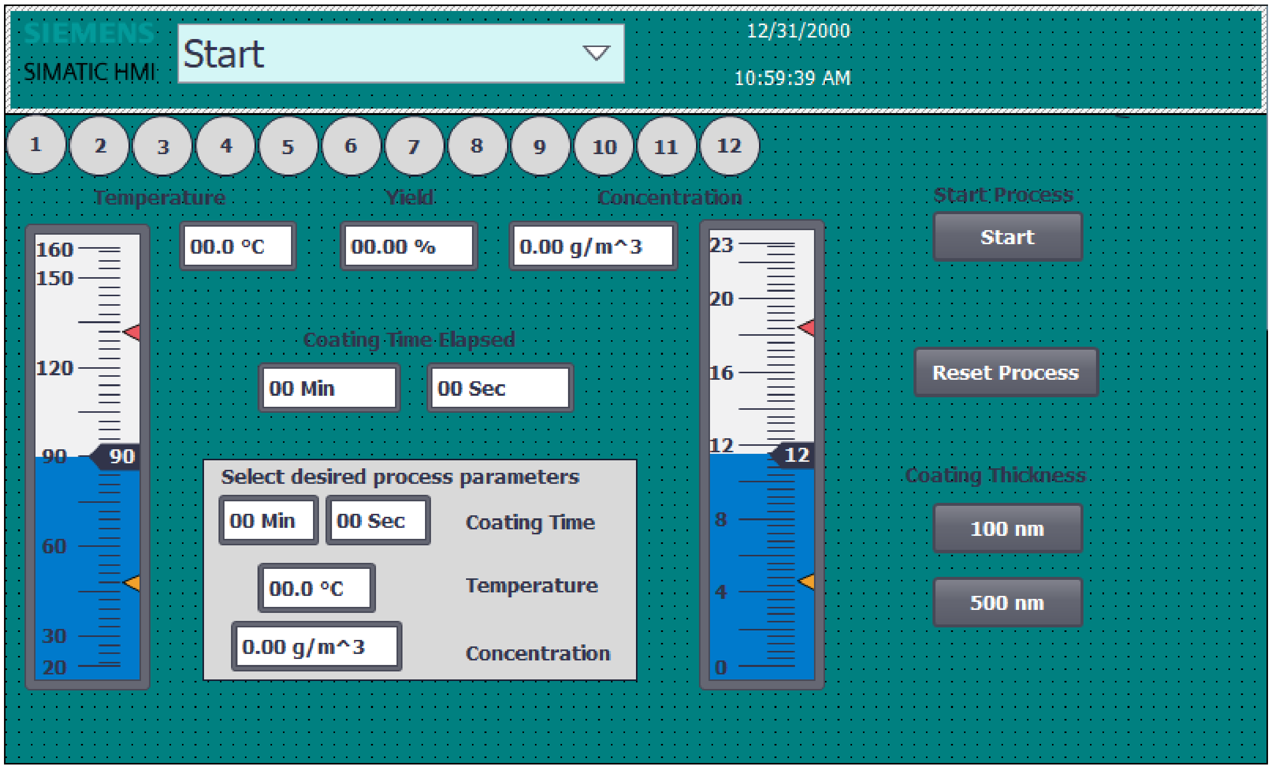
The HMI is made to give the process operator the ability to monitor the process, get an overview of which components that are operating and which step the process is operating in. As well as an overview of the values of the key parameters. The HMI also gives the operator the ability to control the temperature and resorcinol concentration setpoint values, and the coating time of the process. Each screen can be selected in the curtain menu seen on top of the screen in Figure 6.2.1, where it says *Start*.

6.2.1 Process Manipulation and Monitoring

The process manipulation is handled on one screen, the screen called *Start*. The screen also gives the operator an overview of the key parameters of the process, temperature and concentration, an indication of the current step and the yield of the process. The screen can be seen in Figure 6.2.1. The screen has four input fields, the desired coating time in minutes and seconds, the desired reactor temperature, and the desired resorcinol concentration. These fields are the ones in the gray box in Figure 6.2.1. These four inputs are then saved in the PLC/HMI interface DB and the PLC extracts this information from this DB when these values are needed in the different steps. The temperature and concentration setpoint values can be manipulated by the operator during the process if needed.

The screen also includes four buttons, the first button, *Start*, initiates the process when it is pressed. This acts as a trigger from step 0 to step 1 in the PLC program. The two other buttons are the 100 nm and 500 nm, where the process operator selects the desired coating thickness. These buttons are not visible until step 7 is initiated. This removes the possibility of unwanted conflicts of the coating thickness being changed midway the process by mistake. The last button on is the *Reset Process* button, this button becomes visible in step 12, and should be pressed when the coated nanoparticles have been removed from the reactor, which sets the process in the idle step 0.

The temperature is shown by a bar meter that indicates if the temperature is within acceptable levels or exceeds the desired interval. A more precise value is shown in the digital display that is placed beside it. The same setup is used for the concentration. On the top of the screen there are twelve circles present, each circle represent one step, when a step is active the corresponding circle will be green. Between the digital displays for temperature and concentration, is the digital display for the yield rate from the process run-through, this is shown as a percent value. The last two digital displays in the middle of the screen under the text field *Coating Time Elapsed*, show the time the nanoparticles have coated for, in minutes and seconds, respectively.



00.0 °C

00.00 %

0.00 g/m^3

00 Min 00 Sec

Select desired process parameters

00 Min 00 Sec	Coating Time
00.0 °C	Temperature
0.00 g/m^3	Concentration

Figure 6.2.1: Overview of the HMI screen *Start*

Chapter 7

Regulation

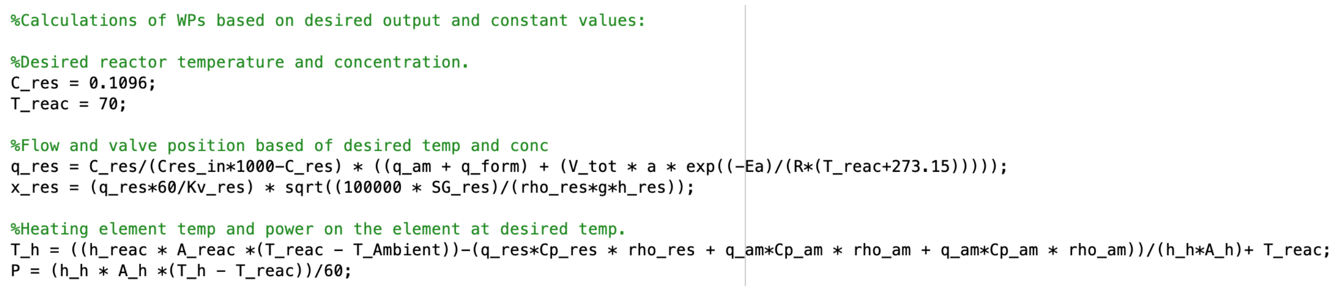
In this chapter, the development of regulation for the resorcinol concentration and temperature in the reactor will be described. Due to the nature of the process, the control of resorcinol concentration and reactor temperature is essential, because it directly impacts the yield of the process. Meaning if these parameters vary a lot or deviated from the desired setpoint, it will have a negative impact on the yield. Therefore, it was wanted to regulate these parameters to a steady-state, even during the presence of disturbances.

To be able to calculate the different transfer functions (TFs), a state-space representation of the process was made. How this was made, and how the general state-space representation is defined, is described in Appendix C. Shortly summarized, a state-space representation is a way to represent the behavior of a physical system mathematically through a set of variables and their relations by first-order differential equations. The sets of variables are sorted into into four vectors and their corresponding matrices, with the matrices containing the differential equations. The vectors are: the state vector $x(t)$, the input vector $u(t)$, the disturbance vector $v(t)$, and the output vector $y(t)$. The values in the matrices corresponding to these vectors decide how much each vector variable affects the change of a state variable. Each value in the matrix is calculated at a defined working point (WP) in the system (a specific phase), making the values differ in other phases of the process. Therefore, it is needed to linearize the state-space at multiple WPs.

In order to determine the different WPs for the different phases and then calculate the different TFs, a Matlab script was made. This script calculates the different matrix values around a set WP and then converts the representation from state-space to TF by using the functions $ss(...)$ and $tf(...)$. This function does the Laplace transformation of the linearized state-space representation and completes the following calculations:

$$H(s) = \frac{y(s)}{u(s)} = C \cdot (sI - A)^{-1} \cdot B = C \cdot \frac{adj(sI - A)}{det(sI - A)} \cdot B \tag{7.1}$$

where $H(s)$ is the TF at the WP decided in the state-space representation. During the calculations of the different TFs, there are only two vector values that are changed, which are the state vector and the input vector. The power on and temperature in the heating element, and the flow out of and position of the valve are the variables having an effect on the desired temperature, concentration, and the values of the other states variables. Consequently, in the Matlab script, the other inputs and state variables are first defined and then calculated as shown in Figure 7.0.1.



```
%Calculations of WPs based on desired output and constant values:

%Desired reactor temperature and concentration.
C_res = 0.1096;
T_reac = 70;

%Flow and valve position based of desired temp and conc
q_res = C_res/(Cres_in*1000-C_res) * ((q_am + q_form) + (V_tot * a * exp((-Ea)/(R*(T_reac+273.15)))));
x_res = (q_res*60/Kv_res) * sqrt((100000 * SG_res)/(rho_res*g*h_res));

%Heating element temp and power on the element at desired temp.
T_h = ((h_reac * A_reac *(T_reac - T_Ambient))-(q_res*Cp_res * rho_res + q_am*Cp_am * rho_am + q_am*Cp_am * rho_am))/(h_h*A_h)+ T_reac;
P = (h_h * A_h *(T_h - T_reac))/60;
```

Figure 7.0.1: Calculation of the WPs.

The WPs of the remaining state variables (mass of compounds) are defined by multiplying the flow rate with density and the time it has flowed. The interested reader can examine the entire Matlab script in Appendix E. The following section will describe how this Matlab script was used to calculate the TFs at different WPs and then the following analysis based on the results.

7.1 Reactor Temperature Regulation

The temperature in the reactor is an important parameter regarding the yield of the process. Therefore, it was wanted to minimize its deviation from the desired value. This issue is solved by designing one or multiple PID-controllers depending on how much the given TF deviates in different phases of the process. The control of temperature is present in the following steps: $4\_Heating70$, $5\_ResandAm$, and 6_Form. Some of these steps have phases that change the WPs of the state-space model. In those steps, an analytical approach will be used to check if only one PID-controller can be used for all the WPs or if the transfer functions deviate too much from each other.

For controlling the reactor temperature, the power applied to the heating element, P, is the control variable of the TF, also known as the input $u(s)$. The control method used is a simple closed-loop feedback. One drawback is that this method is sensitive towards noise, as the noise will get amplified. However, because this process is slow, it was deemed good enough for the use case in this project.

7.1.1 Transfer functions

For the reactor temperature, there are several transfer functions of interest, where the WPs have changed due to addition of different compounds. The phases of interest are in the following steps; one in $4\_Heating70$, two in $5\_ResandAm$, and four in 6_Form.

4_Heating70:
This step aims to heat the contents of the reactor to 70 °C. To calculate the TF for this phase, the WP needs to be defined. The WPs for the states are the following: $T_{reac} = 70°\text{C}$, $m_{res} = 0$ kg, $m_{am} = 0$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg. T_h is calculated in the Matlab script based on the desired reactor temperature, which resulted in $T_h = 78.43°\text{C}$. The resorcinol concentration is zero in this phase, $C_{res} = 0$ $\frac{g}{m^3}$. The input of interest is the power on the heating element, which is calculated to be $P = 16.866$ kW. All other input variables were equal to zero. These WPs yielded the following TF:

$$H_{4,1}(s) = \frac{0.0002059}{s^2 + 6.567 \cdot s + 0.0731} = \frac{2.82 \cdot 10^{-3}}{(89.68 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.2}$$

5_ResandAm:
This step can be separated into two different phases. Firstly, when the addition of resorcinol and ammonia starts, and secondly, after the addition of ammonia has stopped. The two different TFs, representing each phase, will be named $H_{5,1}$ and $H_{5,2}$. Firstly, the WPs for the addition phase will be defined. The WPs are in this phase are the following: By setting $C_{res} = 6.56\frac{g}{m^3}$, and $x_{am} = 1$, the calculated values are $x_{res} = 0.052$, T_h= 84.56°C, $P = 29.130$ kW. All other input variables were zero. These WPs yielded the following TF:

$$H_{5,1}(s) = \frac{0.0002059}{s^2 + 6.575 \cdot s + 0.1263} = \frac{1.63 \cdot 10^{-3}}{(51.91 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.3}$$

The second phase is when the ammonia stops getting added to the reactor, the WPs for this phase can be defined as: $T_{reac} = 70$°C, $C_{res} = 6.56$, $m_{am} = 2$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg, the first phase lasted approximately 1 minute, based on this is $m_{res} = 0.4134$ kg. Based on the this the other state/input variables were calculated to be, $P = 18.067$ kW, $T_h = 79.03$°C, and $x_{res} = 0.0522$. These WPs yielded the following TF:

$$H_{5,2}(s) = \frac{2.056 \cdot 10^{-4}}{s^2 + 6.568 \cdot s + 0.0782} = \frac{2.63 \cdot 10^{-3}}{(83.84 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.4}$$

6_Form:
This step can be separated into four different phases, one during the first formaldehyde addition, one after the first formaldehyde addition, one during the second formaldehyde addition, and finally one after the second formaldehyde addition. The four different TFs are called: $H_{6,1}$, $H_{6,2}$, $H_{6,3}$, and $H_{6,4}$. Firstly, the WPs for the first phase are defined as: $T_{reac} = 70$°C, $m_{am} = 2$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg, $C_{res} = 6.56$, $x_{form} = 1$, the previous phases lasted, in total, approximately six minutes, based on this $m_{res} = 2.488$ kg. The remaining states were calculated to be: $T_h = 91.04$, $P = 42.096$ kW, and $x_{res} = 0.0525$. All other input variables were set to zero. These WPs yielded the following TF:

$$H_{6,1}(s) = \frac{2.056 \cdot 10^{-4}}{s^2 + 6.584 \cdot s + 0.1822} = \frac{1.13 \cdot 10^{-3}}{(35.98 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.5}$$

The second phase is after the first addition of formaldehyde, the WPs for this phase can be defined as the same as the previous step with the exceptions of: $T_h = 79.15$, $P = 18.307$ kW, $m_{res} = 2.905$ kg, $x_{res} = 0.529$, $m_{form} = 6$kg, and $x_{form} = 0$. All other input variables were set to zero. These WPs yielded the following TF:

$$H_{6,2}(s) = \frac{2.056 \cdot 10^{-4}}{s^2 + 6.568 \cdot s + 0.07924} = \frac{2.59 \cdot 10^{-3}}{(82.74 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.6}$$

The third phase is during the second addition of formaldehyde, the WPs for this phase are the following: $T_{reac} = 70$°C, $T_h = 90.93$, $P = 41.876$ kW $m_{am} = 2$ kg, $m_{form} = 6$ kg, $m_{prep} = 548.71$ kg, $C_{res} = 0$, $x_{form} = 1$, $x_{res} = 0$, and $m_{res} = 28.14$ kg. All other input variables were set to zero. These WPs yielded the following TF:

$$H_{6,3}(s) = \frac{2.049 \cdot 10^{-4}}{s^2 + 6.583 \cdot s + 0.1806} = \frac{1.13 \cdot 10^{-3}}{(36.3 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.7}$$

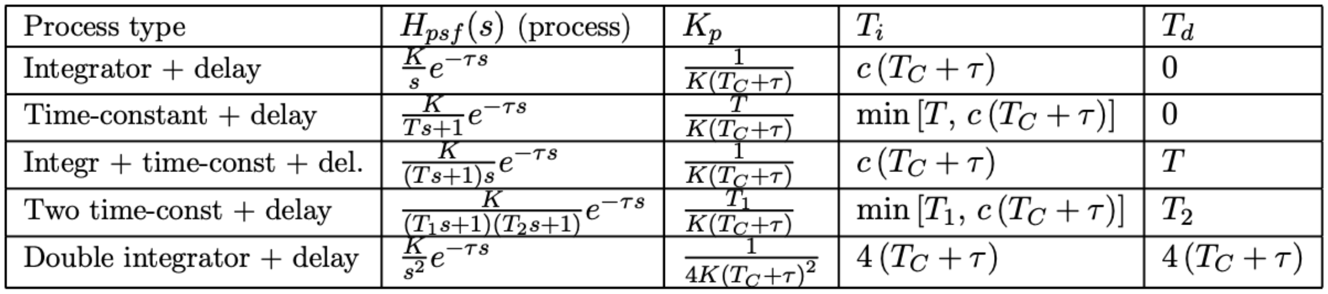
Lastly, the fourth phase is after the second addition of formaldehyde, the WPs for this phase only differ on the following variables from the previous phase: $T_h = 79.03$, $P = 18.065$ kW, $m_{form} = 12$ kg, and $x_{form} = 0$. All other input variables were set to zero. These WPs yielded the following TF:

$$H_{6,4}(s) = \frac{2.047 \cdot 10^{-4}}{s^2 + 6.567 \cdot s + 0.07784} = \frac{2.63 \cdot 10^{-3}}{(84.2 \cdot s + 1)(0.153 \cdot s + 1)} \tag{7.8}$$

The phases where there is only a flow of resorcinol, or no flow at all, have a TF that are very similar to each other, referring to the following TFs: $H_{4,1}(s)$, $H_{5,2}(s)$, $H_{6,2}(s)$, and $H_{6,4}(s)$. The three remaining phases, where other compounds are flowing into the reactor, are also similar to each other. This leads to using two differently tuned PIDs to regulate the temperature, one for each group of phases with similar behavior according to the TFs.

7.1.2 PID tuning

For the tuning, the Skogestad method was used due to its strong results relative to its simplicity. The Skogestad method is a tuning method that uses a number of set equations based on the type of system that is present. The equation table is shown in Figure 7.1.1, based on [7].



Process type	$H_{psf}(s)$ (process)	K_p	T_i	T_d
Integrator + delay	$\frac{K}{s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$c\,(T_C+\tau)$	0
Time-constant + delay	$\frac{K}{Ts+1}e^{-\tau s}$	$\frac{T}{K(T_C+\tau)}$	$\min\,[T,\, c\,(T_C+\tau)]$	0
Integr + time-const + del.	$\frac{K}{(Ts+1)s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$c\,(T_C+\tau)$	T
Two time-const + delay	$\frac{K}{(T_1s+1)(T_2s+1)}e^{-\tau s}$	$\frac{T_1}{K(T_C+\tau)}$	$\min\,[T_1,\, c\,(T_C+\tau)]$	T_2
Double integrator + delay	$\frac{K}{s^2}e^{-\tau s}$	$\frac{1}{4K(T_C+\tau)^2}$	$4\,(T_C+\tau)$	$4\,(T_C+\tau)$

Figure 7.1.1: Skogestad's method equations with different systems

Line number four, in Figure 7.1.1, corresponds to the system order of the temperature TFs. T_1 is the biggest of the two time constants and T_2 is the smallest. $c = 4$ is used to give the best set point tracking, as described in [7]. To calculate the different PID parameters, the time constant, T_C, needs to be decided. T_C is defined as the time constant that you want to design your controller towards, meaning this will be the time constant of the controlled system. Due to the process being relatively slow, a time constant of 10 minutes is chosen and checked if this causes the heating element to go into saturation.

With the time constant being defined, the PID tuning parameters for $H_{6,4}(s)$ were calculated. This specific TF was used because it was approximately the middle value between the other three TFs ($H_{4,1}(s)$, $H_{5,2}(s)$, and $H_{6,2}(s)$) that had similar time constants T_1 and T_2. The PID parameters were the following:

$$K_p = \frac{T_1}{K(T_C + \tau)} = \frac{84.2}{2.63 \cdot 10^{-3} \cdot 10} = 3201.52 \tag{7.9}$$

$$T_i = min[T_1, c(T_C + \tau)] = min[84.2, 4 \cdot 10] = 40 \tag{7.10}$$

$$T_d = T_2 = 0.153 \tag{7.11}$$

Now the tuning parameters for the other group of TFs will be presented. This group consists of $H_{5,1}(s)$, $H_{6,1}(s)$, and $H_{6,3}(s)$. $H_{6,3}(s)$ will be used as a basis because this is the middle value of the three. As with $H_{5,2}(s)$, firstly the time constant must be defined. a faster T_C, of one minute, is chosen due to the fact that it should counter the flows of compounds, which affects the temperature in the reactor quickly.

These flows are considered as disturbances, therefore, the value for c is decided to be 1.4, as according to [7], this is the best value for disturbance compensation.

$$K_p = \frac{T_1}{K(T_C+\tau)} = \frac{36.3}{1.13 \cdot 10^{-3} \cdot 1} = 32113.89 \tag{7.12}$$

$$T_i = min[T_1, c(T_C+\tau)] = min[36.3, 1.5 \cdot 1] = 1.5 \tag{7.13}$$

$$T_d = T_2 = 0.153 \tag{7.14}$$

7.1.3 Analysis

With the two PID controllers being tuned, it is needed to test if the controller regulates the process within a acceptable range. Firstly the controller intended for $H_{4,1}(s)$, $H_{5,2}(s)$, $H_{6,2}(s)$, and $H_{6,4}(s)$. as described in the previous section this PID controller is tuned using $H_{6,2}(s)$ as a basis, but it needs to be tested if these tuning parameters also fit the other three steps in the process. A Simulink scheme was made to test the tuning, this scheme is shown in Figure 7.1.2.

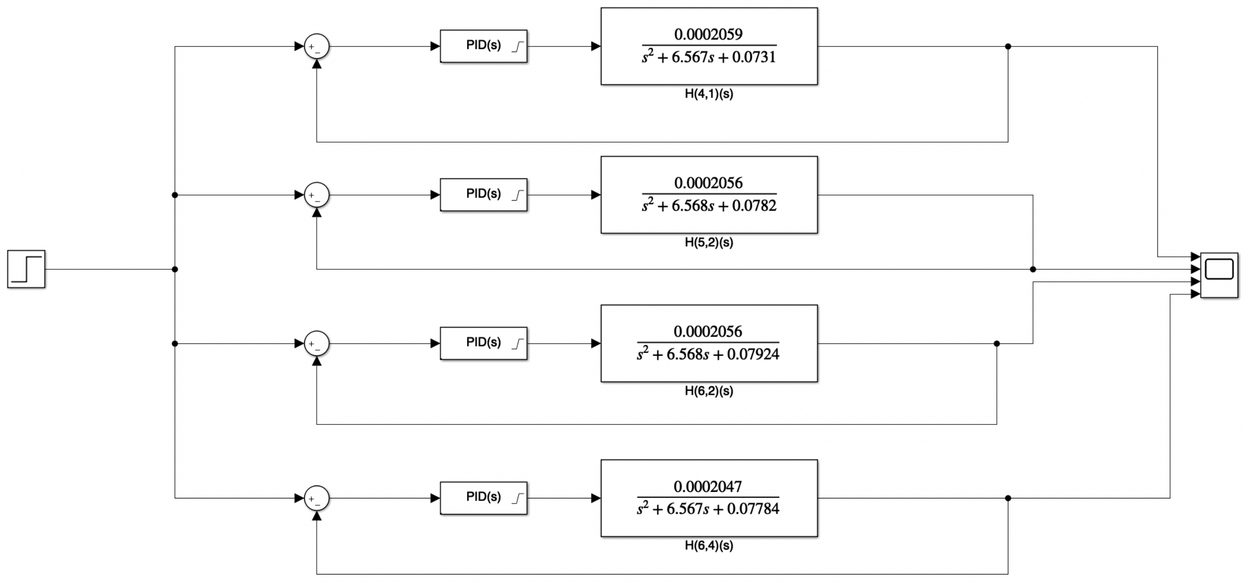

PID(s)
$\frac{0.0002059}{s^2+6.567s+0.0731}$
H(4,1)(s)
PID(s)
$\frac{0.0002056}{s^2+6.568s+0.0782}$
H(5,2)(s)
PID(s)
$\frac{0.0002056}{s^2+6.568s+0.07924}$
H(6,2)(s)
PID(s)
$\frac{0.0002047}{s^2+6.567s+0.07784}$
H(6,4)(s)

Figure 7.1.2: Simulink Scheme to test PID tuning.

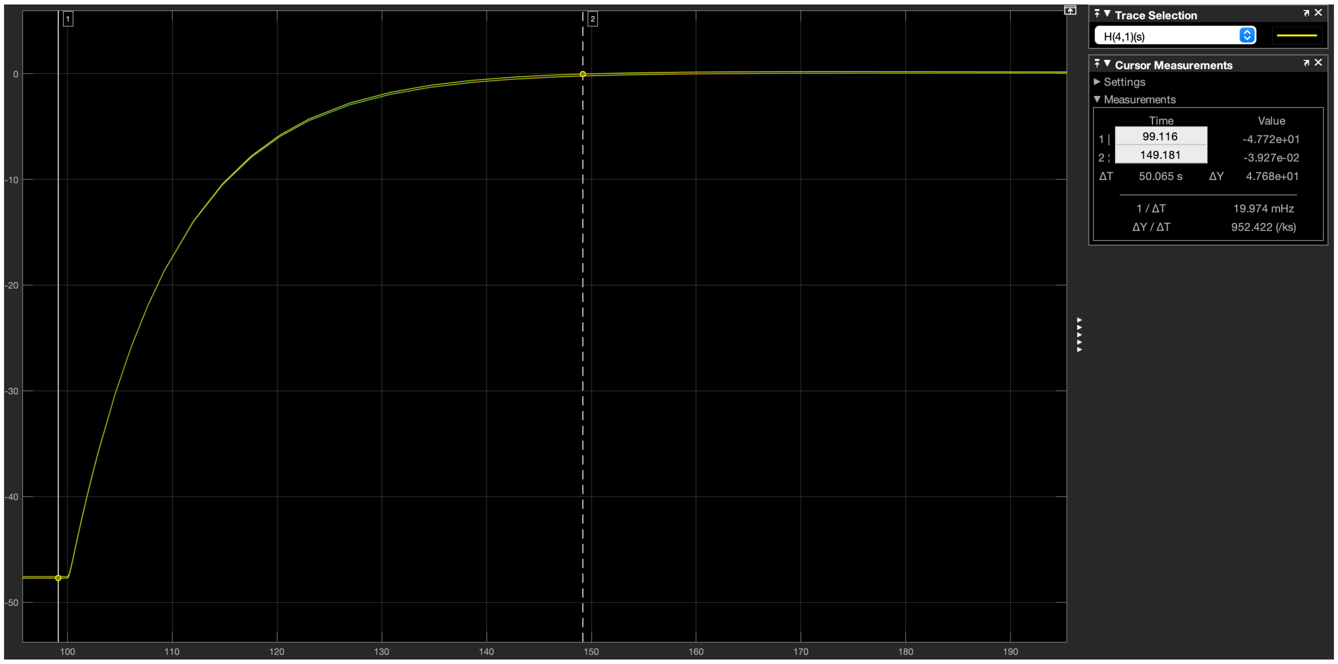
Due to the fact that the PID in Simulink is a parallel PID, and the Skogestad method works under the assumption of a serial PID, conversion from parallel to serial needs to be completed. The conversion calculation is done in accordance with [7].

$$K_{p,p} = K_{p,s} \cdot \left(1 + \frac{T_{d,s}}{T_{i,s}}\right) = 3201.52 \cdot \left(1 + \frac{0.153}{40}\right) = 3213.77 \tag{7.15}$$

$$T_{i,p} = T_{i,s} \cdot \left(1 + \frac{T_{d,s}}{T_{i,s}}\right) = 15 \cdot \left(1 + \frac{0.153}{40}\right) = 40.153 \tag{7.16}$$

$$T_{d,p} = T_{d,s} \cdot \frac{1}{1+\frac{T_{d,s}}{T_{i,s}}} = 0.153 \cdot \frac{1}{1+\frac{0.153}{40}} = 0.152 \tag{7.17}$$

The tuning yielded the step result shown in Figure 7.1.3.



	Time		Value
1	99.116		-4.772e+01
2	149.181		-3.927e-02
ΔT	50.065 s	ΔY	4.768e+01
	1 / ΔT		19.974 mHz
	ΔY / ΔT		952.422 (/ks)

Figure 7.1.3: Step response of $H_{4.1}(s)$, $H_{5.2}(s)$, $H_{6.2}(s)$, and $H_{6.4}(s)$, with tuned PID controller

The step is made from -47.5 to 0, which is equal to a step from 22.5 to 70 due to the WP for these TFs being 70. The controller regulates the temperature to the set point at a acceptable rate, and only deviates with 0.03 after 50 minutes, the system is stable and a bit over-damped.

Secondly, the step response for the other group of TFs needs to be tested in the same way. A Simulink scheme was made to test this as well. This can be seen in Figure 7.1.4.

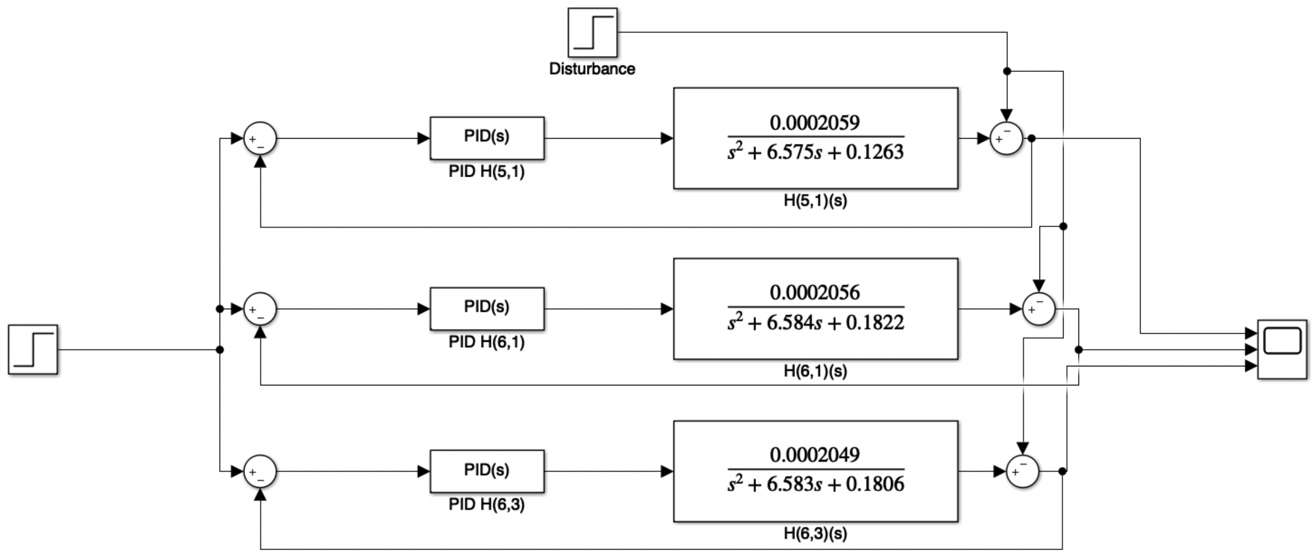

Disturbance
PID(s)
PID H(5,1)
$\frac{0.0002059}{s^2+6.575s+0.1263}$
H(5,1)(s)
PID(s)
PID H(6,1)
$\frac{0.0002056}{s^2+6.584s+0.1822}$
H(6,1)(s)
PID(s)
PID H(6,3)
$\frac{0.0002049}{s^2+6.583s+0.1806}$
H(6,3)(s)

Figure 7.1.4: Simulink Scheme to test PID tuning.

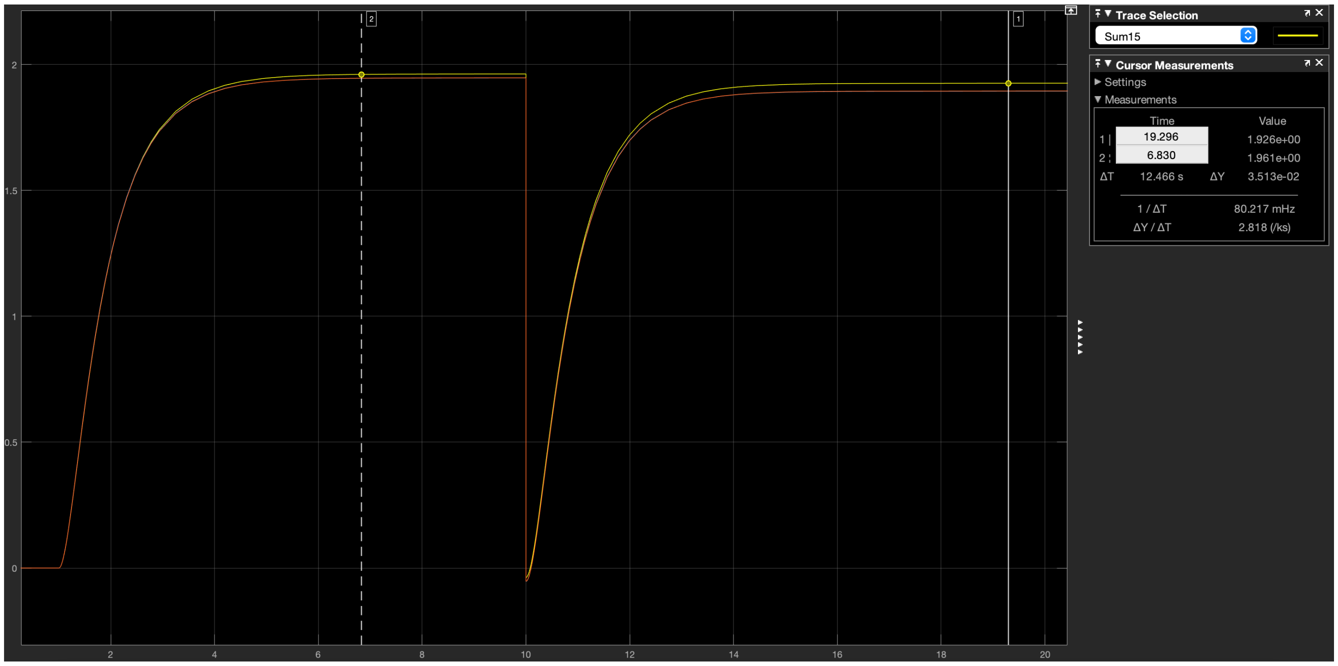
The conversion of the PID parameter for this group of TFs became the following:

$$K_{p,p} = K_{p,s} \cdot \left(1 + \frac{T_{d,s}}{T_{i,s}}\right) = 32113.89 \cdot \left(1 + \frac{0.153}{1.5}\right) = 32441.45 \tag{7.18}$$

$$T_{i,p} = T_{i,s} \cdot \left(1 + \frac{T_{d,s}}{T_{i,s}}\right) = 1.5 \cdot \left(1 + \frac{0.153}{1.5}\right) = 1.653 \tag{7.19}$$

$$T_{d,p} = T_{d,s} \cdot \frac{1}{1 + \frac{T_{d,s}}{T_{i,s}}} = 0.153 \cdot \frac{1}{1 + \frac{0.153}{1.5}} = 0.139 \tag{7.20}$$

The tuning of the PID controller used for this group, yielded the results shown in Figure 7.1.5.



	Time		Value
1	19.296		1.926e+00
2	6.830		1.961e+00
ΔT	12.466 s	ΔY	3.513e-02
	1 / ΔT		80.217 mHz
	ΔY / ΔT		2.818 (/ks)

Figure 7.1.5: Step response of $H_{5.1}(s)$, $H_{6.1}(s)$, and $H_{6.3}(s)$, with tuned PID controller

The TFs that deviates in its values from the other two the most, $H_{5.1}(s)$, has a behavior that differs from the other two, but it tracks to a value that is only marginally lower than the other two. A disturbance was also introduces, at 10 minutes, and all three managed to compensate for this disturbance. The disturbance in the real process will not be as abrupt, so the value wont drop as much form the set point in the real process. The results from the regulation of the tuning parameters implemented to the PID controllers in the TIA Portal program, can be seen in Figure 7.1.6.

Initially the setpoint was set to 70°C, which it reached at about 27 minutes. Then the set point was changed to 8070°C, approximately 30 seconds after it initially reached 70°C. The behavior seems linear, this is due to the fact that the heating element i saturated, meaning it delivers at 100 % capacity, this gives the linear looking behavior. It can also be seen that the linear slope becomes steeper at about 29 minutes, here the flow of ammonia stopped and the PID was switched from the disturbance compensation PID controller, to the tracking one. The total rise time is approximately 6 minutes which is acceptable and the tracking PID regulates the temperature well to 80°C showing that the regulation works as desired. In conclusion, both sets of the tuning parameters for the two PID controllers, yields results that gives desired qualities.

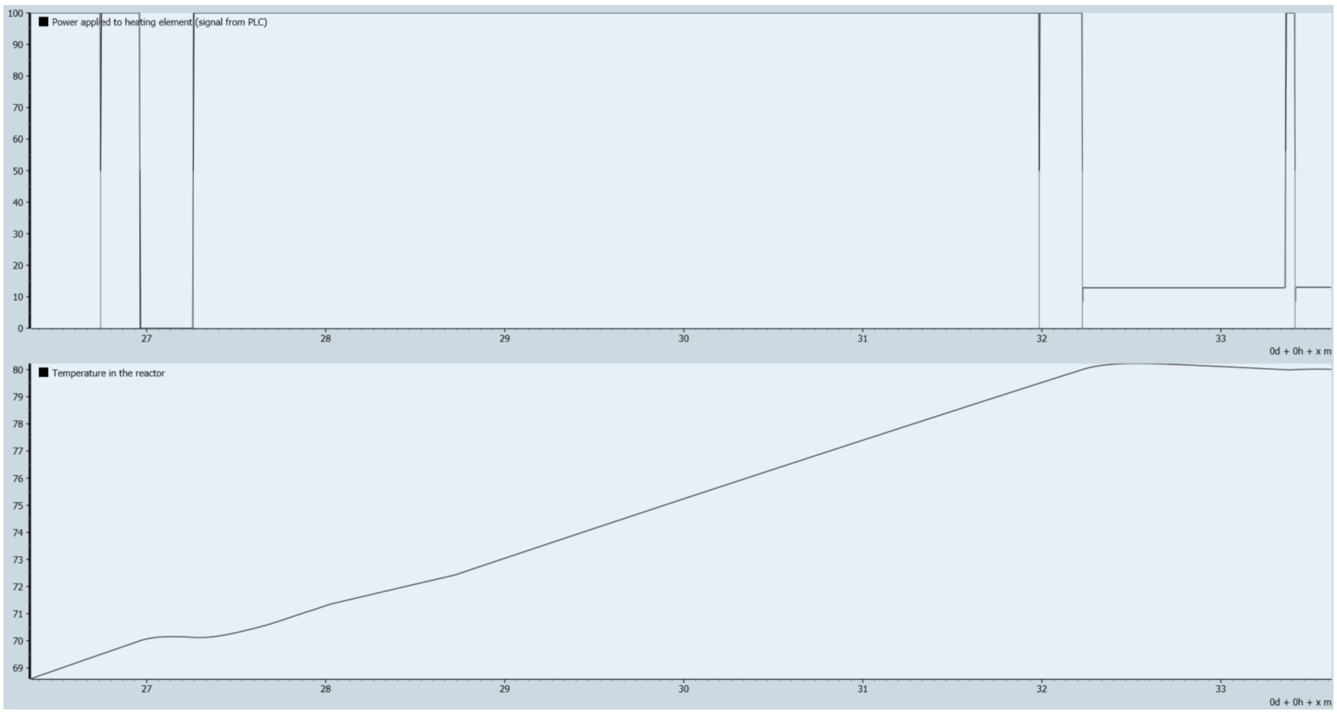


Figure 7.1.6: Temperature behavior of the tuned PID controllers implemented in SIMIT

7.2 Resorcinol Concentration Control

The resorcinol concentration is an essential parameter for the yield of the process, which can be described as the success criteria. To keep this concentration at a steady-state is therefore important. The control is present in five phases, in the following steps: two in 5_*ResandAm*, and two in 6_*Form*.

7.2.1 Transfer function

5_ResandAm:
In this step there are two phases present, one when the resorcinol and ammonia addition starts, and one after the ammonia addition is completed. The TFs, for the two phases will be named $H_{C,5.1}(s)$, and $H_{C,5.2}(s)$. Firstly, the TF for the first phase was calculated. This is done with the Matlab script as with the reactor temperature control TFs. The WPs needs to be defined and are the following: $T_{reac} = 70°\text{C}$, $m_{res} = 0$ kg, $m_{am} = 0$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg, $T_h = 84.56$, $P = 29.130$ kW, $C_{res} = 6.56$, $x_{res} = 0.052$, and $x_{am} = 1$. All order states variables, and inputs are set to be zero. These WPs yielded the following TF:

$$H_{C,5.1}(s) = \frac{991.9 \cdot s^3 + 6521 \cdot s^2 + 121.3 \cdot s - 0.06487}{s^4 + 14.42 \cdot s^3 + 51.72 \cdot s^2 + 0.9907 \cdot s} \tag{7.21}$$

Secondly, the TF for the second phase was calculated. The WPs used to calculate this in the Matlab script were the following: $T_{reac} = 70°\text{C}$, $T_h = 79.03$, $P = 18.067$ kW, $m_{res} = 0.4134$ kg, $m_{am} = 2$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg, $C_{res} = 6.56$, $x_{res} = 0.0522$, and $x_{am} = 0$. The masses are calculated from the flow from the corresponding valve position over one minute. All order states variables, and inputs are set to be zero. These WPs yielded the following TF:

$$H_{C,5.2}(s) = \frac{987.5 \cdot s^3 + 6485 \cdot s^2 + 73.28 \cdot s - 0.03001}{s^4 + 14.42 \cdot s^3 + 51.65 \cdot s^2 + 0.614 \cdot s} \tag{7.22}$$

6_Form:
In this step, there are two phases present, one during the addition of formaldehyde, and one after this addition is completed. The TFs, for the two phases will be named $H_{C,6.1}(s)$, and $H_{C,6.2}(s)$. Firstly, the TF for the first phase was calculated. The WPs used in the Matlab script were the following: $T_{reac} = 70$°C, $m_{res} = 2.488$ kg, $m_{am} = 2$ kg, $m_{form} = 0$ kg, $m_{prep} = 548.71$ kg, $T_h = 91.04$, $P = 42.096$ kW, , $C_{res} = 6.56$, $x_{res} = 0.0525$, and $x_{form} = 1$. The masses are calculated from the flow from the corresponding valve position over six minute. All order states variables, and inputs are set to be zero. These WPs yielded the following TF:

$$H_{C,6.1}(s) = \frac{983.9 \cdot s^3 + 6477 \cdot s^2 + 175.3 \cdot s - 0.1044}{s^4 + 14.42 \cdot s^3 + 51.79 \cdot s^2 + 1.428 \cdot s} \tag{7.23}$$

Secondly, the TF for the second phase was calculated. The WPs used in the Matlab script were the following: $T_{reac} = 70$°C, $m_{res} = 2.905$ kg, $m_{am} = 2$ kg, $m_{form} = 6$ kg, $m_{prep} = 548.71$ kg, $T_h = 79.15$, $P = 18.307$ kW, $C_{res} = 6.56$, $x_{res} = 0.0529$, and $x_{form} = 0$. The masses are calculated from the flow from the corresponding valve position over seven minute. All order states variables, and inputs are set to be zero. These WPs yielded the following TF:

$$H_{C,6.2}(s) = \frac{974.3 \cdot s^3 + 6398 \cdot s^2 + 73.32 \cdot s - 0.03015}{s^4 + 14.42 \cdot s^3 + 51.65 \cdot s^2 + 0.6222 \cdot s} \tag{7.24}$$

All of the four TFs are similar to each other, therefor only one of them will be used as a basis for the tuning of the PID controller. More specifically, the TF, $H_{C,5.1}(s)$, as this has the middle value of the parameter that differs the most.

7.2.2 PID tuning

As with the tuning for the temperature regulation, the Skogestad method is preferred. An obstacle to overcome, is that the method is most commonly used for lower order systems. But a higher order system can be modified to suit the Skogestad method. This is done by using model reduction techniques. The model reduction aims to lower the systems order to match one of the equations described in Figure 7.1.1. As mentioned, the basis for the PID tuning was the TF $H_{C,5.1}(s)$. To reduce the model order, the technique, Skogestad‘s Half Rule, was used, the rule is described in [21]. For the Skogestad half rule to be implemented the original model needs to be in a specific form, the form shown in Equation 7.25:

$$\frac{\prod_j \Big(-T_{j0}^{inv} + 1 \Big)}{\prod_i \tau_{i0} s + 1} \cdot e^{-\theta_0 s} \tag{7.25}$$

where $-T_{j0}^{inv}$ are the negative time constants, j, in the numerator in the original equation form, and τ_{i0} are the time constants, i, of the lags in the denominator.

Firstly, the integrator will be factored out, and neglected from this point on, it will be introduced again when the model reduction is completed. The TF that is left is a division of two 3rd degree polynomial terms. Both of these terms will be factored to their corresponding first order terms, these two adjustments leads to the following equation:

$$g_0 = \frac{-0.0655 \cdot (-1922.2 \cdot s + 1) \cdot (52.15 \cdot s + 1) \cdot (0.153 \cdot s + 1)}{(51.93 \cdot s + 1) \cdot (0.152 \cdot s + 1) \cdot (0.128 \cdot s + 1)} \tag{7.26}$$

As seen this does not equal to the wanted model form shown in Equation 7.25. There are two positive time constant in the numerator. In [21], an approximation rule to remove these two positive numerator time constants is given. The equations for the approximations are shown in Figure 7.27.

$$\frac{T_0 \cdot s+1}{\tau_0 \cdot s+1} \approx \begin{cases} T_0/\tau_0 \quad for\ T_0 \geq \tau_0 \geq \theta \quad (Rule\ T1) \\ T_0/\theta \quad for\ T_0 \geq \theta \geq \tau_0 \quad (Rule\ T1a) \\ 1 \quad for\ \theta \geq T_0 \geq \tau_0 \quad (Rule\ T1b) \\ T_0/\tau_0 \quad for\ \tau_0 \geq T_0 \geq 5\theta \quad (Rule\ T2) \\ \frac{\tilde{\tau}_0/\tau_0}{(\tilde{\tau}_0-\tau_0)\cdot s+1} \quad for\ \tilde{\tau}_0 \overset{def}{=} min(\tau_0, 5\theta) \geq T_0 \quad (Rule\ T3) \end{cases} \tag{7.27}$$

To be able to decide which of the five approximation rules to use from Figure 7.27, the delay, θ, needs to be estimated. The equation for the delay for the reduced first order model is the following:

$$\theta = \theta_0 + \frac{\tau_{20}}{2} + \sum_{i\geq 3} \tau_{i0} + \sum_j T_{j0}^{inv} + \frac{h}{2} \tag{7.28}$$

where h is the sampling frequency of the PID. As T_{j0}^{inv} is much bigger than all the other time constants, it is possible to assume that θs value will be close to the time constant of T_{j0}^{inv}, therefore, a value of 1922 is used as an estimation for θ.

With θ estimated, it can be decided which rule that should be used. In [21], it is purposed to start the approximation by starting with the largest numerator term, T_0, against the neighboring denominator term, τ_0. Since there is no τ_0 larger than $52.15s + 1$, the closest smaller τ_0 is chosen. This is the denominator term, $51.93s + 1$. In this circumstance rule T1, T1a or T1b, can be used. Since, θ is greater than both T_0 and τ_0, rule T1b will be used. This rule approximates $\frac{T_0 s+1}{\tau_0 s+1}$ to 1, yielding the following equation:

$$g_0 = \frac{-0.0655 \cdot (-1922.2 \cdot s+1) \cdot (0.153 \cdot s+1)}{(0.152 \cdot s+1) \cdot (0.128 \cdot s+1)} \tag{7.29}$$

The last positive numerator time constant can be approximated using the same rule, as this T_0 is also greater than its neighboring τ_0. Resulting in the following equation:

$$g_0(s) = \frac{-0.0655 \cdot (-1922.2 \cdot s+1)}{(0.128 \cdot s+1)} \tag{7.30}$$

This is now the form of the original equation that is wanted, and shown in Equation 7.25. The Skogestad Half Rule can now be implemented to create the first order reduced model. The calculation of the delay and the time constant for the reduced first order model are the following:

$$\tau_1 = \tau_{10} + \frac{\tau_{20}}{2} \tag{7.31}$$

where τ_{10} is the biggest τ_0 from the original model approximated to be in the form shown in Equation 7.25, and τ_{20} is the second biggest. Based on this the value for the time constant becomes:

$$\tau_1 = 0.128 + \frac{0}{2} = 0.128 \tag{7.32}$$

The delay is calculated using the Equation 7.28. The value for the delay becomes the following:

$$\theta = 0 + 0 + 0 + 1922 + 0 = 1922 \tag{7.33}$$

The sampling rate is ignored as the inverse response numerator time constant is so big. With all the values calculated the reduced first order model of $H_{C,5.1}$ becomes the following:

$$g(s) = \frac{-0.0655}{0.128 \cdot s+1} \cdot e^{-1922.2 \cdot s+1} \tag{7.34}$$

By multiplying in the integrator term, that was factored out in previous steps, the following equations becomes the result, which is possible to implement the Skogestad method for tuning on.

$$H_{Reduced}(s) = \frac{-0.0655}{s \cdot (0.128 \cdot s + 1)} \cdot e^{-1922.2 \cdot s + 1} \tag{7.35}$$

This equation is equal to the *integr + timeconst + del.* part of the Skogestad method of tuning. A time constant, T_C, of 2 seconds is chosen. Which gives the following tuning parameters:

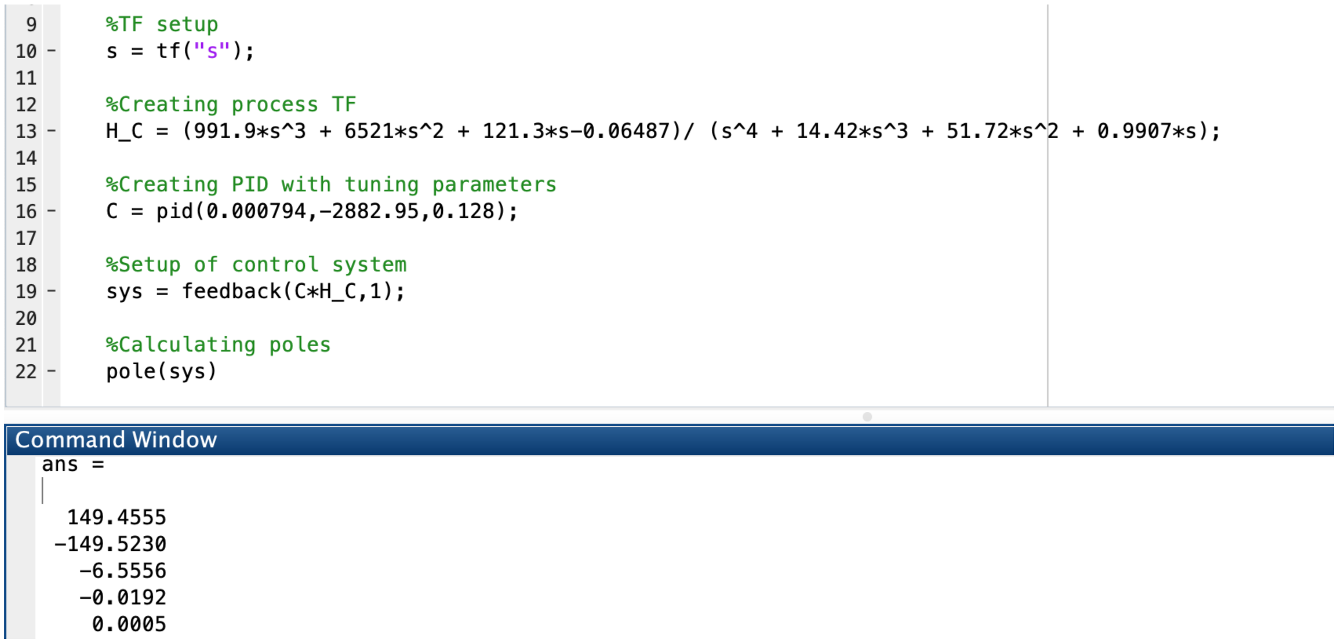
$$K_p = \frac{1}{K(T_C + \tau)} = \frac{1}{-0.0655 \cdot (\frac{2}{60} - 1922)} = 7.94 \cdot 10^{-4} \tag{7.36}$$

$$T_i = c(T_C + \tau) = 1.5 \cdot (\frac{2}{60} - 1922) = -2883.25 \tag{7.37}$$

$$T_d = T = 0.128 \tag{7.38}$$

To check if the tuning is stable, the systems poles are checked by using the $pole(...)$ command in Matlab. A pole is the values for s in the denominator, when the denominator is zero. If the pole is positive that means that the system is unstable. Therefore, it is wanted to have only negative poles. By multiplying the PID controllers TF, and the $H_{C,5.1}(s)$ TF, the TF representing the system is derived. The control system was made using the $feedback(...)$ command on the control systems TF. By using the $pole(...)$ command on this feedback system, the result became what seen in Figure 7.2.1. The use of a PD controller, was also tested to seen if the absence of another integrator would be a work around, but it also yielded unstable control.

The control system has two positive poles, meaning the system is unstable. This could possibly be due to problems generated by the integrator part that was factored out from the original model, and therefore, the reduction of the model does not deliver correct results, by approximating the negative time constant in the numerator as a delay. Which is what happens when Skogestads half rule is implemented.



```
%TF setup
s = tf("s");

%Creating process TF
H_C = (991.9*s^3 + 6521*s^2 + 121.3*s-0.06487)/ (s^4 + 14.42*s^3 + 51.72*s^2 + 0.9907*s);

%Creating PID with tuning parameters
C = pid(0.000794,-2882.95,0.128);

%Setup of control system
sys = feedback(C*H_C,1);

%Calculating poles
pole(sys)
```

Command Window

```
ans =

  149.4555
 -149.5230
   -6.5556
   -0.0192
    0.0005
```

Figure 7.2.1: Calculation of poles for control system

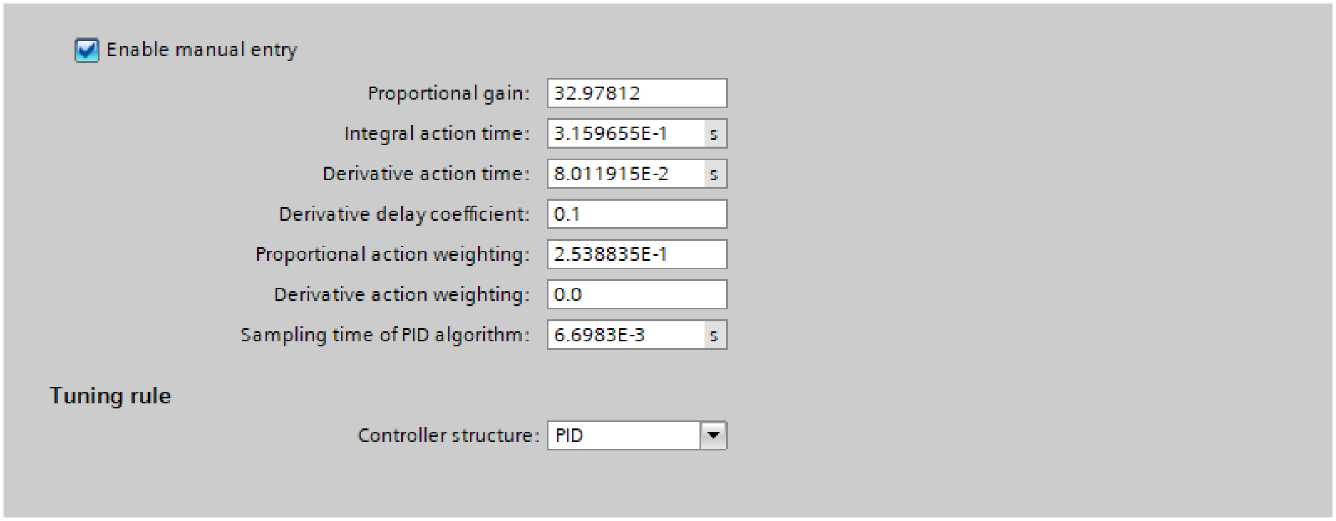
Due to the tuning when using Skogestads method in combination with Skogestads half rule, becoming unstable, another tuning method must be used. Therefore, the PID-tuning method integrated in the

TIA Portal was used. The PID controller, *PID_Compact*, has two tuning methods, pretuning, and fine tuning. The functionality and implementation of the integrated PID-Controller, is described in [15]. It is here the functionality of the two tuning methods are described. The two methods are, for the best control, used in combination. By firstly, completing the pretuning, and then the fine tuning. The pretuning works by applying a step on the process and thereafter searching for the infliction point of the response, this point is where the rate of change is at its maximum. The calculation of the parameters bases itself on this in combination with the dead time that is apparent when the step is applied. After the step is completed, the controller uses these parameters to tune itself. By completing the fine tuning in combination with the pretuning, the control system gains better disturbance compensation. The fine tuning generates a small oscillation of the process value, and then calculates the PID parameters based of of the behaviour of the system. The PIDs algorithm, that calculates the output response based of the input, operates according to the following equation:

$$y = K_p \cdot \left[(b \cdot w - x) + \frac{1}{T_i \cdot s}(w - x) + \frac{T_d \cdot s}{a \cdot T_d \cdot s + 1}(c \cdot w - x)\right] \tag{7.39}$$

where y is the output of the PID, K_p is the proportional gain, s is the Laplace operator, b is the proportional action weighting factor, w is the setpoint, x is the process value, T_i is the integral action time, a is the derivative delay coefficient, T_d is the derivative action time, and c is the derivative action weighting.

By implementing both of the tuning tools, using the procedure described in section 4.1.2 in [15], the resulting PID parameters shown in Figure 7.2.2 were derived.


✓ Enable manual entry

Proportional gain:	32.97812	
Integral action time:	3.159655E-1	s
Derivative action time:	8.011915E-2	s
Derivative delay coefficient:	0.1	
Proportional action weighting:	2.538835E-1	
Derivative action weighting:	0.0	
Sampling time of PID algorithm:	6.6983E-3	s

Tuning rule

Controller structure: PID

Figure 7.2.2: PID parameters using the TIA portal tuning tools

7.2.3 Regulation Analysis

This section describes the step tests of the PID controller, using the PID parameters derived in the previous section. The test was completed by starting the setpoint at 10 $\frac{g}{m^3}$, then changing it to 6.56, the basis value of the process, and at the end setting it to 7.56. This gives the step response both with an increase and an decrease of the concentration, to test the control limitations in both directions. The completed test can be seen in Figure 7.2.3.

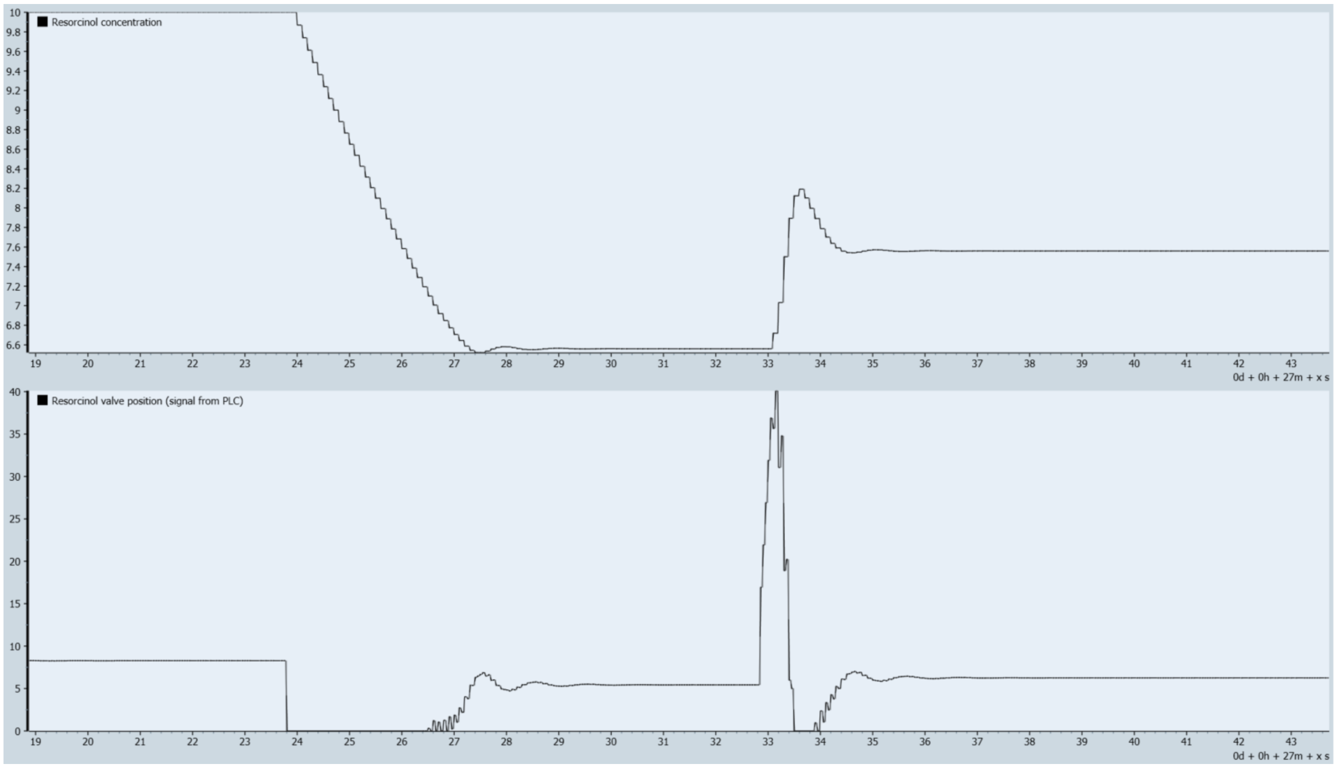


Figure 7.2.3: Step response of regulated resorcinol concentration

The y axis of the *Resorcinol Concentration* chart is the concentration in $\frac{g}{m^3}$, while the x axis is the time in seconds, which is equal for both of the charts in Figure 7.2.3. The y axis for the chart *Resorcinol Valve Position* is the valves position as a percentage from 0-100 %. At approximately 24 seconds, the concentration setpoint was changed from 10 to 6.56. The behavior can seem to be similar to linear. This is due to the fact that the valve position is saturated at 0, meaning it cant apply a lesser value on the valve. The square shape of the signal is due to the sampling interval of the PLC. The regulation has a small overshoot, suggesting the system is under-damped, but it is acceptable. At 33 seconds the setpoint was changed to 7.56. There, it can be seen that the response is quicker due to the valve not reaching a saturation value, the time constant of the control system can be estimated to be around a quarter of a second, which is a good value for this quick process. For this step the overshoot is bigger, this is because the tuning method was implemented using 6.56 as a setpoint, meaning it does not behave as well at other setpoints. Another reason could be that the sampling rate is to slow. It is also possible to assume that the valve will go into saturation at higher steps because the valve is opened at 40 % on at 1 $\frac{g}{m^3}$ step. Meaning it would need to apply a greater output to reach a higher setpoint, which could lead to saturation.

By increasing the volume $V(t)$, it acts as a disturbance. This can be seen mathematically in Equation 3.49. This disturbance was introduced at approximately 44 seconds, with the H2O valve being opened, as this has the highest flow, and would give the biggest disturbance. It is important to notice that this big of a disturbance is unrealistic in the process, as the biggest disturbance introduced in the process is in reality the flow of formaldehyde, which is about 60 times smaller than the H2O flow. H2O is only used to test the compensation capabilities of the PID controller. As seen in Figure 7.2.4, the controller start compensating for the increased volume by opening the valve more, as seen in the *Resorcinol Valve Position* chart, consequentially the concentration drops minimally, for reference, the peak seen on the left in the *Resorcinol Concentartion* chart, is approximately 0.001 higher than 7.56, the axis is not scaled because SIMIT struggles with smaller values due to rounding.

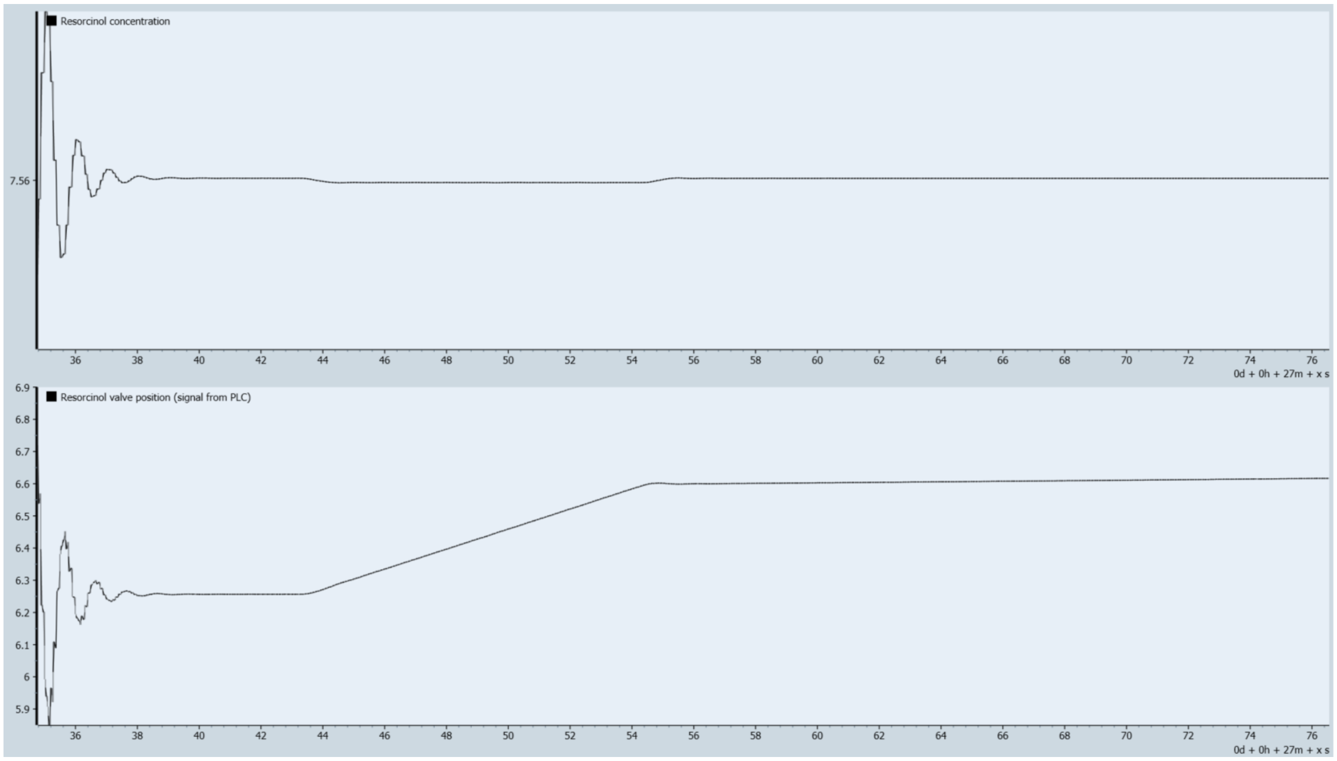


Figure 7.2.4: Step response of regulated resorcinol concentration with applied disturbance

In conclusion, it is possible to say that the controlled system, with the PID controller being tuned with the builtin TIA Portal tuning tool was robust against disturbances and has good regulation performance around the tuning WP. It is also seen that it gives higher overshoots when deviating from the tuning WP value and struggles with higher steps due to saturation issues. For this project the solution is deemed viable as it performance well around the tuned WP. If the desired setpoint is changed due to a more optimal concentration. The builtin tuning tool in the TIA Portal is deemed suitable for regulation at this new WP, as long as it is retuned to against this new setpoint.

Chapter 8

Testing and Simulations

This chapter will describe the tests that have been done to assess if the simulation model and control system function properly, both separately and together. The first section shows the tests of the SIMIT model independently, including basic tests of logic throughout the SIMIT project, as well as evaluating the behavior of some dynamic values depending on inputs. The second section is specifically about the PLC control. In this section, every FB corresponding to the different steps of the process will be tested concerning the internal functionality and the triggers for changing to the next stage. Supporting FBs will also be tested. The third section contains the tests checking if the visual elements on the HMI screen are correctly connected to the tags utilized by the PLC. The fourth and last section concerns the system test as a whole. In this section, SIMIT is connected to the PLC. This test is to evaluate if the PLC performs the correct actions based on the signals received from SIMIT, and also if the simulation enters the different steps of the process based on the PLC control signals.

8.1 SIMIT Testing

Testing all the critical aspects of the simulation is a way to verify that the mathematical modeling has been correctly implemented in SIMIT. In order to test the project in SIMIT separately from the PLC, PLC input signals are replaced with switches and sliders that can be operated in SIMIT while the simulation is running. There are several different input control signals to choose from, including, but not limited to, pushbuttons, switches, digital inputs, and sliders. For this project, only switches and sliders were needed. Switches were implemented to control the valves, the sonicator, and the agitator. The switch defaults to a binary $FALSE$, and when it is clicked, it will be set to $TRUE$ until it is clicked again, resetting it to $FALSE$. This behavior is the same as the control signals from the PLC; the only difference is that the switches are controlled inside SIMIT by mouse clicks, as is wanted for the internal SIMIT testing. There are three main customizations for the sliders: initial value, end value, and increment. If a power signal from 0 W - 200 W is wanted, the initial value would be 0, the end value 200, and the increment would depend on which value is needed for the step increments in this interval.

A chart called *Overview* has been developed to centralize the monitoring and control of the process. In this chart, the switches and sliders are connected to connectors used in their respective charts, and the important variables of the process are monitored. *Overview* also includes visual representations of the process components to indicate which of them are active in a given moment, based on input signals. However, the illustrated components do not correctly represent their real counterparts in the least.

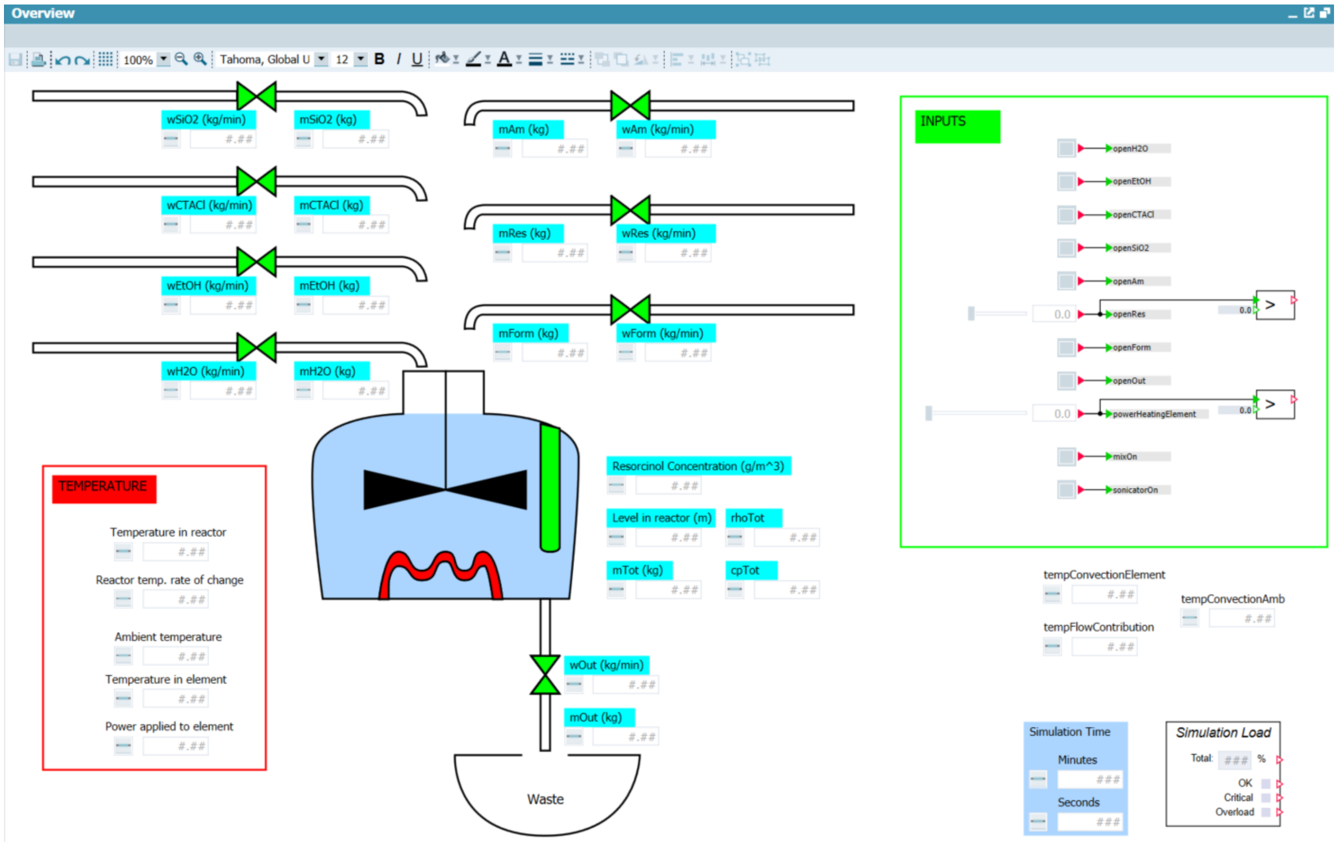
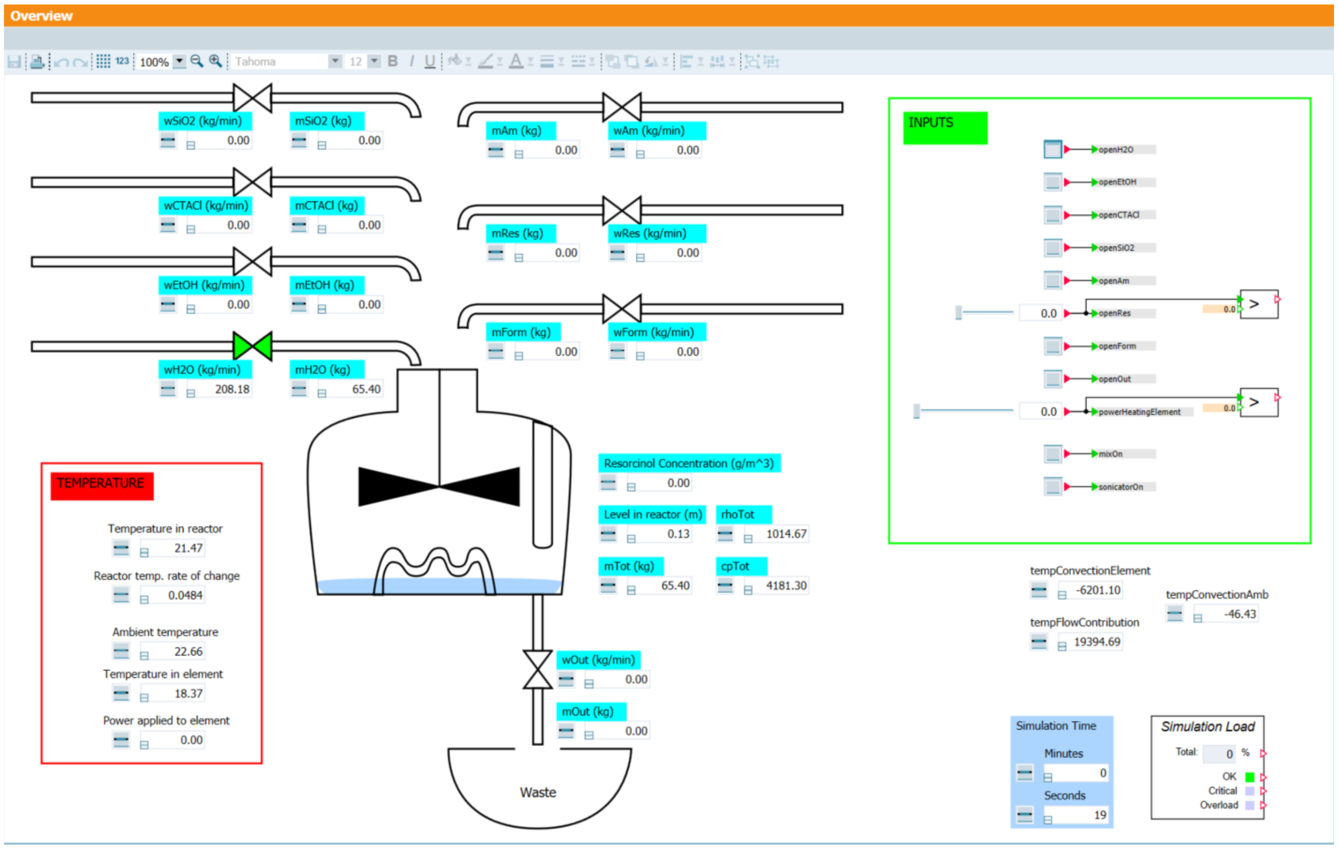


Figure 8.1.1: Overview of the internal monitoring and control chart

Figure 8.1.1 shows the *Overview* chart. The left side contains the illustrated process components and the monitored values, while the right side has all the inputs and *SimulationLoad*, which monitors the processing power required as a percentage of the PC's CPU capabilities. The seven green components above the illustrated tank reactor represent the seven inlet valves. Each of the cyan text boxes beneath each valve describes which value is monitored, and the box containing #.## below each text box shows the current value, which will only be actual values when the simulation is running. The valves themselves are not always green during the simulation. They are animated to turn green once the valve is opened by the input signal to see which valves are open at any given time easily. The three components inside the reactor are the sonicator (green), agitator (black), and heating element (red). These are also animated to indicate being active based on the input signals. The two cyan text boxes to the right of the reactor are for monitoring the solution's level and the total mass inside the reactor. Below the reactor, is the outlet valve and an illustrated waste tank. The red box on the left outlines the main monitored variables related to the temperature in the reactor.

8.1.1 Flow Dynamics

The first thing to test is the flow through the valves based on the valve opening and the subsequent mass accumulation. For this test, the largest potential flow rate of H_2O will be calculated mathematically and compared to the monitored flow rate in SIMIT while the valve is fully open. To verify the accumulated mass, the valve will be opened upon initiation of the simulation, and the flow rate will be multiplied by the elapsed time.



wSiO2 (kg/min) 0.00
mSiO2 (kg) 0.00
wCTACl (kg/min) 0.00
mCTACl (kg) 0.00
wEtOH (kg/min) 0.00
mEtOH (kg) 0.00
wH2O (kg/min) 208.18
mH2O (kg) 65.40
mAm (kg) 0.00
wAm (kg/min) 0.00
mRes (kg) 0.00
wRes (kg/min) 0.00
mForm (kg) 0.00
wForm (kg/min) 0.00
Resorcinol Concentration (g/m^3) 0.00
Level in reactor (m) 0.13
rhoTot 1014.67
mTot (kg) 65.40
cpTot 4181.30
wOut (kg/min) 0.00
mOut (kg) 0.00

Temperature in reactor 21.47
Reactor temp. rate of change 0.0484
Ambient temperature 22.66
Temperature in element 18.37
Power applied to element 0.00

openRes 0.0 0.0
powerHeatingElement 0.0 0.0

tempConvectionElement -6201.10
tempConvectionAmb -46.43
tempFlowContribution 19394.69

Simulation Time
Minutes 0
Seconds 19

Simulation Load
Total: 0 %
OK
Critical
Overload

Figure 8.1.2: Flow rate test, a still picture of the simulation in progress.

Figure 8.1.2 shows the *Overview* chart while a simulation is in progress, with H_2O flowing into the reactor as indicated by the green valve. As seen in the figure, the flow rate $wH2O = 208.18\ kg/min$. The flow rate is defined in Equation 3.4, which states that:

$$q_i(t) = \frac{K_v \cdot x_i(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i(t)}{100000 \cdot SG_i}} \qquad \left[\frac{m^3}{min}\right] \tag{8.1}$$

where $K_{v,H_2O} = 40$, $x_{H_2O} = 1$, $\rho_{H_2O} = 998$, $g = 9.81$, $h_{H_2O} = 1$, and $SG_{H_2O} = 1$. Inserting these values into Equation 8.1, the flow rate becomes:

$$q_{H_2O}(t) = \frac{40 \cdot 1}{60} \cdot \sqrt{\frac{998 \cdot 9.81 \cdot 1}{100000 \cdot 1}} = 0.208597 \qquad \left[\frac{m^3}{min}\right] \tag{8.2}$$

The mass flow rate can be obtained with the following relation:

$$w_{H_2O}(t) = \rho_{H_2O} \cdot q_{H_2O}(t) = 998 \cdot 0.208597 = 208.18 \qquad \left[\frac{kg}{min}\right] \tag{8.3}$$

This value matches the value of $wH2O$ shown in Figure 8.1.2, meaning that the flow rate equation has been implemented correctly in SIMIT. To evaluate the precision of the accumulated mass, the flow rate is multiplied by the elapsed time. The match will not be exact due to two factors: The valve is opened by a mouse click, which will not happen at the exact time of simulation initiation when the elapsed time starts counting, and the time is only displayed in minutes and seconds, meaning that the currently displayed time of 19 seconds could in reality be anywhere between 19 and 20 seconds. Another factor that has to be taken into account is the valve's opening time of one second. Therefore, one of the elapsed seconds has to be multiplied with the average flow rate during the first second, which is half of the maximum flow rate because the valve is linear. Recalling that the rate of change in mass equals the flow rate, the accumulated mass will then equal the integral of the flow rate:

$$\frac{dm(t)}{dt} = w(t) \quad \Leftrightarrow \quad m(t) = \int_0^{\tau} w(t)dt \tag{8.4}$$

Since the flow rate during the first second is different from the maximum flow rate, the integral will be split into two parts. The flow rate is defined in kg/min, the whole expression should therefore be divided by 60 when integrating with respect to seconds. Applying these modifications to $m(t)$ in Equation 8.4, the mass of H_2O becomes:

$$\begin{aligned} m_{H_2O}(t) &= \frac{1}{60}\left(\int_0^1 \overline{w_{H_2O}}(t)dt + \int_1^{19} w_{H_2O,max}dt\right) \\ &= \frac{1}{60}\left(\frac{1}{2}\int_0^1 w_{H_2O,max}dt + \int_1^{19} w_{H_2O,max}dt\right) \\ &= \frac{1}{60}\left(\frac{1}{2}\Big[t\cdot w_{H_2O,max}\Big]\Big|_0^1 + \Big[t\cdot w_{H_2O,max}\Big]\Big|_1^{19}\right) \\ &= \frac{1}{60}\left(\frac{1}{2}\cdot w_{H_2O,max} + \big(19\cdot w_{H_2O,max} - 1\cdot w_{H_2O,max}\big)\right) \\ &= \frac{1}{60}\left(\frac{208.18}{2} + 18\cdot 208.18\right) = 64.19 \quad [kg] \end{aligned} \tag{8.5}$$

There is an error of 1.21 kg, which is reasonable because the flow rate is 3.47 kg/s and the margin of error on the displayed time elapsed is a maximum of one second. By doing the calculation in Equation 8.4 again, but substituting 19 seconds with 19.5 seconds, the result is $m_{H_2O} = 65.92$ kg. This shows that the mathematically modeled flow rate and accumulated mass are implemented correctly in SIMIT.

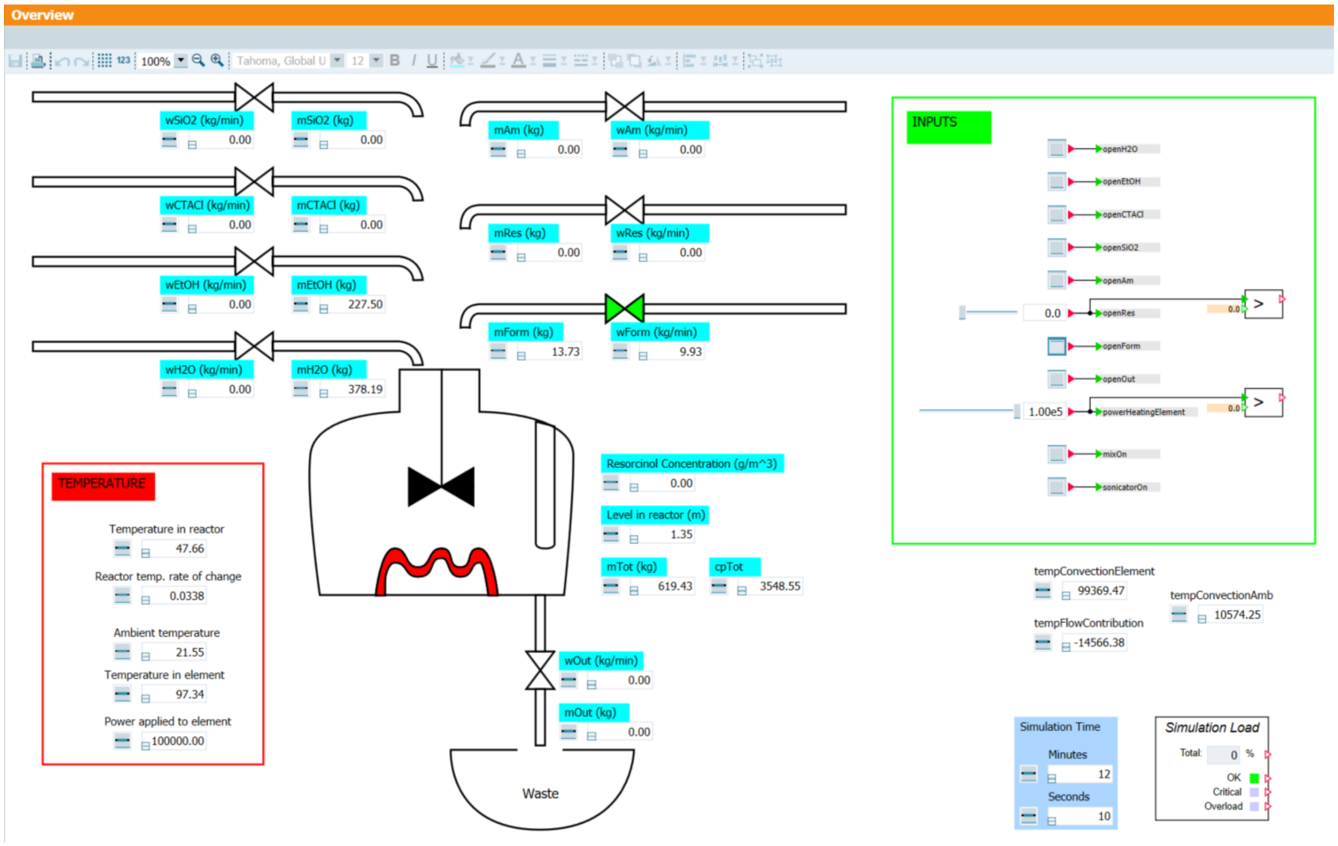
8.1.2 Temperature Dynamics

The temperature's rate of change in the reactor is tested while formaldehyde is flowing into the reactor and the heating element is turned on. This is to include every variable affecting the temperature, as seen in Equation 3.34, meaning that a single test will cover everything that can impact the modeled temperature. The compounds added into the reactor prior to turning the heating element on and adding formaldehyde is only to have a large mass inside the reactor, the amount of compounds does not represent the real process.

Figure 8.1.3 shows the *Overview* chart during a simulation. The single green valve indicates that this valve is currently open, and the same applies to the red heating element. $mTot$ is equal to 619.43 kg, which can be verified by summing the individual masses, where only the three closest valves above the reactor have been opened during the simulation. On the right side of $mTot$ is $cpTot$, which currently has the value 3548.55. This can be verified by calculating the weighted sum of the individual specific heat capacities in the reactor as such:

$$\begin{aligned} c_{p,tot} &= \frac{1}{m_{tot}}\sum_{i=1}^{7} c_{p,i}\cdot m_i = \frac{1}{m_{tot}}\cdot\Big(c_{p,H_2O}\cdot m_{H_2O} + c_{p,EtOH}\cdot m_{EtOH} + c_{p,form}\cdot m_{form}\Big) \\ &= \frac{1}{619.43}\cdot\Big(4181.3\cdot 378.19 + 2510\cdot 227.5 + 3328\cdot 13.73\Big) = 3548.495 \end{aligned} \tag{8.6}$$

The reason for the marginal difference between $c_{p,tot}$ and the value of $cpTot$ displayed in Figure 8.1.3 is that $c_{p,tot}$ was calculated using other values displayed in the SIMIT digital displays. These displays have been rounded to two decimals, making the calculated $c_{p,tot}$ less precise than $cpTot$ in SIMIT.



wSiO2 (kg/min) 0.00
mSiO2 (kg) 0.00
wCTACl (kg/min) 0.00
mCTACl (kg) 0.00
wEtOH (kg/min) 0.00
mEtOH (kg) 227.50
wH2O (kg/min) 0.00
mH2O (kg) 378.19
mAm (kg) 0.00
wAm (kg/min) 0.00
mRes (kg) 0.00
wRes (kg/min) 0.00
mForm (kg) 13.73
wForm (kg/min) 9.93

openH2O
openEtOH
openCTACl
openSiO2
openAm
0.0 openRes 0.0
openForm
openOut
1.00e5 powerHeatingElement 0.0
mixOn
sonicatorOn

Temperature in reactor 47.66
Reactor temp. rate of change 0.0338
Ambient temperature 21.55
Temperature in element 97.34
Power applied to element 100000.00

Resorcinol Concentration (g/m^3) 0.00
Level in reactor (m) 1.35
mTot (kg) 619.43
cpTot 3548.55
wOut (kg/min) 0.00
mOut (kg) 0.00

tempConvectionElement 99369.47
tempConvectionAmb 10574.25
tempFlowContribution -14566.38

Simulation Time
Minutes 12
Seconds 10

Simulation Load
Total: 0 %
OK 🟩
Critical
Overload

Figure 8.1.3: Temperature test, a still picture of the simulation in progress.

The temperature's rate of change in the reactor can be verified by inserting every known value into Equation 3.34. The only two values not directly shown in Figure 8.1.3 are the contact area between the solution and the reactor surface, A_{reac}, and the temperature of the inlet flow, $T_{i,in}$. Due to the fact that the inlet temperature has an introduced noise in SIMIT, the verification will not precisely match. The median value 22.5 $°C$ is used for this calculation. However, A_{reac} can be calculated using the level in the reactor and the radius with the following equation:

$$A_{reac} = 2\pi \cdot r \cdot h(t) = 2\pi \cdot 0.4 \cdot 1.35 = 3.39292 \tag{8.7}$$

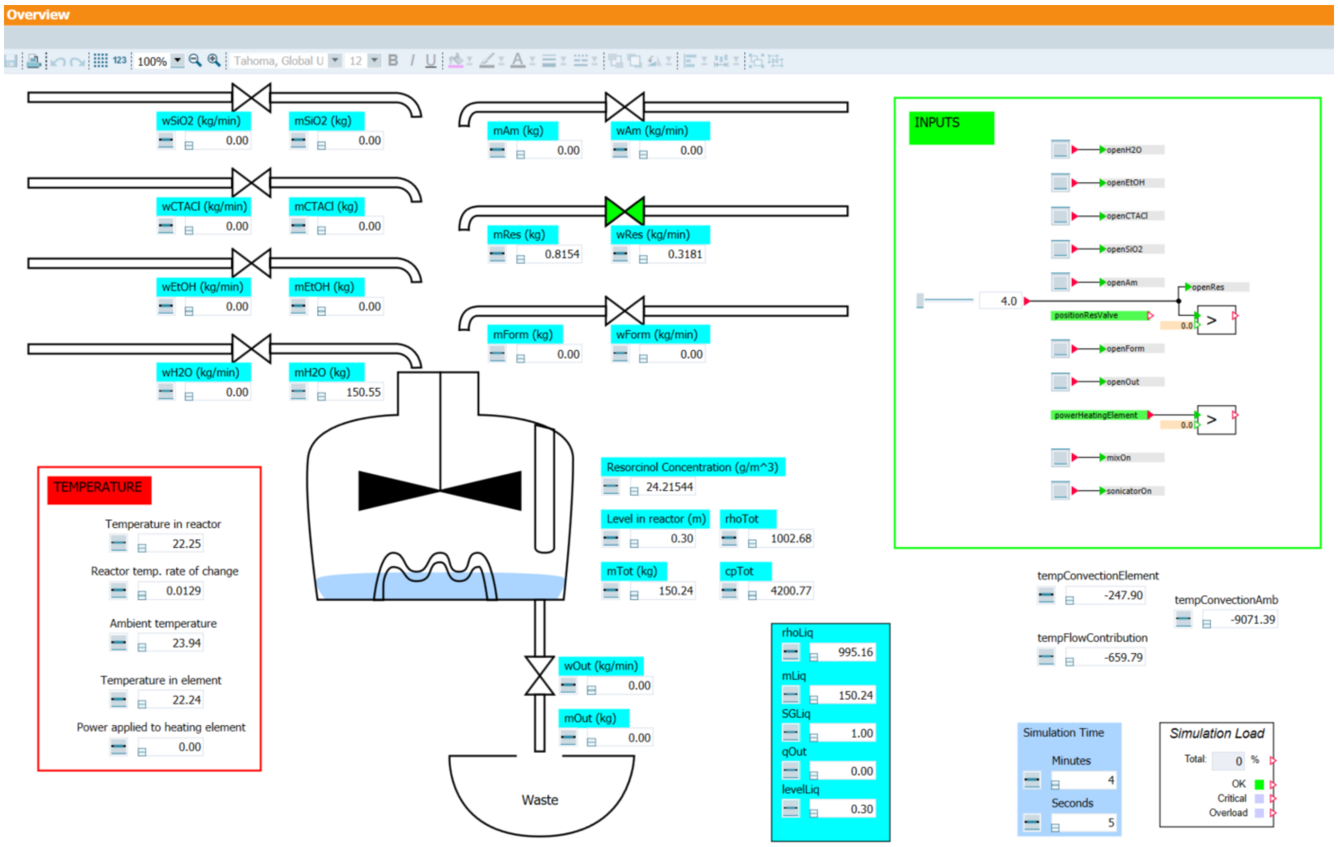
With A_{reac} known, the temperature's rate of change is calculated as follows:

$$\begin{aligned}\frac{dT_{reac}}{dt} =& \frac{1}{619.43 \cdot 3548.55}\Bigg(\frac{1}{60} \cdot 3328 \cdot 9.93 \cdot (22.5 - 47.66) \\ &-119.16 \cdot 3.39292 \cdot (47.66 - 21.55) + 1200 \cdot 1.667 \cdot (97.34 - 47.66)\Bigg) \\ =&0.03410\end{aligned} \tag{8.8}$$

There is a slight difference from *Reactor temp. rate of change* shown in Figure 8.1.3, the reason is the introduced noise for the inlet temperature. By replacing 22.5 in Equation 8.8 with 21.3, the result is an exact match with *Reactor temp. rate of change*, which implies that the temperature of the inflowing formaldehyde was 21.3 $°C$ at this exact moment in the simulation.

8.1.3 Concentration Dynamics

In order to evaluate the implementation of concentration dynamics, the reactor was filled with a compound aside from resorcinol before the resorcinol valve was opened to a set value. After a certain amount of time, another compound is added to test the effect an extra inlet flow has on the concentration. Figure 8.1.4 shows the concentration after about a minute of the resorcinol valve being open, meaning that the concentration has been able to settle.


Overview

wSiO2 (kg/min) 0.00
mSiO2 (kg) 0.00
wCTACl (kg/min) 0.00
mCTACl (kg) 0.00
wEtOH (kg/min) 0.00
mEtOH (kg) 0.00
wH2O (kg/min) 0.00
mH2O (kg) 150.55
mAm (kg) 0.00
wAm (kg/min) 0.00
mRes (kg) 0.8154
wRes (kg/min) 0.3181
mForm (kg) 0.00
wForm (kg/min) 0.00

TEMPERATURE
Temperature in reactor 22.25
Reactor temp. rate of change 0.0129
Ambient temperature 23.94
Temperature in element 22.24
Power applied to heating element 0.00

Resorcinol Concentration (g/m^3) 24.21544
Level in reactor (m) 0.30
rhoTot 1002.68
mTot (kg) 150.24
cpTot 4200.77
wOut (kg/min) 0.00
mOut (kg) 0.00
rhoLiq 995.16
mLiq 150.24
SGLiq 1.00
qOut 0.00
levelLiq 0.30

INPUTS
positionResValve 4.0

tempConvectionElement -247.90
tempConvectionAmb -9071.39
tempFlowContribution -659.79

Simulation Time
Minutes 4
Seconds 5

Simulation Load
Total: 0 %
OK 🟩
Critical
Overload

Figure 8.1.4: Concentration test, a still picture of the simulation in progress.

As seen in Figure 8.1.4, the resorcinol concentration is 24.21544 g/m^3. To verify that this value correlates to the mathematically modeled concentration, the known parameter values can be inserted in Equation 3.49, which states:

$$\frac{dC_{res}(t)}{dt} = \frac{1}{V(t)}\Big(C_{res,in} \cdot q_{res}(t) - C_{res}(t) \cdot \sum_{i=1}^{7} q_i(t)\Big) - A \cdot e^{\frac{-E_a}{R\cdot(Treac(t)+273.15)}} \cdot C_{res}(t) \qquad \left[\frac{g}{min \cdot m^3}\right] \tag{8.9}$$

Because the concentration has had time to settle after opening the resorcinol valve, the concentration's rate of change will be considered equal to 0. Resorcinol is the only compound added at this time, meaning that the sum of inlet flows are equal to the resorcinol flow rate. The equation can be rearranged to solve for $C_{res}(t)$:

$$0 = \frac{1}{V(t)}\Big(C_{res,in} \cdot q_{res}(t) - C_{res}(t) \cdot q_{res}(t)\Big) - A \cdot e^{\frac{-E_a}{R\cdot(Treac(t)+273.15)}} \cdot C_{res}(t)$$

$$A \cdot e^{\frac{-E_a}{R\cdot(Treac(t)+273.15)}} \cdot C_{res}(t) \cdot V(t) = C_{res,in} \cdot q_{res}(t) - C_{res}(t) \cdot q_{res}(t)$$

$$A \cdot e^{\frac{-E_a}{R\cdot(Treac(t)+273.15)}} \cdot C_{res}(t) \cdot V(t) = q_{res}(t)\Big(C_{res,in} - C_{res}(t)\Big)$$

$$\frac{A \cdot e^{\frac{-E_a}{R\cdot(T_{reac}(t)+273.15)}} \cdot V(t)}{q_{res}(t)} = \frac{C_{res,in} - C_{res}(t)}{C_{res}(t)}$$

$$\frac{A \cdot e^{\frac{-E_a}{R\cdot(T_{reac}(t)+273.15)}} \cdot V(t)}{q_{res}(t)} = \frac{C_{res,in}}{C_{res}(t)} - 1$$

$$C_{res}(t) = \frac{C_{res,in}}{\frac{A\cdot e^{\frac{-E_a}{R\cdot(T_{reac}(t)+273.15)}}\cdot V(t)}{q_{res}(t)} + 1}$$

$$C_{res}(t) = \frac{C_{res,in} \cdot q_{res}(t)}{A \cdot e^{\frac{-E_a}{R\cdot(T_{reac}(t)+273.15)}} \cdot V(t) + q_{res}(t)} \tag{8.10}$$

The other variables are: $V(t) = \frac{m_{tot}(t)}{\rho_{tot}(t)} = \frac{150.24}{1002.68}$, $C_{res,in} = 70850$, $q_{res}(t) = \frac{w_{res}(t)}{\rho_{res}} = \frac{0.3181}{953.13}$, $A = 25$, $E_a = 30000$, $R = 75.5$, and $T_{reac}(t) = 22.25$. Inserting these values into Equation 8.10, the resorcinol concentration is the following:

$$C_{res}(t) = \frac{70850 \cdot \frac{0.3181}{953.13}}{25 \cdot e^{\frac{-30000}{75.5\cdot(22.25+273.15)}} \cdot \frac{150.24}{1002.68} + \frac{0.3181}{953.13}} = 24.22 \tag{8.11}$$

The value is approximately equal to the concentration displayed in Figure 8.1.4, and there are two main reasons why they are not exactly equal. Firstly, the values used in 8.11 are taken from the rounded numbers in the SIMIT displays. Secondly, even though the concentration was able to settle, it did not fully reach the equilibrium point. This means that the concentration's rate of change was only close to 0, not exactly equal. Taking these factors into account, the mathematically calculated concentration was close enough to consider it equal to SIMIT's calculation.

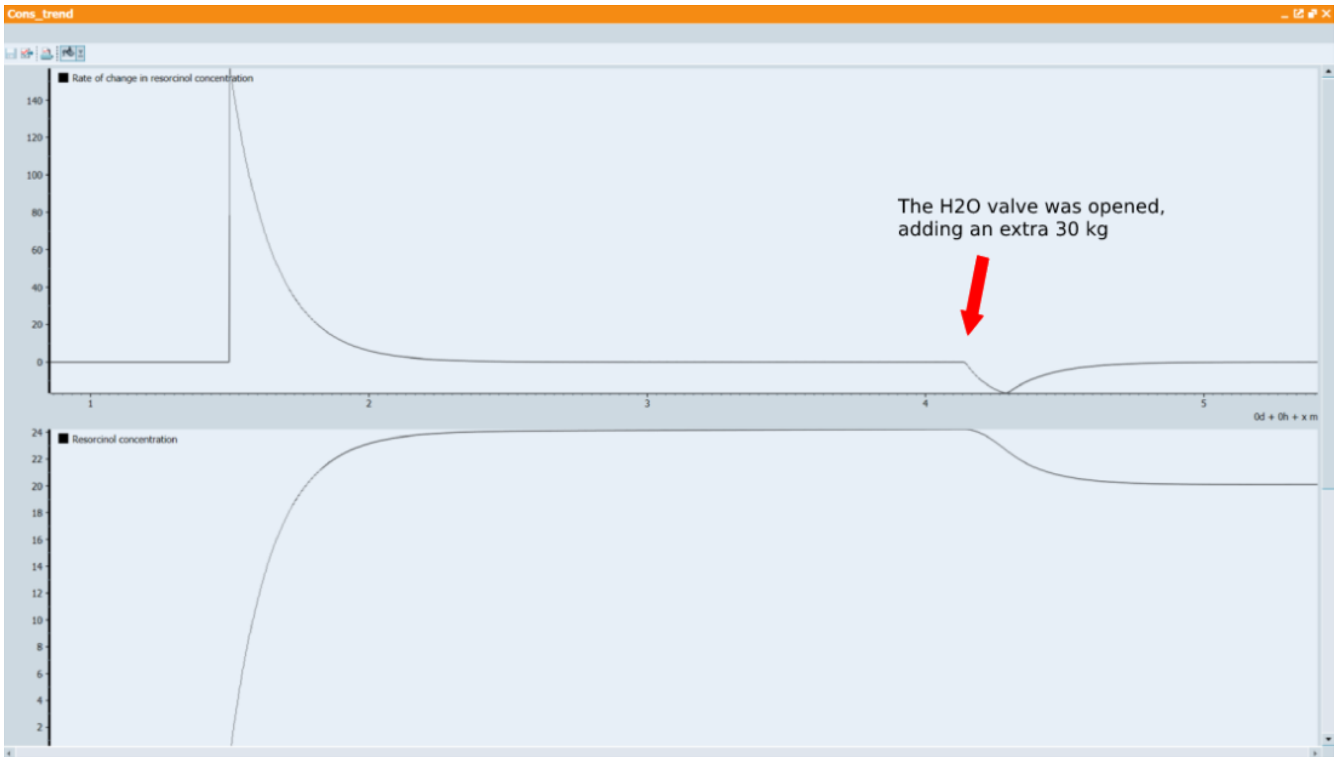


Figure 8.1.5: The concentration dynamics, numbers on the x-axis represent minutes.

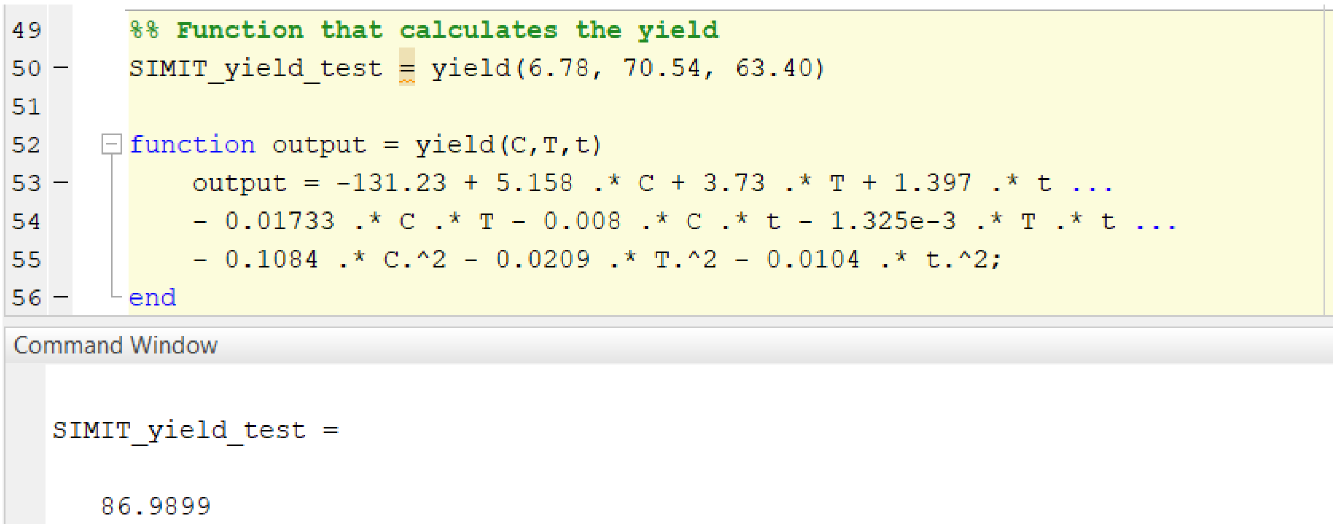
Figure 8.1.5 shows the development of the concentration behavior over time, where the top part is the concentration's rate of change, and the bottom part is the actual concentration. There was no concentration until about 1.5 minutes into the simulation, which is when the resorcinol valve was opened. The result is a spike in the rate of change, which then decreases exponentially, allowing the concentration to approach an equilibrium point as seen in the bottom part of the figure. After a while, the H_2O valve was opened again while the resorcinol valve was still open. The consequence is that the volume increases, making the concentration unable to maintain its current value without increasing the resorcinol valve opening. This behavior is pointed out by the red arrow in the figure.

8.1.4 Yield

There are four requirements for properly testing the yield. Firstly, there needs to be some volume in the reactor for the resorcinol to reach a reasonable concentration level. Secondly, the heating element will need to be turned on to keep the solution close to 70 °C. Thirdly, the resorcinol valve needs to be kept open in order to keep the concentration level above 0 during the coating. Lastly, the formaldehyde valve needs to be opened to start the calculation of the coating time, which also starts the calculation of the average concentration and temperature. After running the simulation for this test, the result was verified by calculating the yield in Matlab with the resulting parameter values from SIMIT.

The first requirement was achieved by filling the reactor with H_2O. The heating element was supplied with 100 000 W to increase the temperature from room temperature, and after a while the supplied power was reduced to an amount sufficient in keeping the solution close to 70 °C, thus fulfilling the second requirement. Next, the resorcinol valve was opened, and was kept open for the duration of this test, fulfilling the third requirement. The last requirement was met when the formaldehyde valve was opened, which then proceeded to start the calculations of the variables in the yield equation.

The time variable used in the yield calculation is supposed to be in minutes. For testing the yield however, minutes was swapped with seconds. This changes nothing regarding yield behavior, if the number of seconds matches the number of minutes, which makes testing a significantly less time consuming procedure. The coating duration lasts until resorcinol has stopped flowing into the reactor. Therefore, the resorcinol valve was closed after approximately 60 seconds.

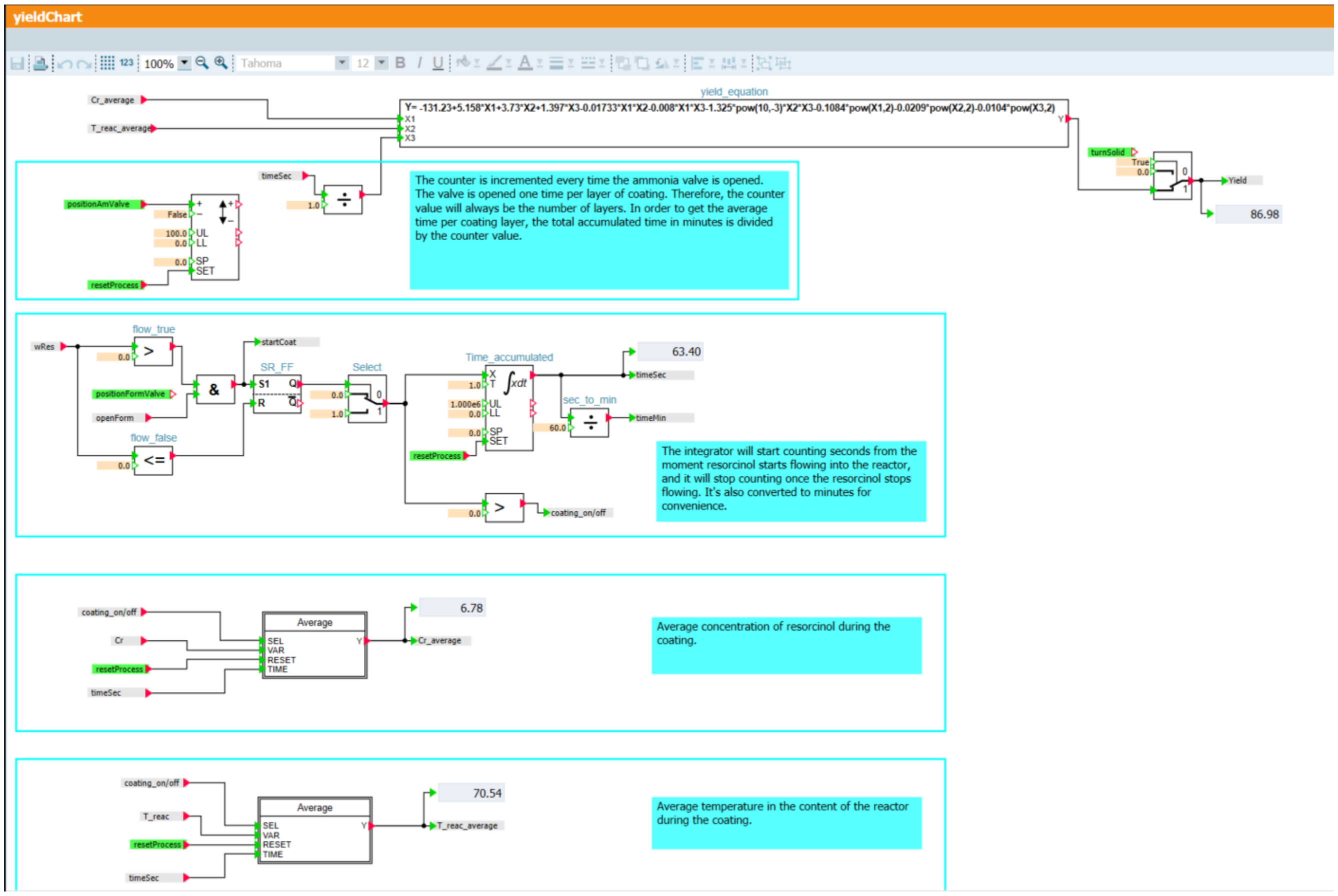


```
%% Function that calculates the yield
SIMIT_yield_test = yield(6.78, 70.54, 63.40)

function output = yield(C,T,t)
    output = -131.23 + 5.158 .* C + 3.73 .* T + 1.397 .* t ...
    - 0.01733 .* C .* T - 0.008 .* C .* t - 1.325e-3 .* T .* t ...
    - 0.1084 .* C.^2 - 0.0209 .* T.^2 - 0.0104 .* t.^2;
end
```

```
SIMIT_yield_test =

   86.9899
```

Figure 8.1.6: The yield calculated in Matlab based on the values from SIMIT.

After running a simulation with the four requirements, the coating step was executed as seen in Figure 8.1.7 with the following resulting variables: $\overline{C}_{res} = 6.78$, $\overline{T}_{reac} = 70.54$, $t = 63.40$, and $Yield = 86.98$. When inserting the three parameters in the Matlab function calculating the yield, the result is $SIMIT\_yield\_test = 86.9899$ as shown in Figure 8.1.6. If rounded to two decimals like the displayed yield in SIMIT, the Matlab answer is off by 0.01. The reason for this, like with the other tests, is that the manually calculated result uses imprecise values displayed in SIMIT. Thus, the yield calculation implemented in SIMIT can be considered correct.



Y= -131.23+5.158*X1+3.73*X2+1.397*X3-0.01733*X1*X2-0.008*X1*X3-1.325*pow(10,-3)*X2*X3-0.1084*pow(X1,2)-0.0209*pow(X2,2)-0.0104*pow(X3,2)

The counter is incremented every time the ammonia valve is opened. The valve is opened one time per layer of coating. Therefore, the counter value will always be the number of layers. In order to get the average time per coating layer, the total accumulated time in minutes is divided by the counter value.

The integrator will start counting seconds from the moment resorcinol starts flowing into the reactor, and it will stop counting once the resorcinol stops flowing. It's also converted to minutes for convenience.

Average concentration of resorcinol during the coating.

Average temperature in the content of the reactor during the coating.

Figure 8.1.7: The yield chart during the simulation after the coating step.

8.2 PLC Control Testing

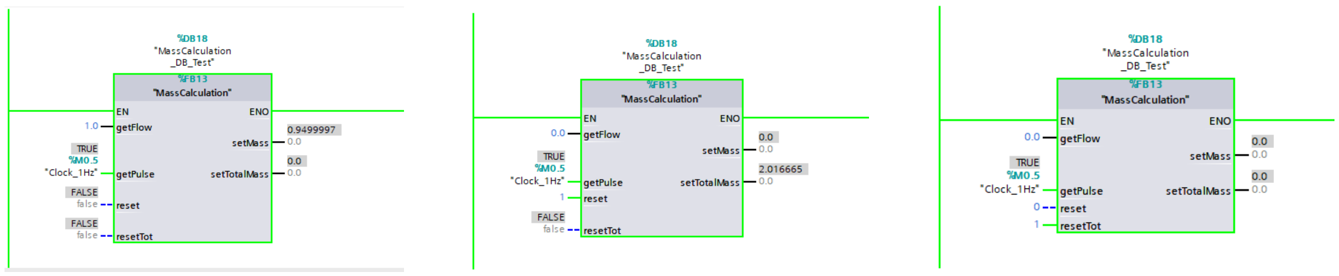
In this section, the testing of each individual FB in the PLC control system will be described and shown. The tests include checking that a specific input gives the expected output. The FBs that represent a step will be tested for their triggers based on the step diagram shown in Figure 2.3.1. The functionality itself will not be described, as this was done in detail in Chapter 6.1.

To execute the tests, an individual OB was made. The different FBs where then placed in this OB, called *Test*, and executed individually. Here the inputs were manipulated, and the output were check for the desired values. All the logic in between is not considered in this test, only the final output value based of the input value. For every FB, the FB inputed into the *Test* LAD based OB will be shown. Above each input and output is a grey square field, in this field is the current value shown. The value underneath is the idle state value.

8.2.1 Supporting Functions

"MassCalculations": The FB has three functionalities, calculation of mass based on a combination of the sampling rate, $getPulse$, and the flow per minute, $getFlow$, the ability to reset the calculated mass value when an adding segment is completed, and reset the total mass of a compound added during the entire process. It is the four inputs $getPulse$, $reset$, $resetTot$ and $getFlow$ that decide these functionalities. Firstly, the calculation itself was tested.

The test parameters were the following, $getPulse = 1$ Hz and $getFlow = 1$ kg/s. During the test, the time from initiation was recorded to monitor the elapsed time, so the manually calculated mass could be compared with the one calculated from the FB. The accumulated mass is equal to the elapsed time multiplied with the flow rate.



(a) Mass Calculation (b) reset (c) resetTot

Figure 8.2.1: Test of $MassCalculation$

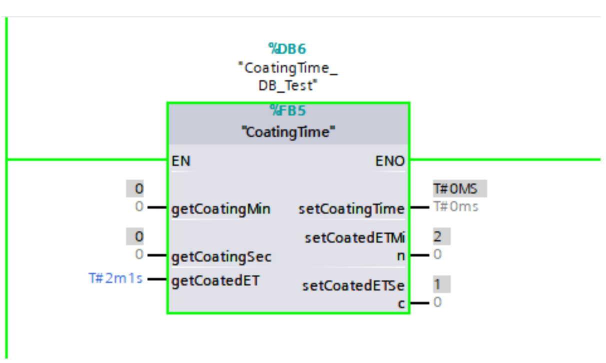
As seen in in Figure 8.2.1a, the mass was equal to 0.9499997 kg, when the recorded time was about 56 seconds. $Mass = 1 \cdot 56/60 = 9.333$. The small deviation is due to the recorded time being inaccurate.

The reset functionality was tested by setting the reset input to 1, equaling $TRUE$. This will only be done when the flow is 0, to reset the mass added in a sequence, as this is used as a trigger value for the steps. As shown in Figure 8.2.1b, when the reset was 1, the $statMass$ variable goes to 0 but the value is stored as $statTotalMass$ to keep a memory of the total amount of a given compound added. The deviation between the mass values in Figure 8.2.1a and in Figure 8.2.1b is due to the simulation not being stopped immediately after the picture was taken.

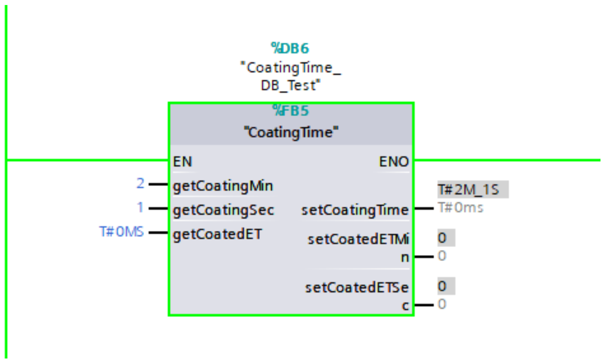
The last functionality is the function of resetting the total mass value $statTotalMass$. This can be done by setting the input, $resetTot$, to 1 or $TRUE$. As seen in Figure 8.2.1c the total mass value gets reset as desired.

"CoatingTime": This FB has two functionalities, converting Time values into two corresponding DINT values representing the time in min and sec, and the opposite. Converting two individual DINT data type values of time to one Time data type value of the time. Firstly, the Time to DINT was tested. The input, $getCoatedET$, was set to a time value of to minutes and one sec, $T\#2m1s$. As seen in Figure 8.2.2a, $setCoatedETMin$ was set to 2, and $setCoatedETSec$ was set to 1. This confirms that the this functionality works.

Secondly, the conversion from DINT to time was tested. The input $getCoatingMin$ was set to 2, and $getCoatingSec$ was set to 1. Which should result in the output returning a value of $T\#2M_1S$. This is confirmed by looking at the right figure in Figure 8.2.2b. It was concluded, based on these tests, that the FB works as desired.



(a) Time to DINT



(b) DINT to Time

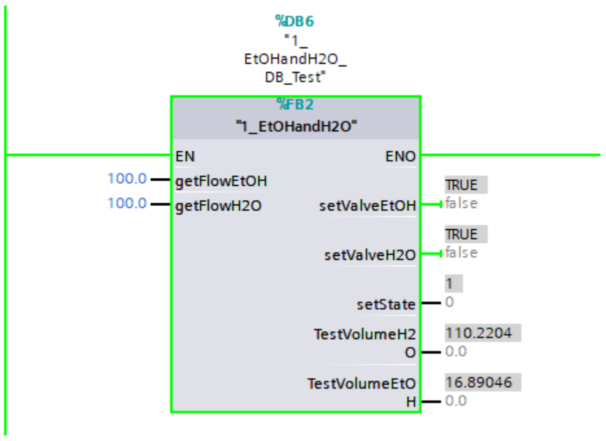
Figure 8.2.2: Test of *CoatingTime*

8.2.2 Preparation of Solution

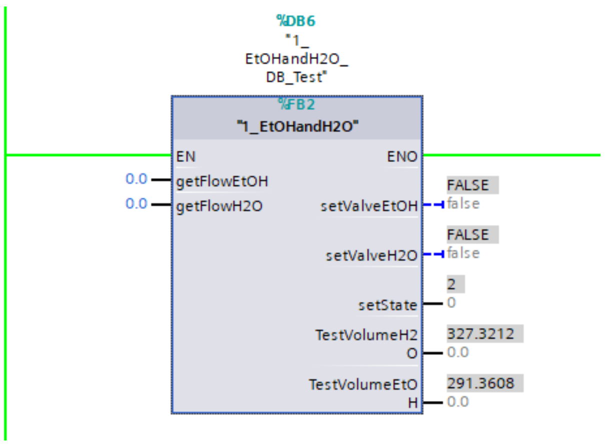
"1_EtOHandH2O": The functionality of this FB is to open the valves that add EtOH and H_2O into the reactor. The trigger to the next step is when the valves are closed and 250 L of EtOH and 320 L of H_2O have been added. This FB has two inputs: $getFlowH2O$ and $getFlowEtOH$, and three outputs: $setValveH20$, $setValveEtOH$ and $setState$. The last two outputs are just for testing purposes.

The inputs $getFlowH2O$ and $getFlowEtOH$ were set to a flow rate of 100 kg/min and were manually closed due to no feedback being present, hence the amount added was not controlled to be a exactly the desired value. The outputs $setValveH20$ and $setValveEtOH$ should be $TRUE$ until the desired amounts have been reached. Then they should switch to $FALSE$, and $setState$ should be set to 2, meaning that the step, 1_$EtOHandH2O$, is done. The inputs were manipulated accordingly as seen in Figure 8.2.3a.

As seen in Figure 8.2.3b, when the variable $TestVolumeH20$ reached 320 L, and $TestVolumeEtOH$ reached 250 L, the valves $setValveH20$ and $setValveEtOH$ were set to $FALSE$, which then set $setState$ to 2, thus changing the step.



(a) Flow present



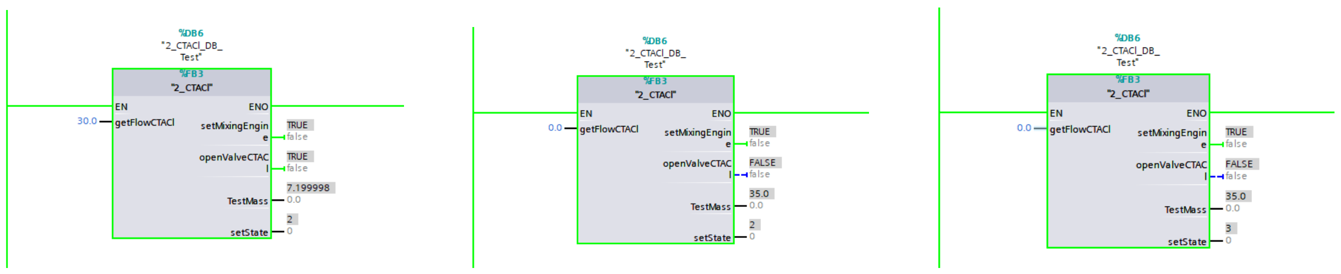
(b) Step Triggered

Figure 8.2.3: Test of 1_$EtOHandH2O$

"2_CTACl": The functionality of this FB is opening a valve to add CTACl into the reactor. The trigger to the next step is when a 15 minute timer is completed, after 30 kg of CTACl has been added into the reactor. This FB has one input: $getFlowCTACl$, and three outputs: $setValveCTACl$, $setMixingEngine$, and $setState$. The other outputs are just for testing purposes.

The input $getFlowCTACl$ was set to a flow rate of 30 kg/min, as seen in Figure 8.2.4a, and was manually closed due to no feedback being present, hence the amount added was not exactly equal to the desired value. The output $setMixingEngine$ is set to $TRUE$ upon initialization of the FB, and the output $setValveCTACl$ should be $TRUE$ until the desired amount is reached. The valve is then closed and the timer is initiated. 15 minutes later, $setState$ should be set to 3, hence switching to step 3, and $setMixingEngine$ should still be $TRUE$. The change of step is an indication that the test was successful.

As seen in Figure 8.2.4b, and 8.2.4c, when the variable $TestMass$ reached 30 kg, $setValveCTACl$ became $FALSE$, and the timer was started. When the timer was completed, the output $setState$ was set to 3, thus changing the step.



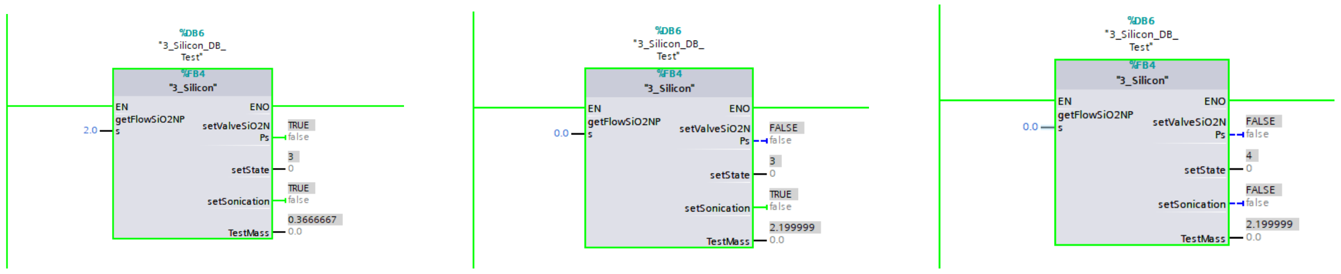
(a) Input Manipulation (b) Waiting for timer (c) Step Triggered

Figure 8.2.4: Test of 2_$CTACl$

"3_Silicon": The functionality of this FB is opening a valve to add silicon nanoparticles into the reactor. The trigger to the next step is when a 30 minute timer is completed, after 20kg of nanoparticles has been added. This FB has one input: $getFlowSiO2NPs$, and three outputs: $setValveSiO2NPs$, $setSonication$, and $setState$. The other output is only for testing purposes.

The input $getFlowSiO2NPs$ was set to a flow of 2 kg/min, as seen in Figure 8.2.5a, and was manually closed due to the lack of a feedback, hence the amount added was not exactly equal to the desired value of 2 kg. The output $setValveSiO2NPs$ should be $TRUE$ until the desired amount is reached. The valve is then closed, the timer initiated, and $setSonication$ is set to $TRUE$. After 30 minutes, when the timer is done, $setSonication$ should be set to $FALSE$, and $setState$ should be set to 4, hence switching to step 4, the next stage of the process. This was an indication that the test is successfully completed.

As seen in Figure 8.2.5b and 8.2.5c, when the variable $TestMass$ reached 2 kg, the valve $setValveSiO2NPs$ became $FALSE$, and the timer was started. When the timer was completed, the $setSonicator$ was set to $FALSE$ and $setState$ to 4, thus changing the step.



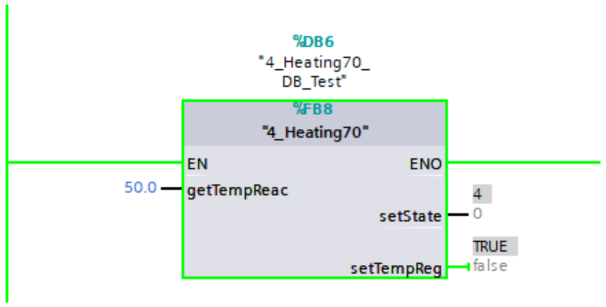
(a) Input manipulation (b) Waiting for timer (c) Step Triggered

Figure 8.2.5: Test of 3_$Silicon$

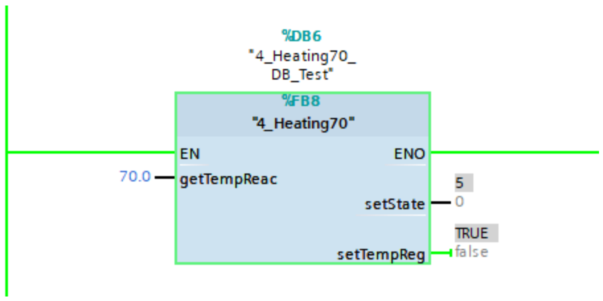
"4_Heating70": The functionality of this FB is to start the heating regulation of the contents in the reactor. The trigger to the next step is when the temperature in the reactor has reached 70°C. This FB has one input: *getTempReac*, and two outputs: *setTempReg*, and *setState*.

The input *getTempReac* was firstly set to 50°C, to check that it did not trigger to the next step. This can be seen in Figure 8.2.6a. Thereafter, the input was set to 70°C, and the following should happen. The output *setTempReg* should be *TRUE* when the FB is initiated and it should also be *TRUE* when the step is changed. *setState* should be set to 5 due to the temperature being at the desired level, hence switching to step 5. The change of step is an indication that the test has been successfully completed.

As seen in the Figure 8.2.6a, with the temperature *getTempReac* being 70°C, *setState* was set to 5, thus changing the step, and the variable *setTempReg* was still *TRUE* as it should have been.



(a) Input on 50, not triggered



(b) Input on 70, triggered

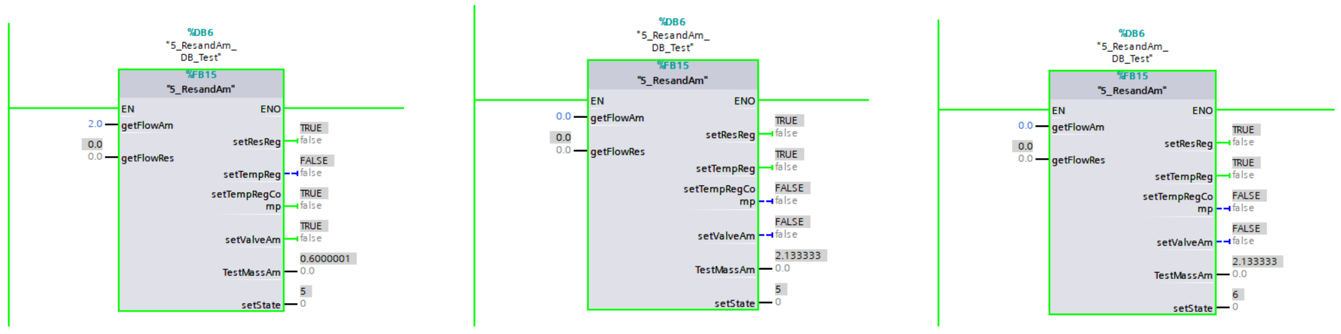
Figure 8.2.6: Test of 4_*Heating70*

8.2.3 Coating of Particles

"5_ResandAm": The functionality of this FB is to start the regulation of the resorcinol valve and to add 2 kg of ammonia. The trigger to the next step is when the ammonia has been added and a five-minute timer is completed. This FB has two inputs: *getFlowAm* and *getFlowRes*, and five outputs: *setResReg*, *setTempReg*, *setTempRegComp* *setValveAm*, and *setState*. The other outputs are just for testing purposes.

To test the FB, the inputs *getFlowAm* was set to 2 kg/min., as seen in Figure 8.2.7a. *getFlowAm* was ignored in this test as the PID normally controls this input. The output *setResReg* should be set *TRUE* when the FB is initiated, and should keep the value *TRUE* when the step is changed. *setValveAm* should be set *FALSE*, after 2 kg is added. Before the valve is closed, the *setTempRegComp* should be *TRUE*, and *setTempReg* *FALSE*, when the valve closes they should switch. When the five-minute timer is completed, *setState* should go to 6, hence switching to step 6. If the outputs are these values, it is an indication that the test has been completed successfully.

As seen in Figure 8.2.7b, and 8.2.7c. when *TestMassAm* reached 2 kg the timer was initiated. when the timer was completed, the FB sets *setState* to 6, thus changing the step, and the variable *setResReg* and *setTempReg* was *TRUE*, and *setValveAm* and *setTempRegComp* was *FALSE* as it should have been.



(a) Input manipulation (b) Waiting for timer (c) Step triggered

Figure 8.2.7: Test of 5_*ResandAm*

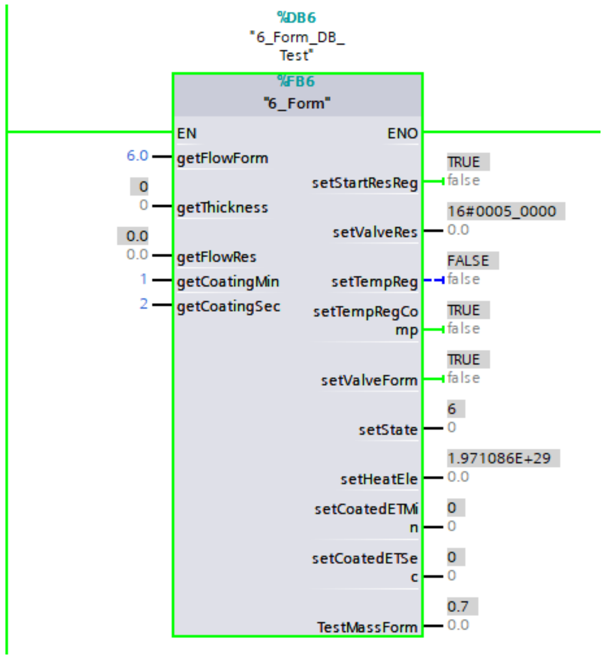
"6_Form": The functionality of this FB is to add the desired amount of formaldehyde into the reactor during the coating phase, and one final addition after the adding of resorcinol is completed. This FB has three triggers to new steps. The first trigger is when the first coating layer of formaldehyde is completed, meaning *getThickness* is 0, and the coating timer is done, which triggers to step 7. The second trigger is when 500nm coating is selected, meaning *getThickness* is 2 and the coating timer is completed, which triggers to step 7 and advancing from there to step 5 starting a new coating layer as long as *statCounter* is less than five, if *statCounter* is five the step is triggered to step 8 after a 60minute timer. The third trigger is a combination of a 60 minute timer being completed, and 100nm coating being selected which triggers to step 8. This FB has five inputs: *getFlowForm*, *getThickness*, *getFlowRes* *getCoatingMin*, and *getCoatingSec*, and nine outputs: *setStartResReg*, *setValveRes*, *setTempReg*, *setTempRegComp*, *setValveForm*, *setHeatingEle*, *setCoatingETMin*, *setCoatingETSec*, and *setState*. The last output is just for testing purposes.

Firstly, the first trigger was tested. The input parameters were set to be *getThickness* = 0, *getFlowForm* = 6, *getCoatingMin* = 1 and *getCoatingSec* = 2. The expected result from this was that 6 kg of formaldehyde would be added, and when the coating timer was completed, the *statCounter* would increment to 1 and *setState* would be set to 7.

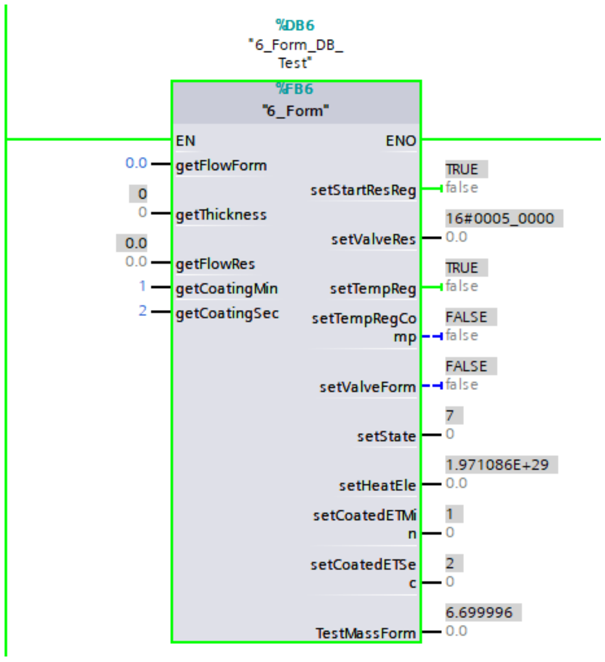
As seen in Figure 8.2.8a, and 8.2.8a. when the variable *TestMassForm* reached 6 kg, the valve was closed and the *statStartCoatingTimer* became *TRUE*, thus starting the coating timer. The timer waited for the desired time, as seen by the values of *setCoatedETMin* and *setCoatedETSec*. The conditions shown above lead to *setStep* being set to 7. The values for *setValveResReg*, and *setHeatEle* are confusing due to the fact that the PID is not getting a setpoint value to regulate to.

Secondly, the second trigger was tested. The input parameters were set to be *getThickness* = 1, *getFlowForm* = 6, *getCoatingMin* = 1, *getCoatingSec*, and = 2. The expected result from this is that 6 kg of formaldehyde will be added, which will start the coating timer. When the coating timer is completed, *setState* will be set to 8. Before the step is triggered, the temperature regulation, the regulation of resorcinol concentration, *setStartResReg* should stop, and the resorcinol valve, *setValveRes*, should be set to 0 and, the PID with better compensation capabilities, *setTempRegComp*, should be used.

As seen in Figure 8.2.9a, and 8.2.9b, When the variable *getThickness* was 1 and 6 kg of formaldehyde had been added, the valve *setValveForm*, *setTempRegComp* went *FALSE*, and the coating timer all other variables were set to 0, with the exception of *setState*. *setState* was set to 5, hence changing the step.

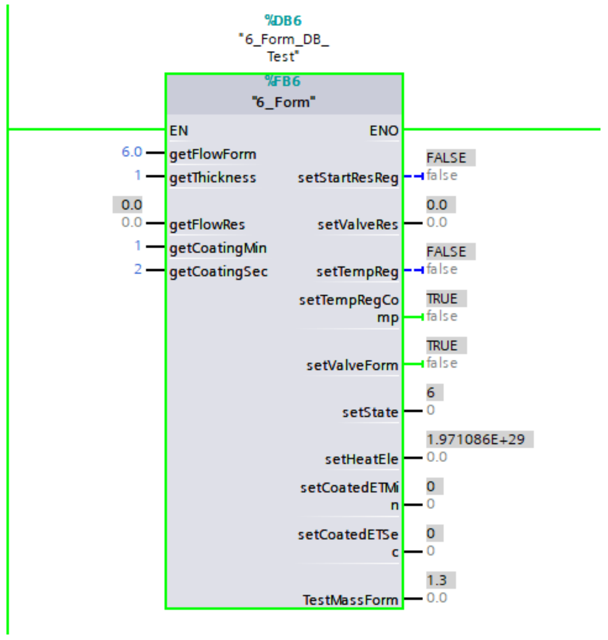


(a) Input manipulated

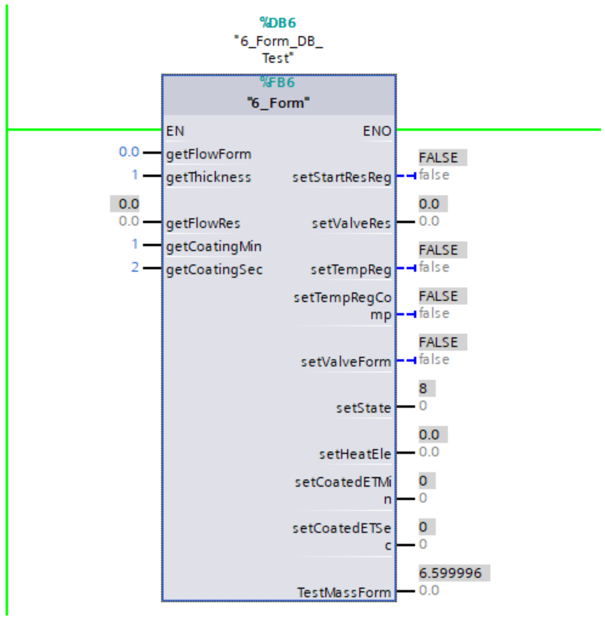


(b) Step triggered

Figure 8.2.8: Test of 6_*Form* at first addition



(a) Input manipulated

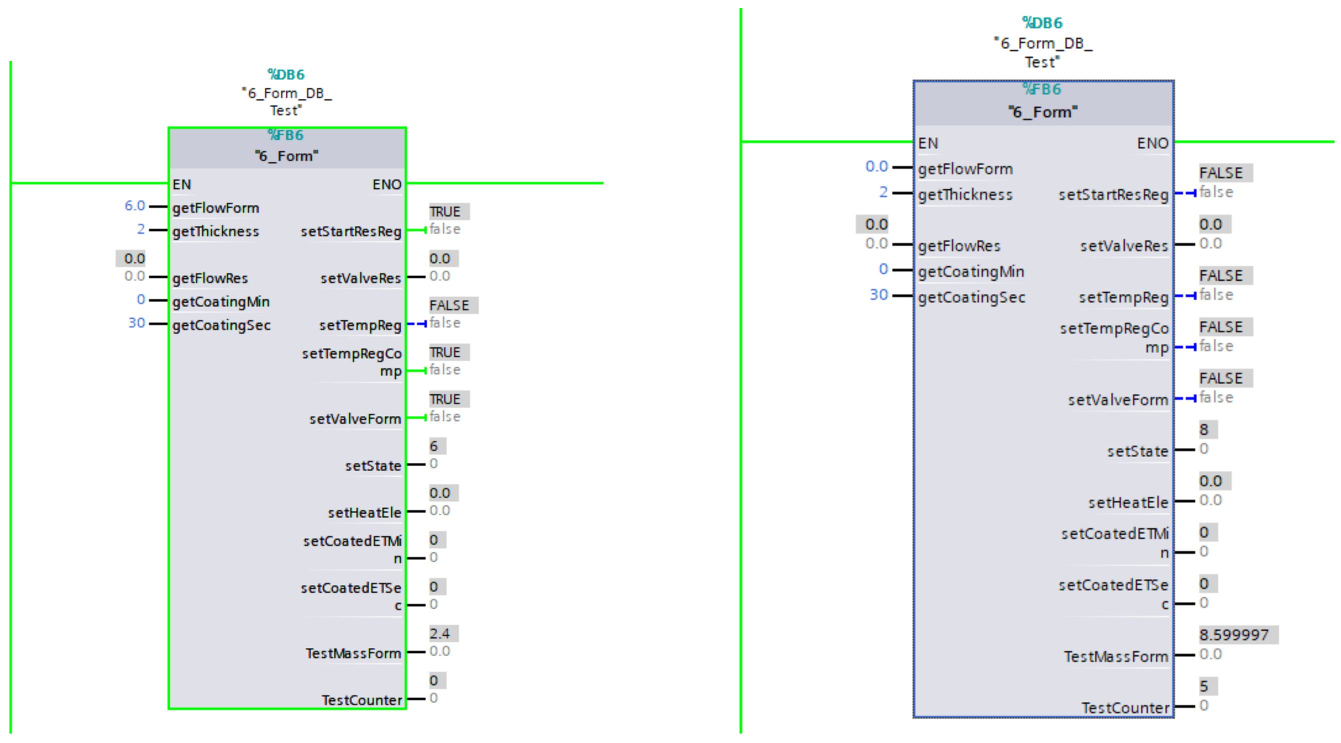


(b) Step Triggered

Figure 8.2.9: Test of 6_*form* with 100 *nm* coating selected

Finally, the third trigger was tested. The input parameters were set to be *getThickness* = 2, *getFlowForm* = 6, *getCoatingMin* = 1, and *getCoatingSec* = 2 . The expected result from this is that 6 kg of formaldehyde will be added, which will initiate a timer. When the timer is completed, *setState* will be set to 8. Before 6 kg is reached, and *statCounter* less than five 5, the *setTempRegComp* should be *TRUE*, regulating the reactor temperature, and the resorcinol concentration, *setStartResReg*, should be *TRUE*, until *statCounter* reaches five, then it should switch to *FALSE*.

As seen in Figure 8.2.10, when the variables $statCounter = 5$ and $getThickness = 2$, the valve $setValveForm$ became $FALSE$ after 6 kg of formaldehyde was added. $setState$ was set to 8, hence changing the step. $setStartResReg$ and $setTempReg$ also became $FALSE$, turning off the resorcinol and temperature regulation. With both the heating element, $setHeatingEle$, and the resorcinol valve, $setValveRes$, being set to 0 as well.



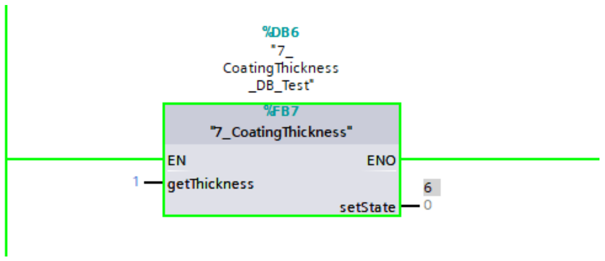
(a) Input manipulation (b) Step triggered

Figure 8.2.10: Test of 6_$Form$ with 500 nm coating selected

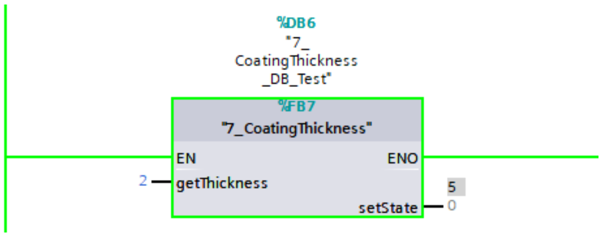
"7_CoatingThickness": The functionality of this FB is to be a "resting" step in which the HMI screen can show the coating thickness selection buttons, hence giving the operator the ability to select the desired coating thickness. This FB has two triggers, one when 100 nm is selected, and one when 500 nm is selected. This FB has one input: $getThickness$, and one output: $setState$.

Firstly, the input $getThickness$ was set to 2, to test the selection of 500 nm coating. The output $setState$ should be set to 5, hence switching to step 5. This is an indication that the test is successful. As seen in Figure 8.2.11a, when the variable $getThickness = 2$, $setState$ goes to 5, hence changing the step.

Secondly, the input $getThickness$ was set to 1, to test the selection of 100 nm coating. The output $setState$ should be set to 6, hence switching to step 6. This is an indication that the test is successful. As seen in the Figure 8.2.11b, when the variable $getThickness = 1$, $setState$ goes to 6, hence changing the step.



(a) 100 *nm* coating selected



(b) 500 *nm* coating selected

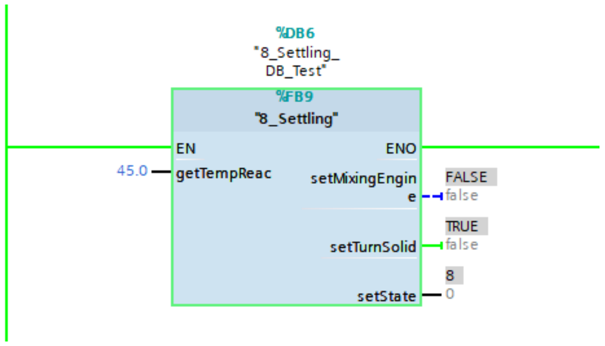
Figure 8.2.11: Test of 7_*CoatingThickness*

8.2.4 Washing and Waste Removal

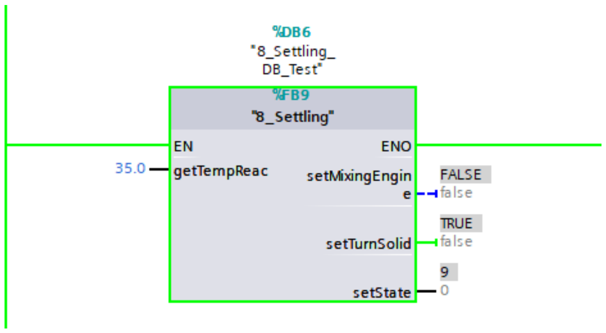
"8_Settling": The functionality of this FB is to let the contents of the reactor settle before changing to the next step. The FB has one trigger, when the timer for 10 hours is completed. This FB has one input: *getTempReac*, and three outputs: *setMixingEngine*, *setTurnSolid* and *setState*.

The input *getTempReac* was firstly set to 45°C, to check if it triggers to the new step. A seen in Figure 8.2.12a, it does not trigger to the next step, which is correct. Thereafter, the input *getTempReac* was set to 35°C. The ten-hour timer should then start, and when the timer is completed, the output *setState* should be set to 9, hence switching to step 9. This is an indication that the test is successful.

As seen in Figure 8.2.12b, when the variable *getTempReac* was equal to 35, *setState* was set to 9, hence changing the step.



(a) Temperature input set to 45, not triggered



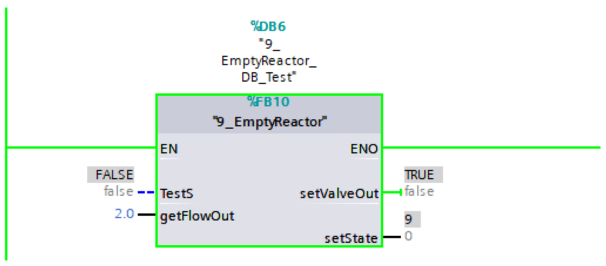
(b) Temperature input set to 35, triggered

Figure 8.2.12: Test of 8_*Settling*

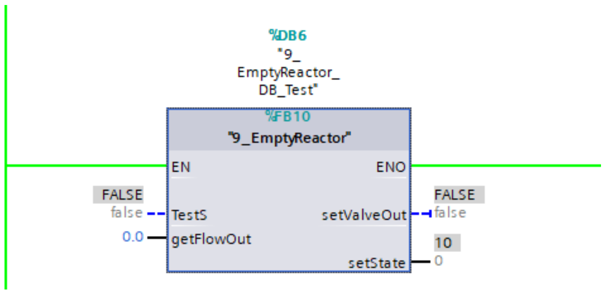
"9_EmptyReactor": The functionality of this FB is to empty the reactor of its liquid contents before changing to the next step. The FB has two triggers, the first to remove the waste from the coating phase and the second to remove the washing water. This FB has one input: *getFlowOut*, and two outputs: *setValveOut* and *setState*. *Tests* is just for testing purposes.

Firstly, to test the first trigger, the input *getFlowOut* was set to 2 and then to 0, to simulate that the reactor is emptied. The edge detector should then detect the negative edge. *statS* is set to *FALSE* by *TestS*, signaling that 8_*Settling* was the previous step. Therefore, the output *setState* should be set to 10, hence switching to step 10. This is an indication that the test is successful.

As seen in Figure 8.2.13, when the negative edge was detected, and *statS* was set to *FALSE*, the *setState* was set to 10, hence changing the step.



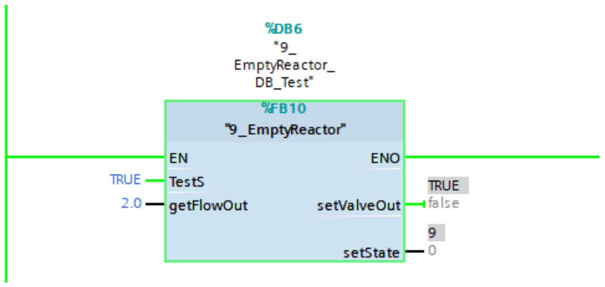
(a) Input manipulated



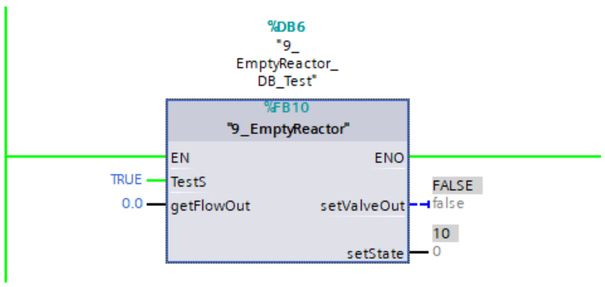
(b) Edge detector triggered, step switched

Figure 8.2.13: Test of 9_*EmptyReactor* with *statS FALSE*

Secondly, the second trigger was tested. Almost every aspect of this trigger is equal to the first trigger, the only difference is that *statS* has been set to *TRUE*, by *TestS*, signaling that step 10_*Washing* was the previous step. Therefore, when a negative edge is detected it the output *setState* should go to 11, hence switching to step 11.
As seen in Figure 8.2.14, when the negative edge was detected and *statS* was *TRUE*, *statS* was set *FALSE* and the *setState* went to 11, hence changing the step.



(a) Input manipulation



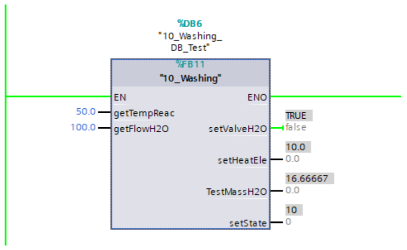
(b) Edge detector triggered, step switched

Figure 8.2.14: Test of 9_*EmptyReactor* with *statS TRUE*

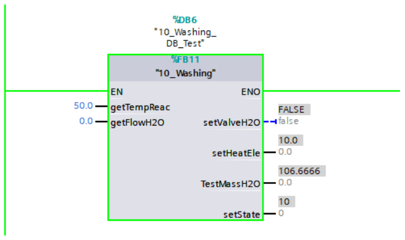
"10_Washing": The functionality of this FB is to wash the coated silicon nanoparticles before changing to the next step. The FB has one trigger when the washing timer is completed. This FB has two inputs: *getFlowH2O* and *getTempReac*, and three outputs: *setValveH2O*, *setTempReg*120 and *setState*. *TestMassH2O* is just for testing.

To test the trigger, the input *getFlowH2O* was set to 100 kg/min. This will then be set to 0 when 100 L is reached, and the the temperature regulation starts by setting *setHeatingEle* to 10. Then the timer is started. When the timer is done, and the desired temperature of 100°C is reached, *setHeatingEle* is set to 0 and the output *setState* should be set to 9, hence switching to step 9. This is an indication that the test is successful.

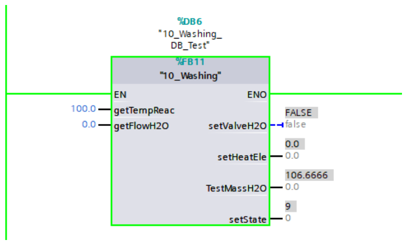
As seen in Figure 8.2.15, when the compound has been added, the valve closed, *setValveH2O* went *FALSE*, and *setHeatingEle* was set to 0 when the desired temperature reached 100°C and the timer was done, then the *setState* went to 10, hence changing the step.



(a) Input Manipulation



(b) Stopping H_2O flow



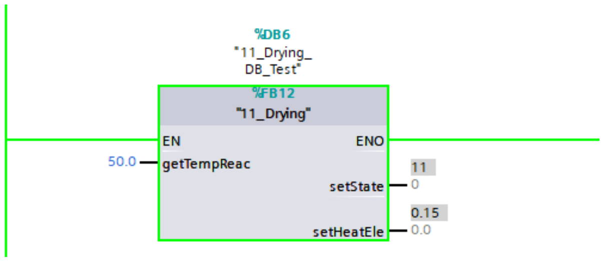
(c) Setting temperature to 100, triggering to next step

Figure 8.2.15: Test of 10_*Washing*

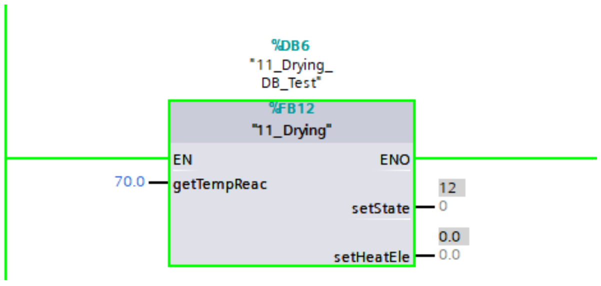
"11_Drying": The functionality of this FB is to dry the coated silicon nanoparticles before changing to the next step. The FB has one trigger when the drying timer is completed. This FB has one input: *getTempReac*, and two outputs: *setHeatEle* and *setState*.

Firstly, the input *getTempReac* was set to 50 °C, to see if it triggered to the next step. As seen in Figure 8.2.16a, the step is not triggered, which is correct. To test the trigger, the input *getTempReac* was set to 70 °C. This should start the drying timer. The output *setHeatEle* should be set to 0.15 when the FB is initiated. When the timer is done, *setHeatEle* is set to 0 and the output *setState* should be set to 12, hence switching to step 12. This is an indication that the test is successful.

As seen in Figure 8.2.16, when the *getTempReac* was 70 and the timer was completed. *setHeatEle* went to 0, and *setState* went to 12, hence changing the step.



(a) In



(b) DINT to Time

Figure 8.2.16: Test of 11_*Drying*

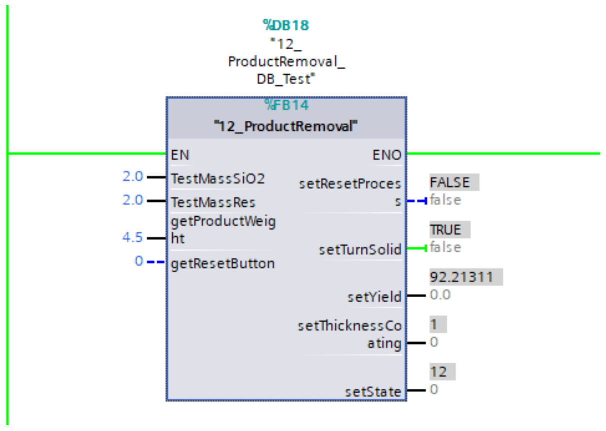
"12_ProductRemoval": The functionality of this FB is to reset the process when the final product, the coated silicon nanoparticles, has been removed from the reactor. The FB has one trigger when the reset button on the HMI panel has been pressed. This FB has two inputs: *getProductWeight* and *getResetButton*, and five outputs: *setResetProcess*, *setTurnSolid*, *setYield*, *setThicknessCoating*, and *setState*. The two other inputs are just for testing purposes.

Firstly, the testing inputs, *TestMassSiO2* and *TestMassRes*, were set to 2 kg, and the *getProductWeight* was set to 4.5 kg. Mathematically this should result in the following yield:

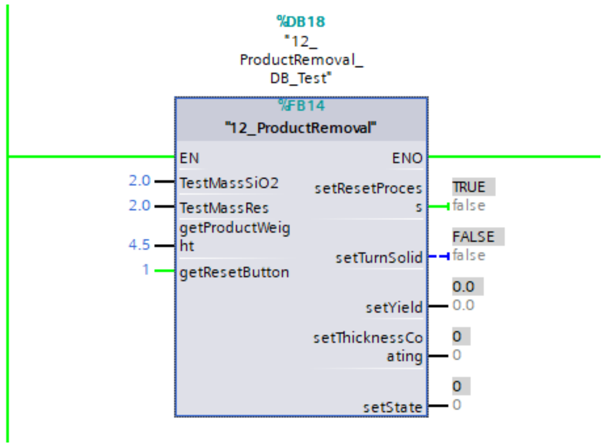
$$setYield = \frac{getProductWeight}{TestMassSiO2 + 1.44 \cdot TestMassRes} = \frac{4.5}{2 + 1.44 \cdot 2} = 0.922 \cdot 100 = 92.22\% \qquad (8.12)$$

If the yield, in Figure 8.2.17a, is compared to the calculated yield above it can be confirmed that the yield calculation is correct.

Then the input *getResetButton* was set to 1, to test the process reset functionality. As seen in Figure 8.2.17b, this sets all values to 0 or inverts it thus resetting the value. *setState* is set to 0, meaning the process is back at the idle step.



(a) Test of yield calculation

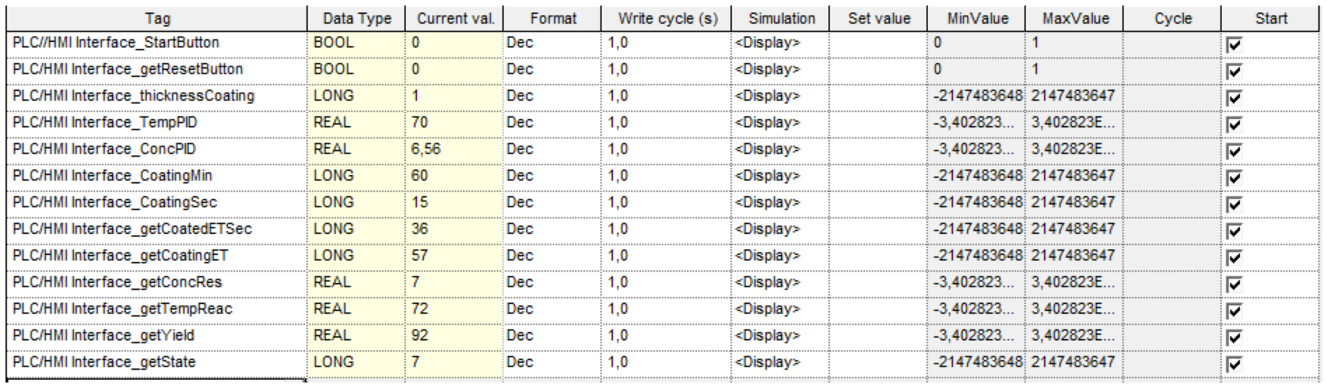


(b) Test of reset function

Figure 8.2.17: Test of 12_*ProductRemoval*

8.3 HMI Testing

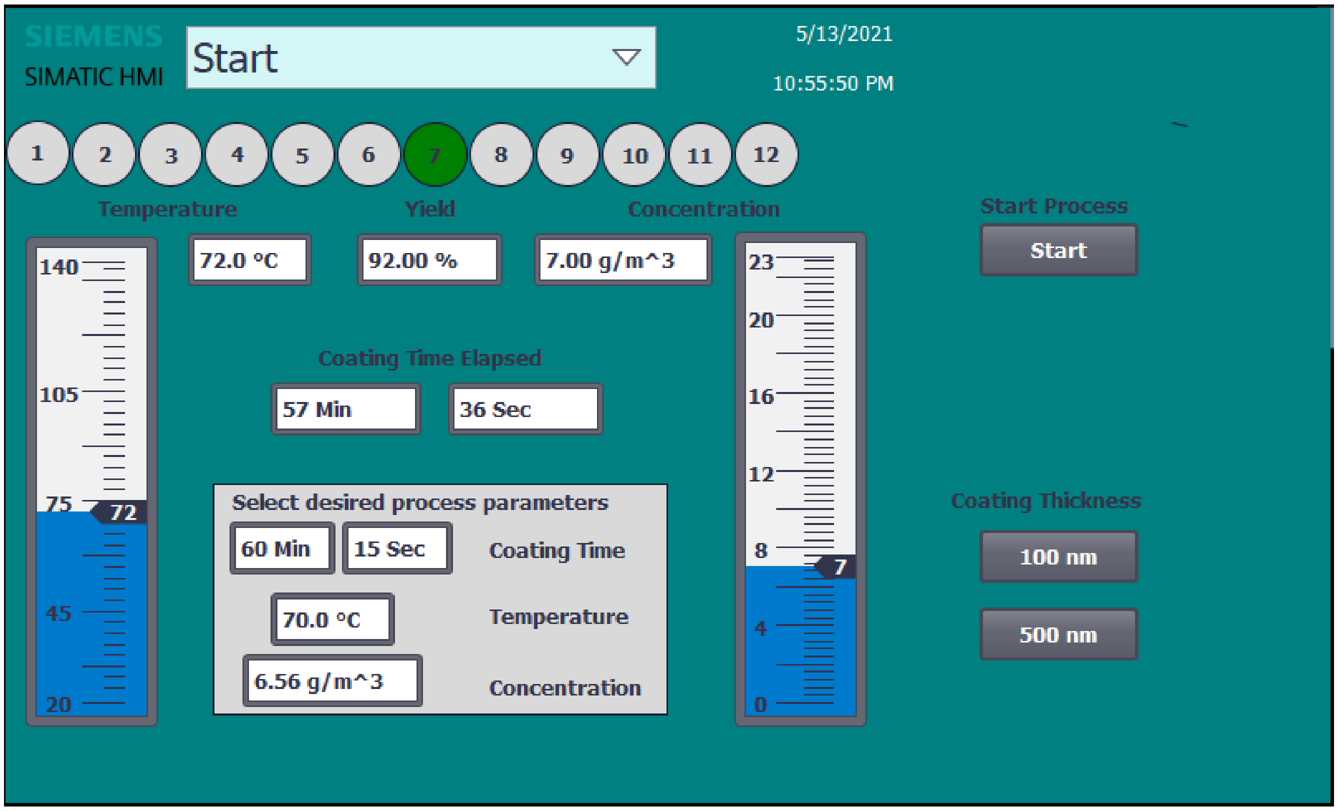
In this section the results of the testing of the individual functionality of the HMI will be shown. The HMIs functionality can be split in to two categories, the ability to display value, and the ability to receive inputs. In the TIA Portal it is possible to run a tag simulation. This gives the tester an input field to the tags that are used on the HMI to either display data or save data, from/to the PLC. Therefore, by manipulating these tags, it is also tested that the correct display changes, and that the correct input fields are connected to the correct tags. The manipulation field for the tags can be seen in Figure 8.3.1.



Tag	Data Type	Current val.	Format	Write cycle (s)	Simulation	Set value	MinValue	MaxValue	Cycle	Start
PLC//HMI Interface_StartButton	BOOL	0	Dec	1,0	<Display>		0	1		✓
PLC/HMI Interface_getResetButton	BOOL	0	Dec	1,0	<Display>		0	1		✓
PLC/HMI Interface_thicknessCoating	LONG	1	Dec	1,0	<Display>		-2147483648	2147483647		✓
PLC/HMI Interface_TempPID	REAL	70	Dec	1,0	<Display>		-3,402823...	3,402823E...		✓
PLC/HMI Interface_ConcPID	REAL	6,56	Dec	1,0	<Display>		-3,402823...	3,402823E...		✓
PLC/HMI Interface_CoatingMin	LONG	60	Dec	1,0	<Display>		-2147483648	2147483647		✓
PLC/HMI Interface_CoatingSec	LONG	15	Dec	1,0	<Display>		-2147483648	2147483647		✓
PLC/HMI Interface_getCoatedETSec	LONG	36	Dec	1,0	<Display>		-2147483648	2147483647		✓
PLC/HMI Interface_getCoatingET	LONG	57	Dec	1,0	<Display>		-2147483648	2147483647		✓
PLC/HMI Interface_getConcRes	REAL	7	Dec	1,0	<Display>		-3,402823...	3,402823E...		✓
PLC/HMI Interface_getTempReac	REAL	72	Dec	1,0	<Display>		-3,402823...	3,402823E...		✓
PLC/HMI Interface_getYield	REAL	92	Dec	1,0	<Display>		-3,402823...	3,402823E...		✓
PLC/HMI Interface_getState	LONG	7	Dec	1,0	<Display>		-2147483648	2147483647		✓

Figure 8.3.1: Overview of the tag manipulation window

The first three tags, are the tags for the buttons, the next four are for the input fields, and all the ones starting with *get* are displays, with the exception of *getState*. The reset button and the coating thickness button should just be visible in step 7, for the coating thickness button, and step 12, for the reset button. As seen in Figure 8.3.1, is *getState* seven, meaning the coating thickness button should be visible on the HMI screen which can be seen in Figure 8.3.2. The visual indicator that the process is in step 7 is confirmed to be functional. The button 100 *nm* was pressed which should have set *thicknessCoating* to 1, which is conformed by Figure 8.3.1. The rest of the inputs and displays can be seen to give the correct value to their corresponding tag or display. Meaning the test is deemed successful.



72.0 °C

92.00 %

7.00 g/m^3

Coating Time Elapsed

57 Min 36 Sec

Select desired process parameters

60 Min 15 Sec Coating Time

70.0 °C Temperature

6.56 g/m^3 Concentration

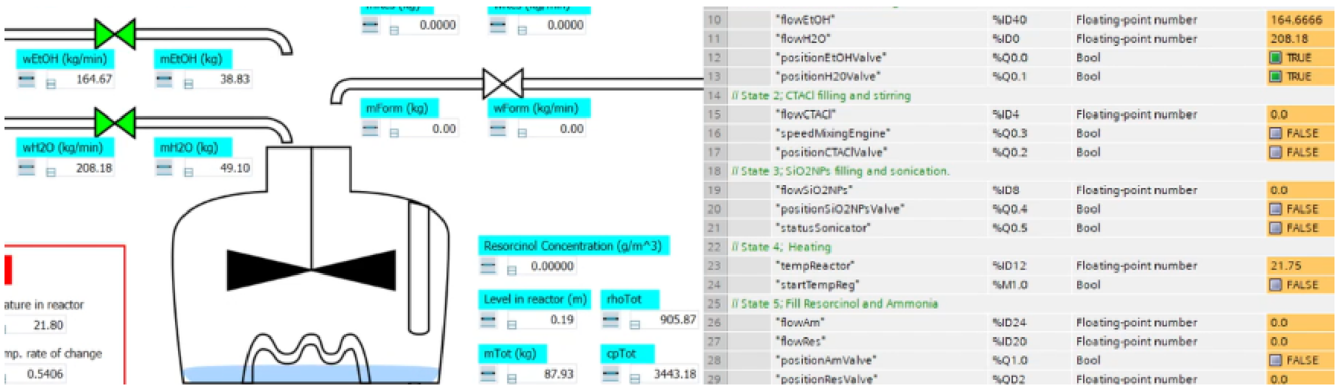
Figure 8.3.2: Test of HMI screen *Start*

8.4 System Testing

This section will show how the simulated process works along with the control system. A PLCSIM Advanced coupling has been established in SIMIT, which will create a PLC instance with its own IP adress once the simulation in SIMIT is run. While it is online, the PLC project can be compiled and uploaded to the PLC instance that SIMIT created, which makes running both together possible. The details of the established connection is explained closer in Chapter 4. The details of the process results will be explained with text and accompanying pictures in this section. It was preferred to also include a video of the process from start to finish because this would make it simple to see the behavior in real time. However, it was unfortunately impossible with the computers available during this project. Several attempts were made using the screen recording software OBS Studio, but the simulation did not run properly due to computer hardware limitations. An inspection of the task manager during the simulations with the coupled system confirmed that the CPU was continuously operating at 95-100 % capacity without having OBS Studio open and running.

The system testing is run on dual screen monitors. The watch table in the TIA Portal can be open on one screen while the *Overview* chart in SIMIT is open on the other screen, and that way, both the simulated process and the PLC can be monitored at the same time. When the simulation is initiated and run in SIMIT, it will take some seconds before the PLC instance is created. After the compiled PLC project is uploaded to this instance, the watch table in the TIA Portal can be set in online monitoring mode and the startbutton on the HMI screen can be pressed to actually start the simulated process.

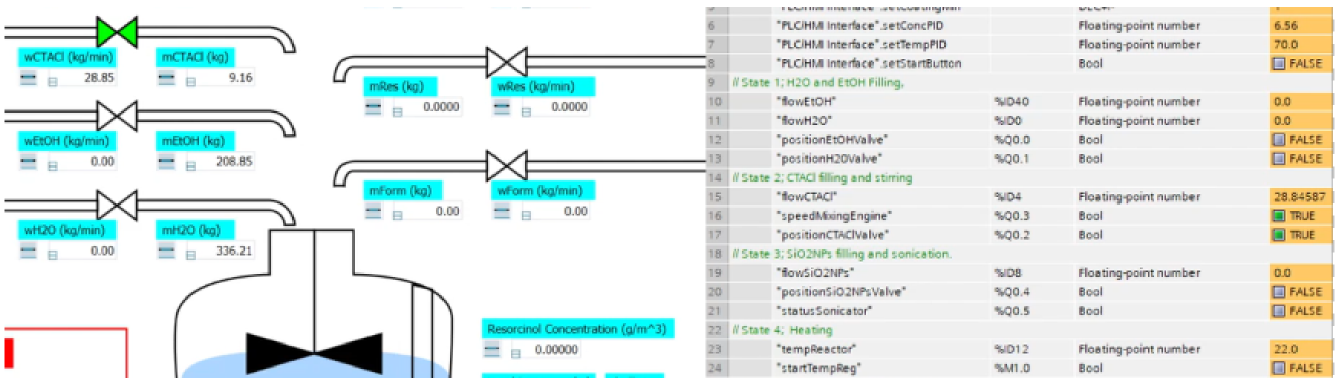
Step 1: Once the startbutton is pressed, the FB *ProcessSequence* in the OB *Main* will receive this signal and go from step 0 to step 1, which is the step where H_2O and $EtOH$ is added. As expected after step 1 is entered, the flow rate and mass accumulation of H_2O and $EtOH$ in SIMIT is affected, the illustrated valves turn green, and the blue colored illustrated level in the reactor starts increasing. Figure 8.4.1 shows the relevant regions of SIMIT and the TIA Portal's watch table side by side. By comparing the values of $wEtOH$ and $wH2O$ in SIMIT to the values of $flowEtOH$ and $flowH2O$ in the watch table, it is an exact match when taking the rounding of decimals into account.



10	"flowEtOH"	%ID40	Floating-point number	164.6666
11	"flowH2O"	%ID0	Floating-point number	208.18
12	"positionEtOHValve"	%Q0.0	Bool	🟩 TRUE
13	"positionH20Valve"	%Q0.1	Bool	🟩 TRUE
14	// State 2; CTACl filling and stirring			
15	"flowCTACl"	%ID4	Floating-point number	0.0
16	"speedMixingEngine"	%Q0.3	Bool	☐ FALSE
17	"positionCTAClValve"	%Q0.2	Bool	☐ FALSE
18	// State 3; SiO2NPs filling and sonication.			
19	"flowSiO2NPs"	%ID8	Floating-point number	0.0
20	"positionSiO2NPsValve"	%Q0.4	Bool	☐ FALSE
21	"statusSonicator"	%Q0.5	Bool	☐ FALSE
22	// State 4; Heating			
23	"tempReactor"	%ID12	Floating-point number	21.75
24	"startTempReg"	%M1.0	Bool	☐ FALSE
25	// State 5; Fill Resorcinol and Ammonia			
26	"flowAm"	%ID24	Floating-point number	0.0
27	"flowRes"	%ID20	Floating-point number	0.0
28	"positionAmValve"	%Q1.0	Bool	☐ FALSE
29	"positionResValve"	%QD2	Floating-point number	0.0

Figure 8.4.1: A view of SIMIT (left) and the PLC (right) during step 1.

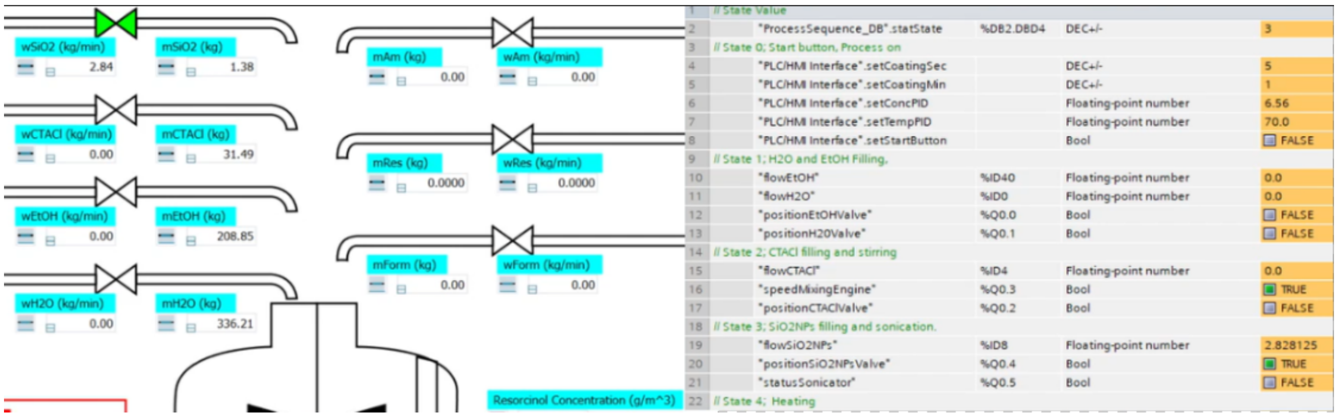
The PLC will close the valves once the desired amounts have been added. However, the total amount of an added compound will not exactly match the desired amount. This is because the valve in SIMIT has a closing time of one second, which means that there is a delay from the closing signal until the valve is fully closed. The compound will continue flowing through the valve during this delay time, ultimately accumulating an excess amount of mass. This can be seen by the closed valves and values of $mH2O$ and $mEtOH$ in Figure 8.4.2. These values should preferably be $mH2O = \frac{320\ L}{1000} \cdot 998\ kg/m^3 = 319.36\ m^3$ and $mEtOH = \frac{250\ L}{1000} \cdot 789.4\ kg/m^3 = 197.35\ m^3$.



6	"PLC/HMI Interface".setConcPID		Floating-point number	6.56
7	"PLC/HMI Interface".setTempPID		Floating-point number	70.0
8	"PLC/HMI Interface".setStartButton		Bool	☐ FALSE
9	// State 1; H2O and EtOH Filling.			
10	"flowEtOH"	%ID40	Floating-point number	0.0
11	"flowH2O"	%ID0	Floating-point number	0.0
12	"positionEtOHValve"	%Q0.0	Bool	☐ FALSE
13	"positionH20Valve"	%Q0.1	Bool	☐ FALSE
14	// State 2; CTACl filling and stirring			
15	"flowCTACl"	%ID4	Floating-point number	28.84587
16	"speedMixingEngine"	%Q0.3	Bool	🟩 TRUE
17	"positionCTAClValve"	%Q0.2	Bool	🟩 TRUE
18	// State 3; SiO2NPs filling and sonication.			
19	"flowSiO2NPs"	%ID8	Floating-point number	0.0
20	"positionSiO2NPsValve"	%Q0.4	Bool	☐ FALSE
21	"statusSonicator"	%Q0.5	Bool	☐ FALSE
22	// State 4; Heating			
23	"tempReactor"	%ID12	Floating-point number	22.0
24	"startTempReg"	%M1.0	Bool	☐ FALSE

Figure 8.4.2: A view of SIMIT (left) and the PLC (right) during step 2.

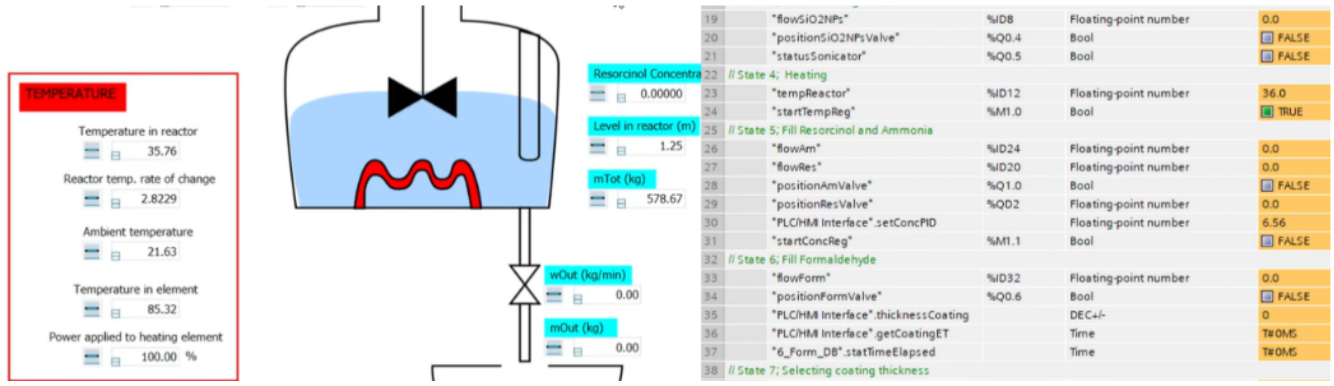
Step 2: Figure 8.4.2 also shows the relevant regions for step 2, where $CTACl$ is added and the agitator is turned on. The $CTACl$ valve is green in SIMIT, and the monitored flow rate in SIMIT is identical the flow rate in the watch table. The mixer can also be seen in a slightly different position from the initial position, this is due to the fact that it is in a spinning animation initiated by the $speedMixingEngine$ signal from the PLC, indicating that this control signal also works as intended. The step will not change before a 15 minute timer is done after the $CTACl$ valve is closed. This was verified by watching the timer value when viewing the FB 2_$CTACl$ in online mode, and also paying attention to whether the components of the simulation are inactive for this duration. The 15 minute timer was replaced by a 15 second timer for testing purposes.



// State Value			
"ProcessSequence_DB".statState	%DB2.DBD4	DEC+/-	3
// State 0; Start button, Process on			
"PLC/HMI Interface".setCoatingSec		DEC+/-	5
"PLC/HMI Interface".setCoatingMin		DEC+/-	1
"PLC/HMI Interface".setConcPID		Floating-point number	6.56
"PLC/HMI Interface".setTempPID		Floating-point number	70.0
"PLC/HMI Interface".setStartButton		Bool	☐ FALSE
// State 1; H2O and EtOH Filling.			
"flowEtOH"	%ID40	Floating-point number	0.0
"flowH2O"	%ID0	Floating-point number	0.0
"positionEtOHValve"	%Q0.0	Bool	☐ FALSE
"positionH20Valve"	%Q0.1	Bool	☐ FALSE
// State 2; CTACl filling and stirring			
"flowCTACl"	%ID4	Floating-point number	0.0
"speedMixingEngine"	%Q0.3	Bool	🟩 TRUE
"positionCTAClValve"	%Q0.2	Bool	☐ FALSE
// State 3; SiO2NPs filling and sonication.			
"flowSiO2NPs"	%ID8	Floating-point number	2.828125
"positionSiO2NPsValve"	%Q0.4	Bool	🟩 TRUE
"statusSonicator"	%Q0.5	Bool	☐ FALSE
// State 4; Heating			

Figure 8.4.3: A view of SIMIT (left) and the PLC (right) during step 3.

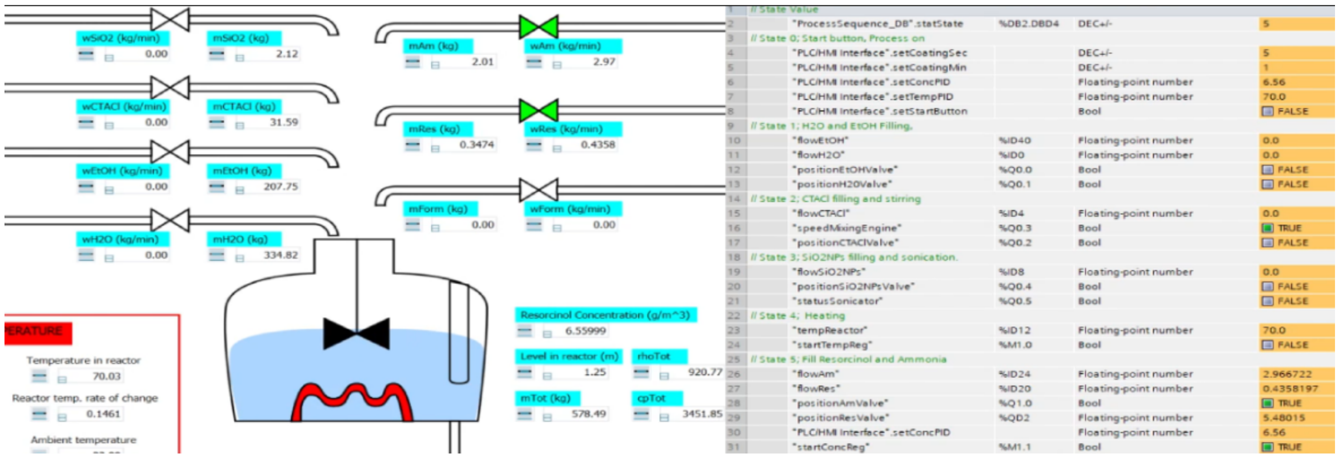
Step 3: Figure 8.4.3 shows the third step, where the silicon nanoparticles are added and the sonicator is activated. Like the previous steps, the open valve is indicated by the green valve in the SIMIT chart, and the flow rate and accumulated mass is monitored directly below the valve. The flow rate $wSiO2$ is also the same as $flowSiO2NPs$ monitored in the watch table, aside from the differing level of precision regarding number of decimals. The sonicator is not turned on in the figure, this is because it will be activated once the $SiO2$ valve is closed. A 30 minute timer will also be activated at the same time, which means that the sonicator is active for 30 minutes, or rather 30 seconds for this run-through, as the timer is replaced by a 30 second timer for testing purposes. The animated green sonicator can be seen in the video of the full run-through, meaning that the *statusSonicator* signal works as intended. Once the timer is done, the PLC will switch to step 4.


TEMPERATURE
Temperature in reactor 35.76
Reactor temp. rate of change 2.8229
Ambient temperature 21.63
Temperature in element 85.32
Power applied to heating element 100.00 %
Resorcinol Concentra 0.00000
Level in reactor (m) 1.25
mTot (kg) 578.67
wOut (kg/min) 0.00
mOut (kg) 0.00

"flowSiO2NPs"	%ID8	Floating-point number	0.0
"positionSiO2NPsValve"	%Q0.4	Bool	☐ FALSE
"statusSonicator"	%Q0.5	Bool	☐ FALSE
// State 4; Heating			
"tempReactor"	%ID12	Floating-point number	36.0
"startTempReg"	%M1.0	Bool	🟩 TRUE
// State 5; Fill Resorcinol and Ammonia			
"flowAm"	%ID24	Floating-point number	0.0
"flowRes"	%ID20	Floating-point number	0.0
"positionAmValve"	%Q1.0	Bool	☐ FALSE
"positionResValve"	%QD2	Floating-point number	0.0
"PLC/HMI Interface".setConcPID		Floating-point number	6.56
"startConcReg"	%M1.1	Bool	☐ FALSE
// State 6; Fill Formaldehyde			
"flowForm"	%ID32	Floating-point number	0.0
"positionFormValve"	%Q0.6	Bool	☐ FALSE
"PLC/HMI Interface".thicknessCoating		DEC+/-	0
"PLC/HMI Interface".getCoatingET		Time	T#0MS
"6_Form_DB".statTimeElapsed		Time	T#0MS
// State 7; Selecting coating thickness			

Figure 8.4.4: A view of SIMIT (left) and the PLC (right) during step 4.

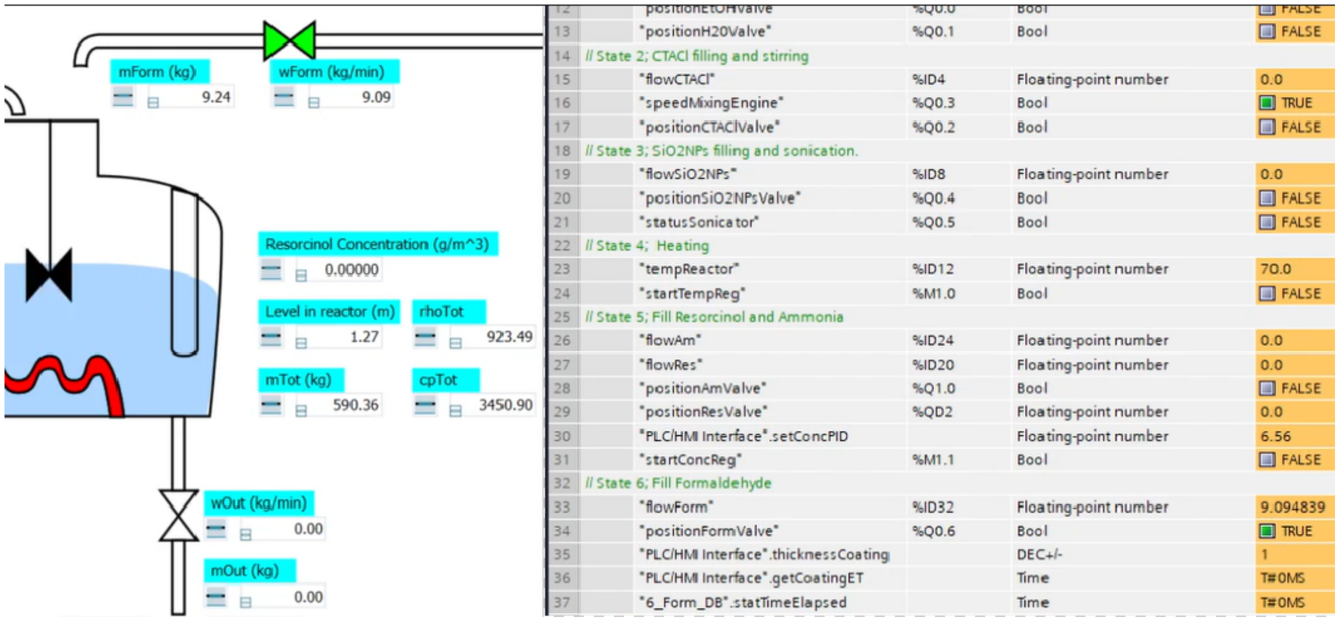
Step 4: Figure 8.4.4 shows step 4 of the process, which is the heating step. As indicated by the illustrated heating element turned red, the simulation has received the control signal with a power value applied to the element. The temperature values and variables relevant to the temperatures are monitored inside the red outline in SIMIT, where *Power applied to heating element* is displayed as a percentage of the full range of power values. The PID made in the TIA Portal controls the temperature using the power signal, and the trigger for the next step is when the temperature reaches 70 °C. The figure also shows some other monitored essential parameters for the process, like $mTot$ and *Level in the reactor*, which is also reflected by the animated blue portion of the illustrated reactor.



1	// State Value			
2	"ProcessSequence_DB".statState	%DB2.DBD4	DEC+/-	5
3	// State 0; Start button, Process on			
4	"PLC/HMI Interface".setCoatingSec		DEC+/-	5
5	"PLC/HMI Interface".setCoatingMin		DEC+/-	1
6	"PLC/HMI Interface".setConcPID		Floating-point number	6.56
7	"PLC/HMI Interface".setTempPID		Floating-point number	70.0
8	"PLC/HMI Interface".setStartButton		Bool	FALSE
9	// State 1; H2O and EtOH Filling.			
10	"flowEtOH"	%ID40	Floating-point number	0.0
11	"flowH2O"	%ID0	Floating-point number	0.0
12	"positionEtOHValve"	%Q0.0	Bool	FALSE
13	"positionH20Valve"	%Q0.1	Bool	FALSE
14	// State 2; CTACl filling and stirring			
15	"flowCTACl"	%ID4	Floating-point number	0.0
16	"speedMixingEngine"	%Q0.3	Bool	🟩 TRUE
17	"positionCTAClValve"	%Q0.2	Bool	FALSE
18	// State 3; SiO2NPs filling and sonication.			
19	"flowSiO2NPs"	%ID8	Floating-point number	0.0
20	"positionSiO2NPsValve"	%Q0.4	Bool	FALSE
21	"statusSonicator"	%Q0.5	Bool	FALSE
22	// State 4; Heating			
23	"tempReactor"	%ID12	Floating-point number	70.0
24	"startTempReg"	%M1.0	Bool	FALSE
25	// State 5; Fill Resorcinol and Ammonia			
26	"flowAm"	%ID24	Floating-point number	2.966722
27	"flowRes"	%ID20	Floating-point number	0.4358197
28	"positionAmValve"	%Q1.0	Bool	🟩 TRUE
29	"positionResValve"	%QD2	Floating-point number	5.48015
30	"PLC/HMI Interface".setConcPID		Floating-point number	6.56
31	"startConcReg"	%M1.1	Bool	🟩 TRUE

Figure 8.4.5: A view of SIMIT (left) and the PLC (right) during step 5.

Step 5: After reaching 70 °C, step 5 is entered, where the valves for ammonia and resorcinol are opened while the heating element and mixer is on. The ammonia valve is open until 2 *kg* has been added, while the resorcinol valve is controlled by a PID in TIA. After the ammonia has been added, a five minute timer is started, which is replaced with a 5 second timer for testing, and which is the trigger for the next step. The relevant regions are shown in Figure 8.4.5. Like the previous filling steps, the valves are displayed as green while open, and the flow rates in SIMIT can be compared to the watch table. The values of temperature and concentration shown in the figure is also an indication of the performance of the PID-controllers.



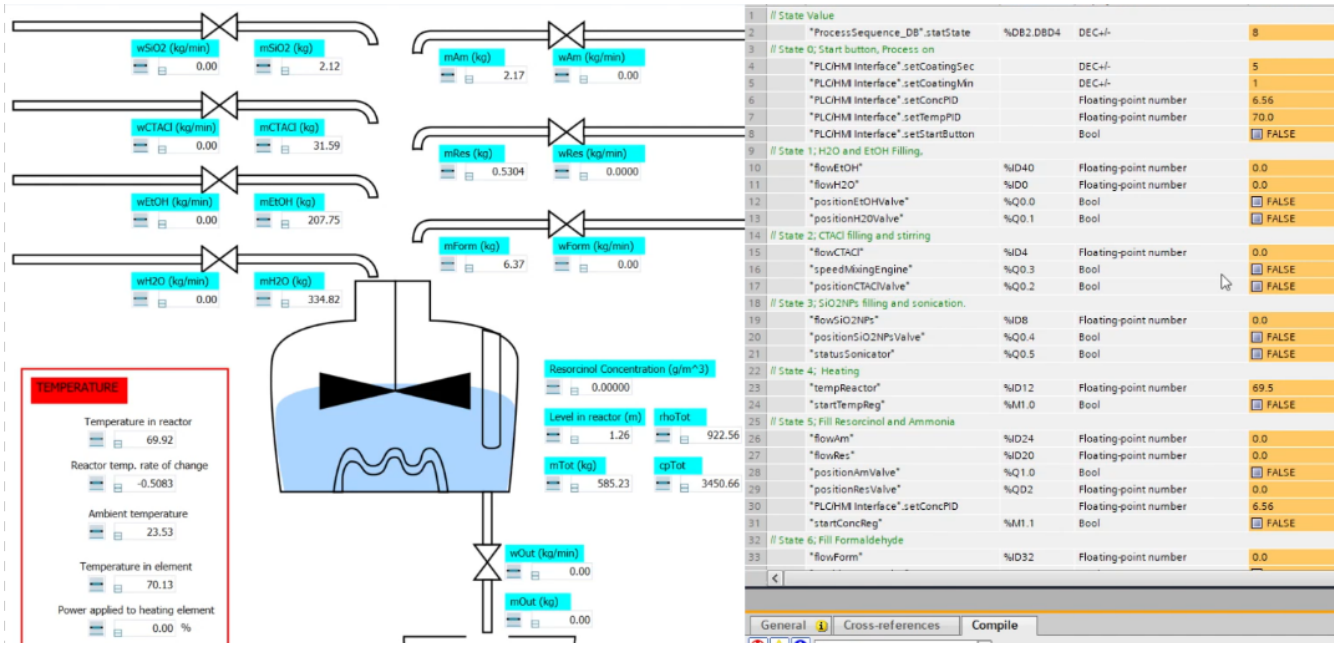
12	"positionEtOHValve"	%Q0.0	Bool	FALSE
13	"positionH20Valve"	%Q0.1	Bool	FALSE
14	// State 2; CTACl filling and stirring			
15	"flowCTACl"	%ID4	Floating-point number	0.0
16	"speedMixingEngine"	%Q0.3	Bool	🟩 TRUE
17	"positionCTAClValve"	%Q0.2	Bool	FALSE
18	// State 3; SiO2NPs filling and sonication.			
19	"flowSiO2NPs"	%ID8	Floating-point number	0.0
20	"positionSiO2NPsValve"	%Q0.4	Bool	FALSE
21	"statusSonicator"	%Q0.5	Bool	FALSE
22	// State 4; Heating			
23	"tempReactor"	%ID12	Floating-point number	70.0
24	"startTempReg"	%M1.0	Bool	FALSE
25	// State 5; Fill Resorcinol and Ammonia			
26	"flowAm"	%ID24	Floating-point number	0.0
27	"flowRes"	%ID20	Floating-point number	0.0
28	"positionAmValve"	%Q1.0	Bool	FALSE
29	"positionResValve"	%QD2	Floating-point number	0.0
30	"PLC/HMI Interface".setConcPID		Floating-point number	6.56
31	"startConcReg"	%M1.1	Bool	FALSE
32	// State 6; Fill Formaldehyde			
33	"flowForm"	%ID32	Floating-point number	9.094839
34	"positionFormValve"	%Q0.6	Bool	🟩 TRUE
35	"PLC/HMI Interface".thicknessCoating		DEC+/-	1
36	"PLC/HMI Interface".getCoatingET		Time	T#0MS
37	"6_Form_DB".statTimeElapsed		Time	T#0MS

Figure 8.4.6: A view of SIMIT (left) and the PLC (right) during step 6.

Step 6: When the timer from step 5 has stopped, step 6 is entered, where formaldehyde is added while the mixer is still active, and the resorcinol valve and heating element is still regulated. The control signal for the formaldehyde valve also works as intended, as seen by the green valve, affected mass calculation *mForm* in SIMIT, and the flow rate which is identical in SIMIT and the watch table.

The duration of this step is dependent on the coating time from the user input on the HMI screen. If 60 minutes and 0 seconds is entered, the step will last for a full hour. For the testing, only just above a minute was entered, and the coating length was verified by watching the active timer in the online view of the FB 6_*Form*. Step 6 will be run at least two times, and possibly more if so specified on the HMI in step 7. The last time formaldehyde is added, the resorcinol valve is closed before the formaldehyde valve is opened. A one hour timer is started after the formaldehyde valve is closed for the last time, which has been set to one minute for the testing, and will trigger the switch to step 8.

Step 7: The first time step 6 is finished, it will switch to step 7, where the user has to specify the coating thickness. For the testing, the thickness was set to 1, which means that step 5 will not be run more than once, and step 6 will only be run twice. After receiving the desired thickness, the step changed accordingly.



wSiO2 (kg/min)	0.00
mSiO2 (kg)	2.12
wCTACl (kg/min)	0.00
mCTACl (kg)	31.59
wEtOH (kg/min)	0.00
mEtOH (kg)	207.75
wH2O (kg/min)	0.00
mH2O (kg)	334.82
mAm (kg)	2.17
wAm (kg/min)	0.00
mRes (kg)	0.5304
wRes (kg/min)	0.0000
mForm (kg)	6.37
wForm (kg/min)	0.00
Resorcinol Concentration (g/m^3)	0.00000
Level in reactor (m)	1.26
rhoTot	922.56
mTot (kg)	585.23
cpTot	3450.66
wOut (kg/min)	0.00
mOut (kg)	0.00

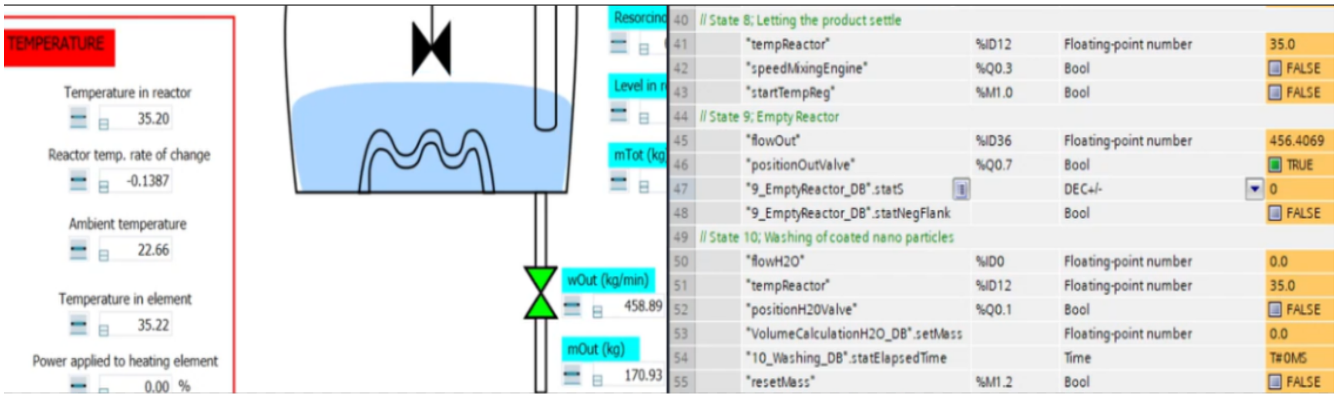
TEMPERATURE

Temperature in reactor	69.92
Reactor temp. rate of change	-0.5083
Ambient temperature	23.53
Temperature in element	70.13
Power applied to heating element	0.00 %

// State Value			
"ProcessSequence_DB".statState	%DB2.DBD4	DEC+/-	8
// State 0; Start button, Process on			
"PLC/HMI Interface".setCoatingSec		DEC+/-	5
"PLC/HMI Interface".setCoatingMin		DEC+/-	1
"PLC/HMI Interface".setConcPID		Floating-point number	6.56
"PLC/HMI Interface".setTempPID		Floating-point number	70.0
"PLC/HMI Interface".setStartButton		Bool	FALSE
// State 1; H2O and EtOH Filling.			
"flowEtOH"	%ID40	Floating-point number	0.0
"flowH2O"	%ID0	Floating-point number	0.0
"positionEtOHValve"	%Q0.0	Bool	FALSE
"positionH20Valve"	%Q0.1	Bool	FALSE
// State 2; CTACl filling and stirring			
"flowCTACl"	%ID4	Floating-point number	0.0
"speedMixingEngine"	%Q0.3	Bool	FALSE
"positionCTAClValve"	%Q0.2	Bool	FALSE
// State 3; SiO2NPs filling and sonication.			
"flowSiO2NPs"	%ID8	Floating-point number	0.0
"positionSiO2NPsValve"	%Q0.4	Bool	FALSE
"statusSonicator"	%Q0.5	Bool	FALSE
// State 4; Heating			
"tempReactor"	%ID12	Floating-point number	69.5
"startTempReg"	%M1.0	Bool	FALSE
// State 5; Fill Resorcinol and Ammonia			
"flowAm"	%ID24	Floating-point number	0.0
"flowRes"	%ID20	Floating-point number	0.0
"positionAmValve"	%Q1.0	Bool	FALSE
"positionResValve"	%QD2	Floating-point number	0.0
"PLC/HMI Interface".setConcPID		Floating-point number	6.56
"startConcReg"	%M1.1	Bool	FALSE
// State 6; Fill Formaldehyde			
"flowForm"	%ID32	Floating-point number	0.0

Figure 8.4.7: A view of SIMIT (left) and the PLC (right) during step 8.

Step 8: This step is the settling of the solution, where the heating element and and mixing engine is turned off, verified by the animations in SIMIT becoming inactive. The PLC also sends a control signal called *turnSolid*, which will let SIMIT treat a specific amount of the mass as a solid mass, based on the amount of silicon nanoparticles, resorcinol, and formaldehyde, as well as the yield calculated from the coating phase. Treating a part of the mass as solid is vital for the next step. Step 8 has a trigger based on a timer and the temperature in the reactor. The timer is for 10 hours, but has been set to an arbitrarily chosen low number during testing to essentially let the temperature be the trigger for the next step.

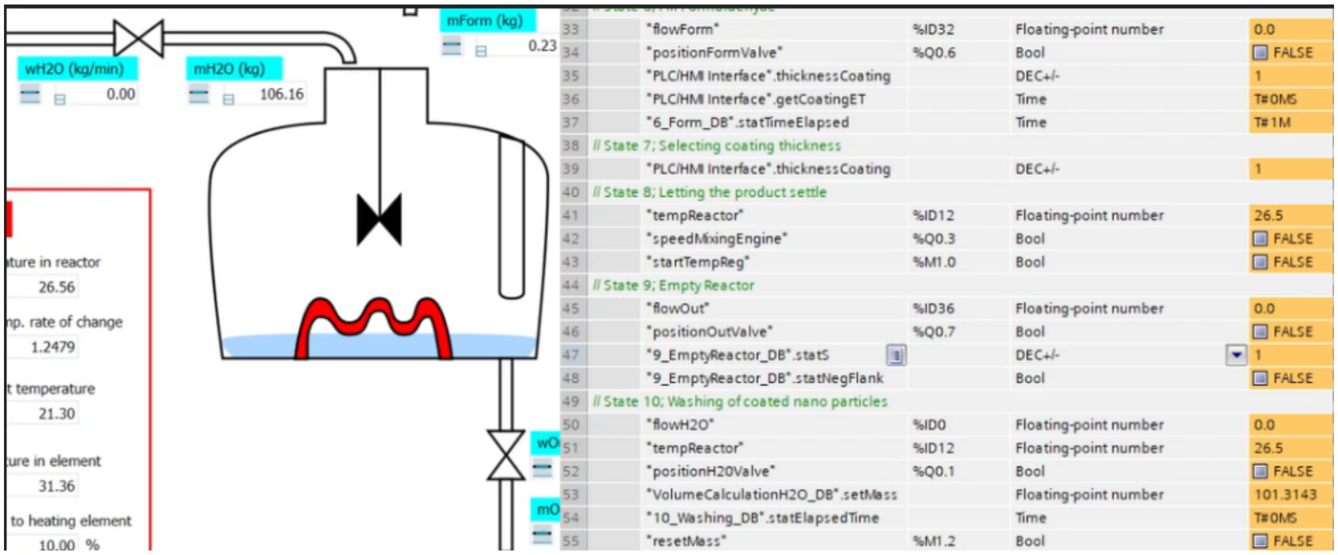


40	// State 8; Letting the product settle			
41	"tempReactor"	%ID12	Floating-point number	35.0
42	"speedMixingEngine"	%Q0.3	Bool	FALSE
43	"startTempReg"	%M1.0	Bool	FALSE
44	// State 9; Empty Reactor			
45	"flowOut"	%ID36	Floating-point number	456.4069
46	"positionOutValve"	%Q0.7	Bool	TRUE
47	"9_EmptyReactor_DB".statS		DEC+/-	0
48	"9_EmptyReactor_DB".statNegFlank		Bool	FALSE
49	// State 10; Washing of coated nano particles			
50	"flowH2O"	%ID0	Floating-point number	0.0
51	"tempReactor"	%ID12	Floating-point number	35.0
52	"positionH20Valve"	%Q0.1	Bool	FALSE
53	"VolumeCalculationH2O_DB".setMass		Floating-point number	0.0
54	"10_Washing_DB".statElapsedTime		Time	T#0MS
55	"resetMass"	%M1.2	Bool	FALSE

Figure 8.4.8: A view of SIMIT (left) and the PLC (right) during step 9.

Step 9: After the reactor content has settled in step 8, it is time to empty the liquid. The emptying step will be run twice, once after the settling step, and once after the washing step to empty the water used for washing. When step 9 is entered, the valve beneath the reactor is opened, as seen by the green valve in Figure 8.4.8. The figure also shows that the mass $mOut$ has increased as a result of opening the valve. The step will switch to either 10 or 11 when the emptying is done, based on whether the previous step was 8 or 10. This was verified by checking if the washing water was only added once, which it was.

Step 10: After the supernatant is emptied in step 9, step 10 is entered, which is when the washing water is added and heated. Upon entering step 10, the H_2O valve is opened as seen by the mass of H_2O exceeding 100 kg in Figure 8.4.9, which is approximately the desired amount. Afterwards, the heating element is supplied with a sufficient amount of power able to keep the water boiling. A two hour timer is started once the water has reached boiling temperature, and this timer is also shortened for testing purposes. As seen in the figure, the timer has not been started yet because the heating element has just received a power signal. When the timer is done, the power is set to 0 and the step is changed back to step 9, which was verified by the valve opening and the level and mass in the reactor dropping.

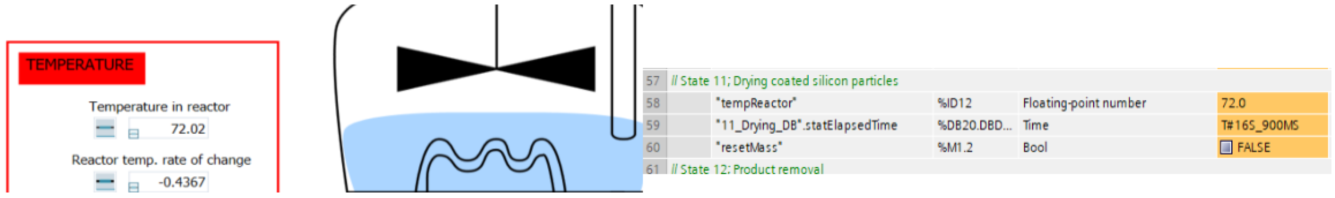


33	"flowForm"	%ID32	Floating-point number	0.0
34	"positionFormValve"	%Q0.6	Bool	FALSE
35	"PLC/HMI Interface".thicknessCoating		DEC+/-	1
36	"PLC/HMI Interface".getCoatingET		Time	T#0MS
37	"6_Form_DB".statTimeElapsed		Time	T#1M
38	// State 7; Selecting coating thickness			
39	"PLC/HMI Interface".thicknessCoating		DEC+/-	1
40	// State 8; Letting the product settle			
41	"tempReactor"	%ID12	Floating-point number	26.5
42	"speedMixingEngine"	%Q0.3	Bool	FALSE
43	"startTempReg"	%M1.0	Bool	FALSE
44	// State 9; Empty Reactor			
45	"flowOut"	%ID36	Floating-point number	0.0
46	"positionOutValve"	%Q0.7	Bool	FALSE
47	"9_EmptyReactor_DB".statS		DEC+/-	1
48	"9_EmptyReactor_DB".statNegFlank		Bool	FALSE
49	// State 10; Washing of coated nano particles			
50	"flowH2O"	%ID0	Floating-point number	0.0
51	"tempReactor"	%ID12	Floating-point number	26.5
52	"positionH20Valve"	%Q0.1	Bool	FALSE
53	"VolumeCalculationH2O_DB".setMass		Floating-point number	101.3143
54	"10_Washing_DB".statElapsedTime		Time	T#0MS
55	"resetMass"	%M1.2	Bool	FALSE

Figure 8.4.9: A view of SIMIT (left) and the PLC (right) during step 10.

Step 11: When the washing water from step 10 has been emptied in step 9, the drying in step 11 is initiated. A two hour timer is started when the reactor temperature is above 70 °C, this timer was monitored to verify the functionality, and can be seen counting after the temperature has turned 70 °C.

After the timer ran out, the heating element was turned off and the step switched to 12.



TEMPERATURE

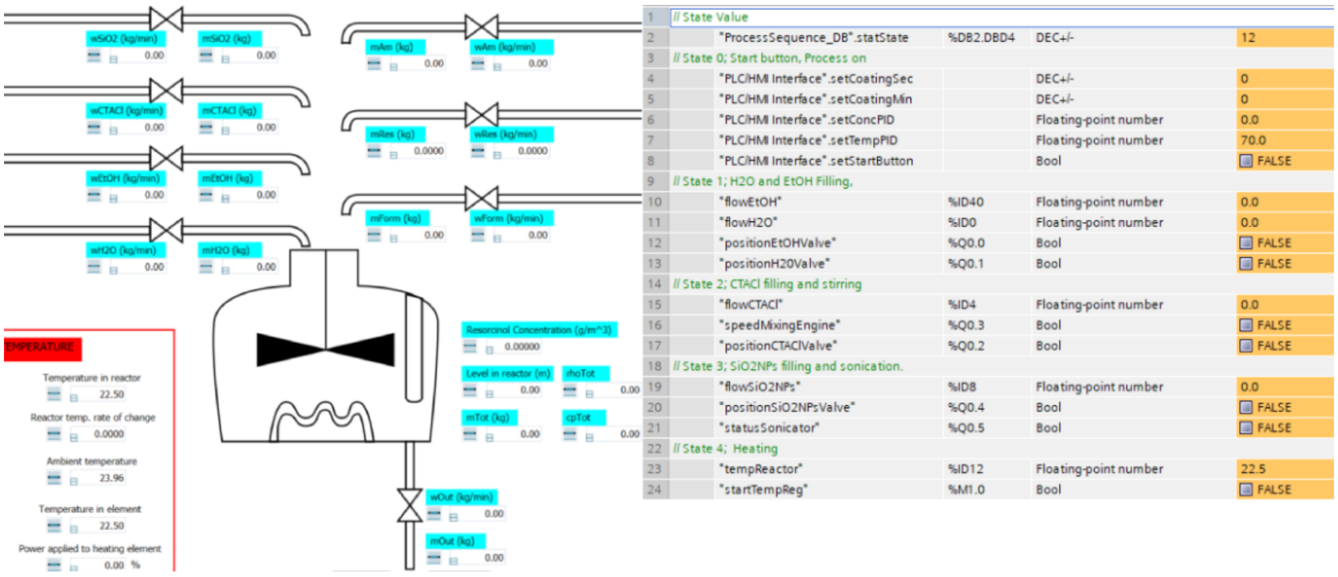
Temperature in reactor 72.02

Reactor temp. rate of change -0.4367

57	// State 11; Drying coated silicon particles			
58	"tempReactor"	%ID12	Floating-point number	72.0
59	"11_Drying_DB".statElapsedTime	%DB20.DBD...	Time	T#16S_900MS
60	"resetMass"	%M1.2	Bool	FALSE
61	// State 12: Product removal			

Figure 8.4.10: A view of SIMIT (left) and the PLC (right) during step 11.

Step 12: Step 12 is the final step. Its main functionalities is to firstly calculate the process yield, and secondly reset all variables in the PLC, as well as send a control signal that resets all variables in SIMIT. When the reset button on the HMI is pressed, every variable is reset, as can be seen in Figure 8.4.11.



wSiO2 (kg/min) 0.00
mSiO2 (kg) 0.00
wCTACl (kg/min) 0.00
mCTACl (kg) 0.00
wEtOH (kg/min) 0.00
mEtOH (kg) 0.00
wH2O (kg/min) 0.00
mH2O (kg) 0.00
mAm (kg) 0.00
wAm (kg/min) 0.00
mRes (kg) 0.0000
wRes (kg/min) 0.0000
mForm (kg) 0.00
wForm (kg/min) 0.00

EMPERATURE

Temperature in reactor 22.50

Reactor temp. rate of change 0.0000

Ambient temperature 23.96

Temperature in element 22.50

Power applied to heating element 0.00 %

Resorcinol Concentration (g/m^3) 0.00000
Level in reactor (m) 0.00
rhoTot 0.00
mTot (kg) 0.00
cpTot 0.00
wOut (kg/min) 0.00
mOut (kg) 0.00

1	// State Value			
2	"ProcessSequence_DB".statState	%DB2.DBD4	DEC+/-	12
3	// State 0; Start button, Process on			
4	"PLC/HMI Interface".setCoatingSec		DEC+/-	0
5	"PLC/HMI Interface".setCoatingMin		DEC+/-	0
6	"PLC/HMI Interface".setConcPID		Floating-point number	0.0
7	"PLC/HMI Interface".setTempPID		Floating-point number	70.0
8	"PLC/HMI Interface".setStartButton		Bool	FALSE
9	// State 1; H2O and EtOH Filling.			
10	"flowEtOH"	%ID40	Floating-point number	0.0
11	"flowH2O"	%ID0	Floating-point number	0.0
12	"positionEtOHValve"	%Q0.0	Bool	FALSE
13	"positionH20Valve"	%Q0.1	Bool	FALSE
14	// State 2; CTACl filling and stirring			
15	"flowCTACl"	%ID4	Floating-point number	0.0
16	"speedMixingEngine"	%Q0.3	Bool	FALSE
17	"positionCTAClValve"	%Q0.2	Bool	FALSE
18	// State 3; SiO2NPs filling and sonication.			
19	"flowSiO2NPs"	%ID8	Floating-point number	0.0
20	"positionSiO2NPsValve"	%Q0.4	Bool	FALSE
21	"statusSonicator"	%Q0.5	Bool	FALSE
22	// State 4; Heating			
23	"tempReactor"	%ID12	Floating-point number	22.5
24	"startTempReg"	%M1.0	Bool	FALSE

Figure 8.4.11: A view of SIMIT (left) and the PLC (right) during step 12.

Chapter 9

Conclusion and Future Work

This chapter describes the conclusion of this thesis, and what work that further should be done in this project, based on what was learned, what was not prioritised due to time constraints or what was not completed due to other circumstances.

9.1 Conclusion

Making a state diagram of the process turned out to be a great visual aid while programming the logic of the control system. The mathematical model had to be developed with a combination of many different laws of physics and chemistry for the full modeled system to properly describe the behavior of the coating process.

The process behavior was successfully implemented in the simulation software SIMIT. This was verified by comparing values of variables in SIMIT to their corresponding mathematically calculated values. The simulated process proved to be a valuable resource for testing the functionality of the PLC in a realistic environment, rather than only testing whether certain lines of code work independently. The testing of the control system on a digital twin of the process is what is known as virtual commissioning, which enables a thorough inspection of the performance under secure circumstances. This can lead to the discovery of critical bugs that could have been disastrous for the real physical system. The control system in this thesis was no different, and several bugs were found and fixed prior to the results presented in this report, which would cause unacceptable behavior if commissioned on the real system.

Regarding temperature regulation, the transfer functions based on the state-space model was sufficient for tuning the PID-controllers. One controller is used while resorcinol is the only compound flowing into the reactor, and one controller is used when either ammonia or formaldehyde is added in bulk, because these two cases have different temperature behavior. The PLC has been programmed to switch between the two controllers when it sends the control signals to open or close the valves of ammonia and formaldehyde. This functionality was verified, and the choice provided a satisfactory regulated temperature, even with the generated noise in SIMIT which introduced disturbances that the controllers had to account for.

As for the concentration regulation, the transfer functions based on the state-space model ended up in a form that could not be tuned the same way as the temperature functions. The reason is due to the fact that while the temperature behavior has certain dependencies, the concentration behavior depends on the temperature in addition to other parameters, which makes it inherit the temperature dependencies. This yields a more complex system of higher order than the Skogestad method can handle, which is why the model reduction technique was used. However, the reduced model yielded only unstable results from the PID-tuning. For this reason, the PID-tuning method integrated in the TIA Portal was used, which provided satisfactory results for the regulated concentration.

It is important to notice that even though the regulation of the parameters works well with the digital twin, it is not given that it will perform as well on the real process, due to the real process having infinite amount of variables that affects its behavior, and the implementation possibilities in a model is finite. But it still gives an idea of how the process behaves.

9.2 Future Work

One essential part that needs to be enhanced is the SIMIT model. More specifically the values for the activation energy, E_a ,and pre-exponential factor ,A. As their values have been based on an arbitrary similar process. When the real upscaled process is available, experiments should be completed to decide these values for the real process. It should also be decided if the process is isothermal, endothermal or exothermal, as in this project, it has been assumed a reaction that is isothermal. An implementation of the real parameter values gathered from the experiments could lead to the discovery of a more eco-friendly process, that is more energy efficient. The SIMIT model could be made even more realistic by implementing delays between the time it takes from the valve is opened, to the sensor actually detects that a flow is present. This is something that would affect the control of all the individual flows, and the regulation of the resorcinol concentration. Another improvement is to compensating for the opening time of the valve. The as is SIMIT model, overshoots the added mass, above the desired mass value, with approximately 5 %. Which, in a process like this, can have a negative effect on the yield. This needs to be compensated for in the PLC control. It should also be assesed if any of the waste from this process can be utilized in other part of the battery cell production, for example the heated water used for washing in step $10\_Washing$, could be used to heating in another part of the process.

The yield model itself should also be looked at, as with the Arrhenius equation parameters for reaction rate. The yield model, this project bases it self on, was chosen due to the real process was unavailable. Therefore, an arbitrary yield model from a similar process was used. This model should therefore be substituted with a model that represents the real process. A suggestion would be to set up a number of IoT sensors, on the real process to collect the relevant data for the process that affects the yield, and record a number of run-troughs. With a connection to Siemens Cloud-based IoT operating system, MindSphere, this data could be used to make a surrogate model of the real process. A scientific paper that describes the use of surrogate modeling and how it could be used in this manner is [11]. In the scientific paper a surrogate model optimization technique is also described which could be used to continuously better the model and make it more accurate. There are a number of surrogate modeling toolboxes for both Matlab and Python which could be used to create the model. A link one for Python can be found here.

The HMI developed in this project, is a very simplistic HMI, it was only intended to give an overall overview of the process and basic control and manipulation capabilities. It is suggested that the HMI functionality should be expanded. A Monitoring screen similar to the *Overview* in SIMIT could be added to give the operator better insight in the processes operating state and performance, thus giving him the tools to take better decisions. Trend charts with, limit areas for temperature and concentration could also be a good addition. An emergency stop function should also be implemented, in combination with an alarm structure, that would warn the operator when the process is near a state of failure or is in danger of destroying the batch.

Bibliography

[1] Beyonder. Game-changing technology, 2019. URL https://www.beyonder.no/technology.

[2] Beyonder. Beyonder and siemens partner to produce the next generation green battery cells, 2020. URL https://www.beyonder.no/news/press-release-beyonder-and-siemens-partner-to-produce-next-generation-green-battery-cells.

[3] Durham College. Thermodynamics – basic concepts, 2011. URL https://durhamcollege.ca/wp-content/uploads/Thermodynamics-Basic-Concepts.pdf.

[4] EBA. European battery alliance, 2018. URL https://ec.europa.eu/growth/industry/policy/european-battery-alliance_en.

[5] Engineers edge. Overall heat transfer coefficient table and equation, Unknown. URL https://www.engineersedge.com/thermodynamics/overall_heat_transfer-table.htm.

[6] Engineers Edge. Convective heat transfer convection equation and calculator, unknown. URL https://www.engineersedge.com/heat_transfer/convection.htm.

[7] Finn Haugen. Model-based pid tuning with skogestad‘s method, 2009. URL https://www.mic-journal.no/PDF/ref/Haugen2009.pdf.

[8] Herman Boysen. kv: what, why, how, whence?, 2011. URL https://web.archive.org/web/20160809184955/http://heating.danfoss.com/PCMPDF/VFHBG202_kv.pdf.

[9] Ilgook Kim, Jong-In Han. Optimization of alkaline pretreatment conditions for enhancing glucose yield of rice straw by response surface methodology, 2012. URL https://www.sciencedirect.com/science/article/pii/S0961953412003388.

[10] Industrial heating systems. How to calculate heat loss?, Unknown. URL https://industrialheatingsystems.com/How-calculate-heat-loss.html.

[11] S. H. Kim and F. Boukouvala. Machine learning-based surrogate modeling for data-driven optimization: a comparison of subset selection for regression techniques. *Optimization Letters*, pages 1–22, 2019. URL https://link.springer.com/content/pdf/10.1007/s11590-019-01428-7.pdf.

[12] Natalia Lebedeva, Franco Di Persio and Lois Boon-Brett. Lithium ion battery value chain and related opportunities for europe, 2016. URL https://ec.europa.eu/jrc/sites/jrcsh/files/jrc105010_161214_li-ion_battery_value_chain_jrc105010.pdf.

[13] Peter Woolf. Valves - modeling dynamics, 2021. URL https://eng.libretexts.org/Bookshelves/Industrial_and_Systems_Engineering/Book%3A_Chemical_Process_Dynamics_and_Controls_(Woolf)/03%3A_Sensors_and_Actuators/3.10%3A_Valves_-_Modeling_Dynamics.

[14] Samson Group. Series v2001 valves · type 3321 globe valves with electropneumatic, pneumatic or electric actuators, 2020. URL https://www.samsongroup.com/document/t81110en.pdf.

[15] Siemens AG. Simatic s7-1200, s7-1500 pid control function manual, 2016. URL https://cache.industry.siemens.com/dl/files/036/108210036/att_896030/v1/s71500_pid_control_function_manual_enUS_en-US.pdf#page214.

[16] Siemens AG. Programming guideline for s7-1200/1500, 2018. URL https://cache.industry.siemens.com/dl/files/040/90885040/att_970576/v1/81318674_Programming_guideline_DOC_v16_en.pdf.

[17] Siemens AG. Programming style guide for simatic s7-1200/ s7-1500, 2020. URL https://cache.industry.siemens.com/dl/files/084/109478084/att_1022099/v1/81318674_Programming_Styleguide_DOC_v20_en.pdf.

[18] Siemens AS. Creating the better battery", 2020. URL https://new.siemens.com/global/en/markets/battery-manufacturing.html.

[19] Siemens AS. Batterifabrikken", 2020. URL https://new.siemens.com/no/no/siemens-i-norge/nyheter/industri/batterifabrikken.html.

[20] Sigurd Skogestad. Process dynamics, 2009. URL https://folk.ntnu.no/skoge/prosessregulering/course-material/Skogestad-Ch11.pdf.

[21] S. Skogestad. Simple analytic rules for model reduction and pid controller tuning. *Journal of process control*, 13(4):291–309, 2003. URL https://www.sciencedirect.com/science/article/pii/S0959152402000628?casa_token=6jMXM8E3jXIAAAAA:8ly7Gj2mYH5GI8628Q4vehTWtp0QT3SLS5jwcpTyFHmPu2R7Jl-cYY0fxvSLmwerVqs1fj5wjSU.

[22] Special Metals. Ncoloy® alloy 800h (uns n08810), 2003. URL https://www.haraldpihl.com/globalassets/pdf/069_incoloy-alloy-800.pdf.

[23] Syed Z. Rizvi, Javad Mohammadpour, Roland Toth and Nader Meskin. A kernel-based approach to mimo lpv state-space identification and application to a nonlinear process system, 2015. URL https://rolandtoth.eu/wp-content/uploads/2018/07/LPVS2015c.pdf.

[24] Thermtest instruments. Introducing the rule of mixtures calculator, Unknown. URL https://thermtest.com/rule-of-mixtures-calculator.

[25] United Nations. Key aspects of the paris agreement, 2015. URL https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement/key-aspects-of-the-paris-agreement.

[26] Verdi Ogewell. Northvolt, siemens and "a goddamn good digital twin", 2020. URL https://www.engineering.com/story/northvolt-siemens-and-a-goddamn-good-digital-twin.

[27] Watlow. Immersion heaters, unknown. URL https://www.watlow.com/-/media/documents/catalogs/immersion-heaters0618.ashx.

[28] World Economic Forum. A vision for a sustainable battery value chain in 2030, 2019. URL http://www3.weforum.org/docs/WEF_A_Vision_for_a_Sustainable_Battery_Value_Chain_in_2030_Report.pdf.

Appendices

Appendix A

Adjustment of Yield Characteristics

It was wanted to find the relation between the parameters temperature, resorcinol concentration, and time to yield. Since it was not possible to collect data from the process to gain this knowledge, it was decided to find an arbitrary process that could have a similar behavior and represent the relation. The temperature, resorcinol concentration, and time are the main parameters affecting the yield, and for this reason, there was a need to find the relation between these parameters and the yield. Since collecting data from the process could not be done, it was decided to find an arbitrary process that could have similar behavior and represent the relation. The scientific paper [9] was found with the relation between the relevant parameters and yield. This appendix will explain how this relation was adjusted to better suit the process in question. There are some known steady-state parameter values for the process under consideration in this project. At $T_{reac} = 70°\mathrm{C}$, $t = 60$ min, and with 2 g of resorcinol added during the coating phase, the final result is one coating layer of 100 nm with a yield of 86 %. Since it is wished to investigate the effects on a future upscaled pilot process line, the mass of resorcinol will be multiplied by 1000, hence becoming 2 kg. These parameter values and the corresponding yield will, from this point on, be called the baseline for the coating process.

The first step is to calculate the volumetric flow that equals to adding 2 kg of resorcinol over 60 minutes. Since the external resorcinol tank consists of a resorcinol solution, the concentration in this solution needs to be calculated. The relation between the solution and resorcinol is $\frac{80}{3}$ L of 60/40 H_2O/EtOH mixture per 2 kg resorcinol giving a resorcinol concentration in the external tank of:

$$C_{res,in} = \frac{m_{res}}{V_{H2O} + V_{EtOH} + V_{res}} = \frac{2 \cdot 1000}{\frac{80 \cdot 0.6}{3} \cdot 10^{-3} + \frac{80 \cdot 0.4}{3} \cdot 10^{-3} + \frac{2}{1280}} = 70850 \quad \left[\frac{g}{m^3}\right] \tag{A.1}$$

The total volume wanted to add to the reactor when wanting to add 2 kg of resorcinol over sixty minutes is the following:

$$q_{res} = \frac{V_{res,solution}}{60} = \frac{\frac{80 \cdot 0.6}{3} \cdot 10^{-3} + \frac{80 \cdot 0.4}{3} \cdot 10^{-3} + \frac{2}{1280}}{60} = 4.705 \cdot 10^{-4} \quad \left[\frac{m^3}{min}\right] \tag{A.2}$$

To simplify the calculations it is assumed that no other compounds will be added while the resorcinol is added when the concentration is at a steady-state. This is expressed mathematically as:

$$\sum_{i=1}^{7} q_i(t) = q_{res}(t) = 4.705 \cdot 10^{-4} \qquad \left[\frac{m^3}{min}\right] \tag{A.3}$$

Equation 3.49 is used to find the resorcinol concentration, under the assumption that the concentration is at a steady-state, hence $\frac{dC_r(t)}{dt} = 0$. The effect of the volume increase is neglected due to the fact that the change is very small when only resorcinol is added, meaning that $V(t) = V$. The formula can be manipulated to find the concentration as such:

$$C_{res}(t) = \frac{C_{res,in}}{1 + \left(k(t) \cdot \frac{V}{q_{res}(t)}\right)} \quad \left[\frac{g}{m^3}\right] \tag{A.4}$$

Due to the fact that it was not possible to complete experiments on the coating process, the values for E_a and A need to be assumed. These values used in this project are taken from [23], which states that $E_a =$ 30 000 (J/kg) and $A = 25\ min^{-1}$. Even though A is defined in seconds in the paper, the same value will be used in minutes to get a response that is not to quick. The temperature in the reactor is known to be $T_{reac} = 70°$C but needs to be converted into Kelvin. This is done by adding 273.15 to 70. The universal gas constant R is equal to 8.314 $J/mol \cdot K$. In this scenario, the individual gas constant for resorcinol specifically is the value in interest. The methodology to compute this is described in [3]. It states that the individual gas constant can be computed by dividing the universal gas constant by the density of the compound, in this case, resorcinol. This will also switch the units into $J/(kg \cdot K)$, which is desired because the calculations for this process are based on kg contrary to mol.

$$R = \frac{8.314\ J}{mol \cdot K} \cdot \frac{1\ mol}{0.1101\ kg} = \frac{75.5\ J}{kg \cdot K} \quad \left[\frac{J}{kg \cdot K}\right] \tag{A.5}$$

Now that all the parameters for k are defined as a set value, it is possible to calculate it.

$$k = A \cdot e^{\frac{-E_a}{R \cdot (T_{reac}+273.15)}} = 25 \cdot e^{\frac{-30000}{75.5 \cdot (70+273.15)}} \approx 7.853 \qquad [min^{-1}] \tag{A.6}$$

The only missing variable is V, which can be calculated as follows:

$$\begin{aligned} V = \sum_{i=1}^{7} \frac{m_i}{\rho_i} &= V_{H_2O} + V_{EtOH} + V_{res,solution} + \frac{m_{CTACl}}{\rho_{CTACl}} + \frac{m_{SiO_2NPs}}{\rho_{SiO_2NPs}} + \frac{m_{Form}}{\rho_{Form}} + \frac{m_{Ammonia}}{\rho_{Ammonia}} \\ &= \left(0.32\ + 0.25\ + \left(\frac{80 \cdot 0.6}{3} \cdot 10^{-3} + \frac{80 \cdot 0.4}{3} \cdot 10^{-3} + \frac{2}{1280}\right) + \frac{30}{878} + \frac{2}{2180} + \frac{12}{1090} + \frac{2}{903}\right) \\ &= 0.647 \qquad [m^3] \end{aligned} \tag{A.7}$$

By inserting the previous values in equation A.4, the resorcinol concentration can be calculated as follows:

$$C_{res} = \frac{70850\ [\frac{g}{m^3}]}{1 + (7.853\ [\frac{1}{min}] \cdot \frac{0.647 [m^3]}{4.705 \cdot 10^{-4}\ [\frac{m^3}{min}]})} = 6.56 \qquad \left[\frac{g}{m^3}\right] \tag{A.8}$$

Now that the base values for the coating process regarding time, temperature, and concentration have been derived, only some of the ranges needs adjustment to better suit the process. There is no need to adjust the ranges for the time and the temperature as they are in reasonable intervals. However, the yield needs both adjustments of range and unit notation, and the resorcinol concentration needs adjustments of the range. A sensible range for the yield is [70,100] %, as the available experimental data lies in this interval. To convert the current unit $\frac{g}{kg}$ to percent and change the range from [215,265] to the one mentioned earlier, the following is done:

$$\Delta x = \frac{\Delta\%}{\Delta \frac{g}{kg}} = \frac{100 - 70}{265 - 215} = 0.6 \tag{A.9}$$

where Δx represents the conversion factor from $\frac{g}{kg}$ to %, meaning $Y_{glucose}$ can be replaced by the general equation:

$$Y_{glucose} = \frac{(Y_{CNP} - 70)}{0.6} + 215 \tag{A.10}$$

The yield value of 86 % can now be converted to $\frac{g}{kg}$:

$$x = \frac{86 - 70}{0.6} + 215 = 241.67 \quad \left[\frac{g}{kg}\right] \tag{A.11}$$

Now it is possible to check which concentration value relates to the base values of the yield, temperature, and time. $Y = 241.67$, $T_{reac} = 70°C$ and t = 60 min. These values are put into the Equation 3.50 to calculate, $\overline{C}$, the average resorcinol concentration.

$$\begin{aligned} 241.67 = &-120.384 + 38.687 \cdot \overline{C} + 6.216 \cdot 70 + 2.329 \cdot 60 - 0.13 \cdot \overline{C} \cdot 70 - 0.06 \cdot \overline{C} \cdot 60 \\ &- 2.208 \cdot 10^{-3} \cdot 70 \cdot 60 - 3.657 \cdot \overline{C}^2 - 0.0349 \cdot 70^2 - 0.0174 \cdot 60^2 \end{aligned} \tag{A.12}$$

$$3.657 \cdot \overline{C}^2 - 25.987 \cdot \overline{C} + 30.1176 = 0 \tag{A.13}$$

The solution to A.13 is given by:

$$\overline{C} = \frac{25.987 \pm \sqrt{(-25.987)^2 - 4 \cdot 3.657 \cdot 30.1176}}{2 \cdot 3.657} \tag{A.14}$$

$$\overline{C}_1 = 5.647, \quad \overline{C}_2 = 1.458 \tag{A.15}$$

since $\overline{C}_1$ is outside of the model range [1,4] in [9], $\overline{C}_2$ is deemed the best candidate. $\overline{C}_2$ represents the value in the paper based model that the concentration value will be fitted to. Since the range for the resorcinol concentration is unknown, the range from the paper will be scaled down to fit our value as such:

$$x_C = \frac{C_{res}}{C_{paper}} = \frac{6.56}{1.458} = 4.5 \tag{A.16}$$

the new range for the resorcinol concentration then becomes: $[1 \cdot 4.5,\ 4 \cdot 4.5] = [4.5, 18]$.

To conclude, the new ranges, units, and relation between the papers values are the following: Temperature and time are unchanged, yield is in %, ranges in [70,100] and has the relation $Y_{glucose} = \frac{(Y_{CNP}-70)}{0.6} + 215$, and concentration is in $\frac{g}{m^3}$, ranges between [4.5, 18] and has the relation $C_{NaOH} = \frac{1}{4.5} \cdot C_{res}$. The adjustments are summarised in Table A.0.1 shown below.

Yield Model Ranges and units		
Parameters	Original	Adjusted
Yield	[216, 255] $\left[\frac{g}{kg}\right]$	[70, 100] [%]
Concentration	[1, 4] [%]	[4.5, 18] $\left[\frac{g}{m^3}\right]$
Temperature	[60, 100] [$°C$]	[60, 100] [$°C$]
Time	[30, 90] [min]	[30, 90] [min]

Table A.0.1: Adjusted ranges and unit for the yield model

Appendix B

Design of Process Components

To calculate the different feedback values to the PLC system, there is a need to assume different constants that affect the process dynamics, like K_v, h_h, and A_h. These constants vary depending on the specific type and model of valve and heating element. Therefore, it was decided to calculate the requirements of the different process components, and then select the specific components accordingly.

B.1 Valves

The constant of interest for the valves is K_v, which is the valve coefficient. K_v states the valve's capacity and is defined as the maximum volume of water that can flow through the valve in one hour at a pressure drop of 1 bar. Consequently, the value of K_v should be decided according to the maximum capacity of volume that is needed to pass through the valve per hour.

To calculate K_v, Equation 3.1 is used:

$$K_v = q(t) \cdot \sqrt{\frac{SG}{\Delta P}}, \quad \left[\frac{m^3}{h \cdot bar}\right] \tag{B.1}$$

Since ΔP in most operating phases is not equal to one, the value needs to be calculated to get the correct K_v corresponding to the desired time to add a compound. The equation for ΔP is the following:

$$\Delta P = \frac{\rho \cdot g \cdot h}{100000} \tag{B.2}$$

By replacing ΔP in Equation B.1 with Equation B.2, adding the index i for a compound specific K_v, and defining $q(t)$ as the maximum volumetric flow wanted, it results in the following:

$$K_{v,i} = q_{max,i} \cdot \sqrt{\frac{SG_i \cdot 100000}{\rho_i \cdot g \cdot h_i}} \tag{B.3}$$

Valve requirements: For this process, it is preferred that every compound added without a controlled flow rate has a high enough K_v to ensure the total amount of compound can be added in 1 minute or less. The valve is also needed to; be electric for the PLC to control it directly, follow metric units, and have a linear plug.

EtOH: To design the valve for the EtOH external tank, the maximum flow that the valve can deliver related to the total amount of EtOH added should be the basis for the valve coefficient $K_{v,EtOH}$. The EtOH valve is only opened during step 1. In this step, a total of 250 L of EtOH is added, which is equal to 0.25 m^3. To add this over 1 minute, the flow needs to be the following:

$$q_{max,EtOH} = 0.25 \cdot 1 \cdot 60 = 15 \qquad \left[\frac{m^3}{h}\right]$$

with $SG_{EtOH} = 0.79$, a constant $h_{EtOH} = 1$ and $\rho_{EtOH} = 789.4$, the desired $K_{v,EtOH}$ value can be calculated as such:

$$K_{v,EtOH} = q_{max,EtOH} \cdot \sqrt{\frac{SG_{EtOH} \cdot 100000}{\rho_{EtOH} \cdot g \cdot h_{EtOH}}} = 15 \cdot \sqrt{\frac{0.79 \cdot 100000}{789.4 \cdot 9.81 \cdot 1}} = 47.909 \qquad \text{(B.4)}$$

Samson's Type 3321 DIN version globe valves with electric actuators [14] complies with the requirements that were made. Due to it being DIN, the dimensions of the valve are in given in mm, it also has a linear globe valve. The valve size with the closest K_v is DN 65 with a K_v of 40. Since the valve is not controlled, but operated with an OPEN/CLOSE signal, under-sizing is not an issue.

This calculation process is equal for the valves for H2O, CTACl, SiO2, Ammonia, Formaldehyde, and Out. Therefore, the calculations will not be shown. The calculated K_v, the chosen valve, and its K_v are shown in Table B.1.1.

Valve design				
Compound	Amount	Calculated Kv	Valve chosen	Valve Kv
EtOH	250 L	47.9	DN 65	40
H2O	320 L	61.36	DN 65	40
CTACl	30 kg	6.55	DN 32	6.3
SiO2	2 kg	0.176	DN 15	0.25
Resorcinol	varying ($\approx$ 2kg)	0.2525	DN 15	1.6
Ammonia	2 kg	0.425	DN 15	0.63
Formaldehyde	6 kg	1.056	DN 15	1.6
Out	0.647 m^3	123.9	DN 80	100

Table B.1.1: Caption

Resorcinol: The valve for resorcinol differs from the other ones because it is a controllable valve, meaning it should be able to give a different range of flow actions than the others. Therefore, it needs to be calculated slightly differently. The highest concentration present in the yield model is 0.3 $\frac{g}{m^3}$, this equals to the following maximum flow rate:

$$q_{res,max} = \frac{k \cdot V \cdot C_{res}}{C_{res,in} - C_{res}} = \frac{471.194 \cdot 0.647 \cdot 0.3}{70.85 \cdot 1000 - 0.3} = 1.29 \cdot 10^{-3} \cdot 60 = 0.0774 \qquad \left[\frac{m^3}{h}\right] \qquad \text{(B.5)}$$

with $SG_{res} = 0.955$, a constant $h_{res} = 1$, and $\rho_{res} = 953.13$, the desired $K_{v,res}$ can be calculated as such:

$$K_{v,res} = q_{res,max} \cdot \sqrt{\frac{SG_{res} \cdot 100000}{\rho_{res} \cdot g \cdot h_{res}}} = 0.0774 \cdot \sqrt{\frac{0.955 \cdot 100000}{953.13 \cdot 9.81 \cdot 1}} = 0.2474 \qquad \text{(B.6)}$$

The valve chosen is the DN 15 with a K_v of 1.6, a valve with a bigger K_v than the calculated need is chosen to enable it to have controllability against disturbances.

B.2 Tank Reactor

The design of the tank reactor's dimensions affects the solution's level dynamics in the reactor, which then affects the outlet flow rate. The surface area of the reactor affects the heat loss to the surrounding environment. These properties make the reactor design a necessary aspect of modeling the system.

Firstly, the design of the reactor dimensions is decided. The shape of the tank reactor is important as it directly affects both the rate of heat loss due to the "contact" area with the surrounding environment, and the flow rate out of the reactor. The shape that has the smallest perimeter, in relation to the area, is a circle, therefore a cylindrical shape is the ideal shape to minimize the heat loss from the reactor. To avoid dead zones in the reactor where the liquids cannot reach the outlet valve, it is wanted that the bottom part of the reactor has an upside down cone shape, with the valve being placed at the tip of the cone. The height of this cone is so small that it is neglected in the calculations.

To size the tank reactor correctly, the most important part is finding the maximum volume it needs to be able to contain. This can be done by looking at the thickest coating layer that is wanted, which is 500 *nm*. The compounds added before the coating phase is always the same and is equal to:

$$V_{Prep}(t) = V_{H2O} + V_{EtOH} + V_{CTACl} + V_{SiO2} = 0.32 + 0.25 + \frac{30}{968} + \frac{2}{815.3} = 0.603 \quad [m^3] \tag{B.7}$$

where V_{CTACl} and V_{SiO2} are calculated with the relation $V = \frac{m}{\rho}$.

With five coating layers, the total amount of added ammonia is 10 *kg*, and total amount of formaldehyde is 36 *kg*. The maximum resorcinol that can be added is based on both the flow rate corresponding to the maximum concentration, and the maximum time, as shown in Figure 3.4.2. Since the amount of resorcinol added to the tank is so small in comparison to the total volume in the reactor, the change due to higher concentration or longer coating time is neglected and 2 *kg* is used.

$$V(t) = V_{Prep}(t) + V_{Am} + V_{Form} + V_{Res} = 0.603 + \frac{10}{609} + \frac{32}{2650} + \left(\frac{2}{1280} + 0.02666\right) = 0.66 \quad [m^3] \tag{B.8}$$

To have some margin, the total volume capacity is decided to be 1 m^3 (1000 L), as this leaves room for adjustments of the volume of preparation phase compounds, in case experiments show that some of the compounds should be added in larger amounts. With the total volume capacity decided, the dimensions of the reactor can be calculated. The equation for the total volume is the volume of the cylinder.

$$V_{reac} = \pi \cdot r^2 \cdot h_{max,reac} \quad [m^3] \tag{B.9}$$

where r is the radius of the tank reactor, $h_{max,reac}$ is the height of the cylinder, which leads to:

$$r = \sqrt{\frac{1}{\pi \cdot h_{max,reac}}} \quad [m] \tag{B.10}$$

The height of the reactor can now be decided and the corresponding radius can then be easily calculated. The taller the reactor, the higher the flow rate it generates, thus it is decided that a fitting height is 2 *m*, which leads to the following radius:

$$r = \sqrt{\frac{1}{\pi \cdot 2}} = 0.399 \quad [m] \tag{B.11}$$

With the reactor's dimensions designed, it is possible to calculate the contact area between the reactor walls and content, which is the area of effective heat loss. This effect is dependent on the level in the reactor. Since there is a waste tank underneath the reactor and several external tanks above it, the heat loss in these directions are neglected, resulting in the following:

$$A_{reac}(t) = 2 \cdot \pi \cdot r \cdot h(t) \quad [m^2] \tag{B.12}$$

Due to the fact that it was not possible to gather data from this reactor to analyze the magnitude of heat loss, it was necessary to research a value for the convective heat loss. The value that was deemed most suitable was found in [10]. This stated that the heat loss through the walls of the reactor was equal to 21 $\frac{Btu}{ft^2 \cdot hr}$ at 160 $°F$, with two inches of insulation. Converted to metric units this is equal to 119.16 $\frac{W}{m^2 \cdot °C}$ at 71.11 $°C$. Since this process operates most of the time at 70 $°C$, it is assumed that the convective heat transfer is constant and is applicable during the entire process range. It is also wanted to have it defined by per minute and not per second. Therefore, it is multiplied by 60, which gives the following, $h_{reac} =$ 7149.6 $\frac{J}{m^2 \cdot °C \cdot min}$.

B.3 Heating Element

With the heating element dynamics being included, a specific heating element needed to be chosen in order to calculate the dynamics based on its physical properties. The choice landed on an immersion type heating element. This is an element placed inside the reactor, meaning that there is no heat loss in the convection from the element to the content in the reactor. If the agitator for stirring the solution does not allow a heating element inside the reactor, or if an immersion heating element is not ideal, a jacketed reactor/heating element is an alternative.

The model of heating element selected was the WATROD™ Screw Plug with Control Assembly BHNB24N3 -W2C11 [27]. The most important parameters for this element are: Power range of [0-100kW], length 919.2 mm, 24 heating rods, diameter 12.1 mm, heating element composite Alloy 800. It is now possible to calculate the values involved with the thermodynamics. A_h is the surface area of the heating element, this can be calculated as follows:

$$\begin{aligned} A_h &= (circumference\ of\ the\ heating\ rods) \cdot (2 \cdot length) \cdot (number\ of\ rods) \\ &= (0.0121 \cdot \pi) \cdot (0.9192 \cdot 2) \cdot (24) \\ &= 1.667 \quad [m^2] \end{aligned} \tag{B.13}$$

From the data sheet [27] it is know that the heat element has a Alloy 800 sheath. This information is used to calculate the mass of the heating element m_h, the heat transfer coefficient h_h and the specific heat capacity $c_{p,h}$. Based on the data sheet [22], the alloy sheath has a $c_{p,h} = 460$ $[\frac{J}{kg \cdot °C}]$, and it is known that it contains 39.5 % iron (minimum), [30-35] % nickel, [19-23] % chromium, and [0.3-1.2] % other compounds. The middle values of the ones with ranges are chosen and the other compounds are neglected and the amount of iron is increased to compensate. With these properties known, m_h can be calculated.

$$\begin{aligned} m_h &= (Area\ heating\ rods) \cdot (2 \cdot length) \cdot (number\ of\ rods) \cdot \big(0.325 \cdot \rho_{nickel} + 0.21 \cdot \rho_{chro} + 0.465 \cdot \rho_{iron}\big) \\ &= (\pi \cdot 0.006^2 \cdot 0.9192 \cdot 2 \cdot 24) \cdot (0.325 \cdot 8900 + 0.21 \cdot 7150 + 0.465 \cdot 7870) \\ &= 40.19 \quad [kg] \end{aligned} \tag{B.14}$$

The last parameter needed to be defined is the heat transfer coefficient h_h, which describes the convective heat transfer between the heating element and the contents of the reactor. Due to the circumstances of neither the real process, nor the heating element chosen being available, h_h was the only available data. To simplify the selection of the value, the contents of the reactor are assumed to have the same properties as water at free convection. The research led to a value of h_h being between $100 - 1200$ $\frac{W}{m^2 \cdot °C}$, as found in [5]. The variances is due to the fact that free convection is fairly complex and relies on a large number of factors. For this project, $h_h = 1200$ $\frac{W}{m^2 \cdot °C}$ was used as a basis, it was then converted to energy transfer per minute, by multiplying it with 60. Making it equal to, $h_h = 72000$ $\frac{J}{m^2 \cdot °C \cdot min}$.

Appendix C

State-Space Modeling

This chapter explains the calculations done to make a state-space model representing the system. In this project, the model is defined to be valid between the control steps $4\_Heating70$ and $8\_Settling$, as these are the only phases where control of the temperature and concentration is wanted. The steps $10\_Washing$ and 11_Drying will only use a set power on the heating element as the exact temperature value is not critical and the concentration of resorcinol is not present.

The state-space representation is comprised of five different vectors. The output vector $y(t)$, which includes the wanted outputs, the state vector $x(t)$, which includes the different state variables, the input vector $u(t)$, which includes the different input variables of the system, and the disturbance vector $v(t)$, which includes the system's disturbances. Firstly, $u(t)$ will be defined. The different input variables of this system are every variable that is controlled by the PLC, and they are the following: $u_1(t) = P(t),\ u_2(t) = x_{out}(t),\ u_3(t) = x_{res}(t),\ u_4(t) = x_{am}(t),\ and u_5(t) = x_{form}(t)$.

Secondly, the disturbances are defined. In the SIMIT model, the different external tank temperatures as well as the ambient air temperature were made to vary to resemble real variations, hence, the following are the disturbances on the system: $v_1(t) = T_{res,in}(t),\ v_2(t) = T_{am,in}(t),\ v_3(t) = T_{form,in}(t),\ and\ v_4(t) = T_{Ambient}(t)$. The states of the system are the time varying parameters whose values are needed to get the correct dynamics. In this system, these variables are the individual masses added to the reactor, the resorcinol concentration in the reactor, the reactor temperature, and the heating element temperature. They can be defined as: $x_1(t) = T_{reac}(t),\ x_2(t) = C_{res}(t),\ x_3(t) = T_h(t),\ x_4(t) = m_{res}(t),\ x_5(t) = m_{am}(t),\ x_6(t) = m_{form}(t)$, and $x_7(t) = m_{prep}(t)$. The general state-space representation then becomes the following:

$$\begin{bmatrix} \Delta\dot{x}_1(t) \\ \Delta\dot{x}_2(t) \\ \Delta\dot{x}_3(t) \\ \Delta\dot{x}_4(t) \\ \Delta\dot{x}_5(t) \\ \Delta\dot{x}_6(t) \\ \Delta\dot{x}_7(t) \end{bmatrix} = A \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \\ x_7(t) \end{bmatrix} + B \cdot \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \\ u_4(t) \\ u_5(t) \end{bmatrix} + B_d \cdot \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \end{bmatrix} \tag{C.1}$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = C \cdot \begin{bmatrix} \Delta x_1(t) \\ \Delta x_2(t) \\ \Delta x_3(t) \\ \Delta x_4(t) \\ \Delta x_5(t) \\ \Delta x_6(t) \\ \Delta x_7(t) \end{bmatrix} \tag{C.2}$$

The four different matrices A, B, B_d, and C describe how much the given variable affects the change of the state at a set WP. Matrix A is the state matrix, which describes how the values of the other states affects the given state, matrix B is the input matrix, which describes how the different input values affect the given state, matrix B_d is the disturbance matrix, which describes how the disturbances affect the given state, and matrix C is the output matrix, which describes which states that are selected as outputs.

The differential equations defining the states need to be linearized to get the equations used to find the matrices above. The differential equation for the states are the following:

$$\begin{aligned}\frac{dT_{reac}(t)}{dt} &= \frac{1}{m_{tot}(t)\cdot c_{p,tot}(t)}\cdot\Bigg(\sum_{i=1}^{3}\Big(c_{p,i}\cdot w_{i,in}(t)\cdot\Big(T_{i,in}(t)-T_{reac}(t)\Big)\Big)\\ &- h_{reac}\cdot A_{reac}(t)\Big(T_{reac}(t)-T_{Ambient}(t)\Big)+h_h\cdot A_h\Big(T_h(t)-T_{reac}(t)\Big)\Bigg)\quad\left[\frac{°C}{min}\right]\end{aligned}\tag{C.3}$$

$$\frac{dT_h(t)}{dt} = \frac{1}{m_h\cdot c_{p,h}}\cdot\Bigg(60\cdot P(t)-h_h\cdot A_h\big(T_h(t)-T_{reac}(t)\big)\Bigg)\quad\left[\frac{°C}{min}\right]\tag{C.4}$$

$$\frac{dC_{res}(t)}{dt} = 1000\cdot\Bigg(\frac{1}{V_{tot}(t)}\Big(C_{res,in}\cdot q_{res}(t)-C_{res}(t)\cdot\sum_{i=1}^{3}q_i(t)\Big)-A\cdot e^{\frac{-E_a}{R\cdot T_{reac}(t)}}\cdot C_{res}(t)\Bigg)\qquad\left[\frac{g}{min\cdot m^3}\right]\tag{C.5}$$

$$\frac{dm_{res}(t)}{dt} = \Big(w_{res}(t)-\frac{m_{res}(t)}{m_{tot}(t)}\cdot w_{out}(t)\Big)\quad\left[\frac{kg}{min}\right]\tag{C.6}$$

$$\frac{dm_{am}(t)}{dt} = \Big(w_{am}(t)-\frac{m_{am}(t)}{m_{tot}(t)}\cdot w_{out}(t)\Big)\quad\left[\frac{kg}{min}\right]\tag{C.7}$$

$$\frac{dm_{form}(t)}{dt} = \Big(w_{form}(t)-\frac{m_{form}(t)}{m_{tot}(t)}\cdot w_{out}(t)\Big)\quad\left[\frac{kg}{min}\right]\tag{C.8}$$

$$\frac{dm_{prep}(t)}{dt} = -\frac{m_{prep}(t)}{m_{tot}(t)}\cdot w_{out}(t)\quad\left[\frac{kg}{min}\right]\tag{C.9}$$

As seen, the equations contain several variables that are neither states, inputs, or disturbances. The variables in question are: $m_{tot}(t)$, $V_{tot}(t)$, $c_{p,tot}(t)$, $\rho_{tot}(t)$, $q_i(t)$, $w_i(t)$, and $A_{reac}(t)$. These variables can be replaced by a function of the different states or a static relation between inputs and can therefore be changed to the following:

$$m_{tot}(t) = \sum_{i=1}^{4} m_i(t)\tag{C.10}$$

$$V_{tot}(t) = \sum_{i=1}^{4}\frac{m_i(t)}{\rho_i}\tag{C.11}$$

$$c_{p,tot}(t) = \frac{\sum_{i=1}^{4}\left(c_{p,i}\cdot m_i(t)\right)}{\sum_{i=1}^{4}\left(m_i(t)\right)}\tag{C.12}$$

$$\rho_{tot}(t) = \frac{\sum_{i=1}^{4}\left(\rho_i\cdot m_i(t)\right)}{\sum_{i=1}^{4}\left(m_i(t)\right)}\tag{C.13}$$

$$A_{reac}(t) = \frac{2 \cdot m_{tot}(t)}{r \cdot \rho_{tot}(t)} = \frac{5 \cdot \left(\sum_{i=1}^{4} m_i(t)\right)^2}{\sum_{i=1}^{4} \left(\rho_i \cdot m_i(t)\right)} \tag{C.14}$$

$$q_i(t) = \frac{K_{v,i} \cdot x_i(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i(t)}{100000 \cdot SG_i}} \tag{C.15}$$

$$w_i(t) = q_i(t) \cdot \rho_i \tag{C.16}$$

The outflow from the reactor $q_{out}(t)$, also includes the level in the reactor $h(t)$, where as the other h_i values are constant, this variable can also be simplified to the following:

$$h(t) = \frac{m_{tot}(t)}{S \cdot \rho_{tot}(t)} = \frac{\left(\sum_{i=1}^{4} m_i(t)\right)^2}{\pi \cdot r^2 \cdot \sum_{i=1}^{4} \left(\rho_i \cdot m_i(t)\right)} \tag{C.17}$$

The final part is to take the partial derivative of every state's differential equation in regards to the variables included in the $x(t)$, $u(t)$, and $v(t)$ vectors apparent in the given differential equation. Thus yielding an equation that enables the opportunity to linearize the system in regards to a WP.

Reactor Temperature ($\mathbf{x}_1$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\begin{aligned}\frac{dT_{reac}(t)}{dt} = \frac{1}{\sum_{i=1}^{4} \left(c_{p,i} \cdot m_i(t)\right)} \cdot \Bigg(h_h \cdot A_h \Big(T_h(t) - T_{reac}(t)\Big) \\ + \sum_{i=1}^{3} \left(\frac{\rho_i \cdot c_{p,i} \cdot K_{v,i} \cdot x_i(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in}(t) - T_{reac}(t)\Big)\right) \\ - \frac{5 \cdot h_{reac} \cdot \left(\sum_{i=1}^{4} m_i(t)\right)^2 \cdot \Big(T_{reac}(t) - T_{Ambient}(t)\Big)}{\sum_{i=1}^{4} \left(\rho_i \cdot m_i(t)\right)} \Bigg)\end{aligned}$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\begin{aligned}\dot{T}_{reac}(t) = f_1\Big(T_{reac}(t),\ T_h(t),\ m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{res}(t), \\ x_{am}(t),\ x_{form}(t),\ T_{res,in}(t),\ T_{am,in}(t),\ T_{form,in}(t),\ T_{Ambient}(t)\Big)\end{aligned} \tag{C.18}$$

The function f_1 is partial derivated in regards to all variables mentioned in Equation C.18.

$$\frac{\partial f_1}{\partial T_{reac}} = -\frac{1}{\sum_{i=1}^{4} \left(c_{p,i} \cdot m_i\right)} \left(h_h \cdot A_h + \sum_{i=1}^{3} \left(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \right) + \frac{5 \cdot h_{reac} \cdot \left(\sum_{i=1}^{4} (m_i)\right)^2}{\sum_{i=1}^{4} \left(\rho_i \cdot m_i\right)} \right)$$

$$\frac{\partial f_1}{\partial T_h} = \frac{h_h \cdot A_h}{\sum_{i=1}^{4} \left(c_{p,i} \cdot m_i\right)}$$

$$\frac{\partial f_1}{\partial m_{res}} = \frac{1}{\Big(\sum_{i=1}^{4}\rho_i\cdot m_i\Big)^2\Big(\sum_{i=1}^{4}c_{p,i}\cdot m_i\Big)^2}\Bigg(m_{res}{}^2\cdot\Bigg(-c_{p,res}\cdot\rho_{res}{}^2\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)$$

$$-c_{p,res}\cdot\rho_{res}{}^2\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)$$

$$+5\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\Big(2\cdot c_{p,res}\cdot\rho_{res}\cdot\sum_{i=1}^{3}(m_i)-c_{p,res}\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)-\rho_{res}\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Big)\Bigg)$$

$$-m_{res}\cdot\Bigg(2\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)\cdot\Bigg(c_{p,res}\cdot\rho_{res}\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)$$

$$+c_{p,res}\cdot\rho_{res}\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)$$

$$+5\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Bigg)-\sum_{i=1}^{3}(\rho_i\cdot m_i)\cdot\Bigg(c_{p,res}\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)$$

$$+c_{p,res}\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)$$

$$+10\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\sum_{i=1}^{3}(m_i)\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Bigg)\Bigg)$$

$$\frac{\partial f_1}{\partial m_{am}} = \frac{1}{\Big(\sum_{i=1}^{4}\rho_i\cdot m_i\Big)^2\Big(\sum_{i=1}^{4}c_{p,i}\cdot m_i\Big)^2}\Bigg(m_{am}{}^2\cdot\Bigg(-c_{p,am}\cdot\rho_{am}{}^2\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)$$

$$-c_{p,am}\cdot\rho_{am}{}^2\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)$$

$$+5\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\Big(2\cdot c_{p,am}\cdot\rho_{am}\cdot\sum_{i=1}^{3}(m_i)-c_{p,am}\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)-\rho_{am}\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Big)\Bigg)$$

$$-m_{am}\cdot\Bigg(2\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)\cdot\Bigg(c_{p,am}\cdot\rho_{am}\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)$$

$$+c_{p,am}\cdot\rho_{am}\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)$$

$$+5\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Bigg)-\sum_{i=1}^{3}(\rho_i\cdot m_i)\cdot\Bigg(c_{p,am}\cdot h_h\cdot A_h\cdot\big(T_h-T_{reac}\big)\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)$$

$$+c_{p,am}\cdot\sum_{i=1}^{3}\Big(\frac{K_{v,i}\cdot\rho_i\cdot c_{p,i}\cdot x_i}{60}\cdot\sqrt{\frac{\rho_i\cdot g\cdot h_i}{100000\cdot SG_i}}\cdot\Big(T_{i,in}-T_{reac}\Big)\Big)\cdot\sum_{i=1}^{3}(\rho_i\cdot m_i)$$

$$+10\cdot h_{reac}\cdot\big(T_{reac}-T_{amb}\big)\cdot\sum_{i=1}^{3}(m_i)\cdot\sum_{i=1}^{3}(c_{p,i}\cdot m_i)\Bigg)\Bigg)$$

$$\frac{\partial f_1}{\partial m_{form}} = \frac{1}{\Big(\sum_{i=1}^{4} \rho_i \cdot m_i\Big)^2 \Big(\sum_{i=1}^{4} c_{p,i} \cdot m_i\Big)^2} \Bigg({m_{form}}^2 \cdot \Bigg(- c_{p,form} \cdot {\rho_{form}}^2 \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big)$$

$$- c_{p,form} \cdot {\rho_{form}}^2 \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big)$$

$$+5 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \Big(2 \cdot c_{p,form} \cdot \rho_{form} \cdot \sum_{i=1}^{3}(m_i) - c_{p,form} \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i) - \rho_{form} \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Big)\Bigg)$$

$$- m_{form} \cdot \Bigg(2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i) \cdot \Big(c_{p,form} \cdot \rho_{form} \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big)$$

$$+ c_{p,form} \cdot \rho_{form} \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big)$$

$$+5 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Big) - \sum_{i=1}^{3}(\rho_i \cdot m_i) \cdot \Bigg(c_{p,form} \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big) \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)$$

$$+ c_{p,form} \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big) \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)$$

$$+10 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \sum_{i=1}^{3}(m_i) \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Bigg)\Bigg)$$

$$\frac{\partial f_1}{\partial m_{prep}} = \frac{1}{\Big(\sum_{i=1}^{4} (\rho_i \cdot m_i)\Big)^2 \Big(\sum_{i=1}^{4} (c_{p,i} \cdot m_i)\Big)^2} \Bigg({m_{prep}}^2 \cdot \Bigg(- c_{p,prep} \cdot {\rho_{prep}}^2 \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big)$$

$$- c_{p,prep} \cdot {\rho_{prep}}^2 \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big)$$

$$+5 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \Big(2 \cdot c_{p,prep} \cdot \rho_{prep} \cdot \sum_{i=1}^{3}(m_i) - c_{p,prep} \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i) - \rho_{prep} \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Big)\Bigg)$$

$$- m_{prep} \cdot \Bigg(2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i) \cdot \Big(c_{p,prep} \cdot \rho_{prep} \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big)$$

$$+ c_{p,prep} \cdot \rho_{prep} \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big)$$

$$+5 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Big) - \sum_{i=1}^{3}(\rho_i \cdot m_i) \cdot \Bigg(c_{p,prep} \cdot h_h \cdot A_h \cdot \big(T_h - T_{reac}\big) \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)$$

$$+ c_{p,prep} \cdot \sum_{i=1}^{3} \Big(\frac{K_{v,i} \cdot \rho_i \cdot c_{p,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} \cdot \Big(T_{i,in} - T_{reac}\Big)\Big) \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)$$

$$+10 \cdot h_{reac} \cdot \big(T_{reac} - T_{amb}\big) \cdot \sum_{i=1}^{3}(m_i) \cdot \sum_{i=1}^{3}(c_{p,i} \cdot m_i)\Bigg)\Bigg)$$

$$\frac{\partial f_1}{\partial x_{res}} = \frac{\rho_{res} \cdot c_{p,res} \cdot K_{v,res} \cdot \Big(T_{res,in} - T_{reac}\Big)}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{res} \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}$$

$$\frac{\partial f_1}{\partial x_{am}} = \frac{\rho_{am} \cdot c_{p,am} \cdot K_{v,am} \cdot \Big(T_{am,in} - T_{reac}\Big)}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{am} \cdot g \cdot h_{am}}{100000 \cdot SG_{am}}}$$

$$\frac{\partial f_1}{\partial x_{form}} = \frac{\rho_{form} \cdot c_{p,form} \cdot K_{v,form} \cdot \Big(T_{form,in} - T_{reac}\Big)}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{form} \cdot g \cdot h_{form}}{100000 \cdot SG_{form}}}$$

$$\frac{\partial f_1}{\partial T_{res,in}} = \frac{c_{p,res} \cdot \rho_{res} \cdot K_{v,res} \cdot x_{res}}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{res} \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}$$

$$\frac{\partial f_1}{\partial T_{am,in}} = \frac{c_{p,am} \cdot \rho_{am} \cdot K_{v,am} \cdot x_{am}}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{am} \cdot g \cdot h_{am}}{100000 \cdot SG_{am}}}$$

$$\frac{\partial f_1}{\partial T_{form,in}} = \frac{c_{p,form} \cdot \rho_{form} \cdot K_{v,form} \cdot x_{form}}{60 \cdot \sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big)} \cdot \sqrt{\frac{\rho_{form} \cdot g \cdot h_{form}}{100000 \cdot SG_{form}}}$$

$$\frac{\partial f_1}{\partial T_{Ambient}} = \frac{5 \cdot h_{reac} \cdot \big(\sum_{i=1}^{4} m_i\big)^2}{\sum_{i=1}^{4} \big(c_{p,i} \cdot m_i\big) \cdot \sum_{i=1}^{4} \big(\rho_i \cdot m_i\big)}$$

The lineazried equation of C.18 is the following :

$$\Delta\dot{T}_{reac}(t) = \frac{\partial f_1}{\partial T_{reac}}\Delta T_{reac}(t) \ + \ \frac{\partial f_1}{\partial T_h}\Delta T_h(t) \ + \ \frac{\partial f_1}{\partial m_{res}}\Delta m_{res}(t) \ + \ \frac{\partial f_1}{\partial m_{am}}\Delta m_{am}(t) \ + \ \frac{\partial f_1}{\partial m_{form}}\Delta m_{form}(t)$$
$$+ \ \frac{\partial f_1}{\partial m_{prep}}\Delta m_{prep}(t) \ + \ \frac{\partial f_1}{\partial x_{res}}\Delta x_{res}(t) \ + \ \frac{\partial f_1}{\partial x_{am}}\Delta x_{am}(t) \ + \ \frac{\partial f_1}{\partial x_{form}}\Delta x_{form}(t) \ + \ \frac{\partial f_1}{\partial T_{res,in}}\Delta T_{res,in}(t)$$
$$+ \ \frac{\partial f_1}{\partial T_{am,in}}\Delta T_{am,in}(t) \ + \ \frac{\partial f_1}{\partial T_{form,in}}\Delta T_{form,in}(t) \ + \ \frac{\partial f_1}{\partial T_{Ambient}}\Delta T_{Ambient}(t)$$

Resorcinol Concentration ($\mathbf{x}_2$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\frac{dC_{res}(t)}{dt} = \left(\frac{1}{\sum_{i=1}^{4}\left(\frac{m_i(t)}{\rho_i}\right)} \cdot \left(\frac{C_{res,in} \cdot K_{v,res} \cdot x_{res}(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}\right.\right.$$
$$\left.\left. -C_{res}(t) \cdot \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i(t)}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}}\Big)\right) - A \cdot e^{\frac{-E_a}{R \cdot (T_{reac}(t)+273.15)}} \cdot C_{res}(t)\right)$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{C}_{res}(t) = f_2\Big(T_{reac}(t),\ C_{res}(t),\ m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{res}(t),\ x_{am}(t),\ x_{form}(t)\Big) \quad \text{(C.19)}$$

The function f_2 is partial derivated in regards to all variables mentioned in Equation C.19.

$$\frac{\partial f_2}{\partial T_{reac}} = -\frac{A \cdot E_a \cdot C_{res}}{R \cdot (T_{reac} + 273.15)^2} \cdot e^{\frac{-E_a}{R\cdot(T_{reac}+273.15)}}$$

$$\frac{\partial f_2}{\partial C_{res}} = \frac{1}{\sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)} \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}}\Big) - A \cdot e^{\frac{-E_a}{R\cdot(T_{reac}+273.15)}}$$

$$\frac{\partial f_2}{\partial m_{res}} = \frac{1}{\rho_{res} \cdot \Big(\sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)\Big)^2} \cdot \Bigg(C_{res} \cdot \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} - \frac{C_{res,in} \cdot K_{v,res} \cdot x_{res}}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}\Bigg)$$

$$\frac{\partial f_2}{\partial m_{am}} = \frac{1}{\rho_{am} \cdot \Big(\sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)\Big)^2} \cdot \Bigg(C_{res}(t) \cdot \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} - \frac{C_{res,in} \cdot K_{v,res} \cdot x_{res}}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}\Bigg)$$

$$\frac{\partial f_2}{\partial m_{form}} = \frac{1}{\rho_{form} \cdot \Big(\sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)\Big)^2} \cdot \Bigg(C_{res}(t) \cdot \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} - \frac{C_{res,in} \cdot K_{v,res} \cdot x_{res}}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}\Bigg)$$

$$\frac{\partial f_2}{\partial m_{prep}} = \frac{1}{\rho_{prep} \cdot \Big(\sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)\Big)^2} \cdot \Bigg(C_{res}(t) \cdot \sum_{i=1}^{3}\Big(\frac{K_{v,i} \cdot x_i}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_i}{100000 \cdot SG_i}} - \frac{C_{res,in} \cdot K_{v,res} \cdot x_{res}}{60} \cdot \sqrt{\frac{\rho_i \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}\Bigg)$$

$$\frac{\partial f_2}{\partial x_{res}} = \frac{K_{v,res} \cdot \big(C_{res,in} - C_{res}\big)}{60 \cdot \sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)} \cdot \sqrt{\frac{\rho_{res} \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}$$

$$\frac{\partial f_2}{\partial x_{am}} = -\frac{K_{v,am} \cdot C_{res}}{60 \cdot \sum_{i=1}^{4}\left(\frac{m_i}{\rho_i}\right)} \cdot \sqrt{\frac{\rho_{am} \cdot g \cdot h_{am}}{100000 \cdot SG_{am}}}$$

$$\frac{\partial f_2}{\partial x_{form}} = -\frac{K_{v,form} \cdot C_{res}}{60 \cdot \sum_{i=1}^{4} \left(\frac{m_i}{\rho_i}\right)} \cdot \sqrt{\frac{\rho_{form} \cdot g \cdot h_{form}}{100000 \cdot SG_{form}}}$$

The linearized equation of Equation C.19 is the following:

$$\begin{aligned}\Delta \dot{C}_{res}(t) = {} & \frac{\partial f_2}{\partial T_{reac}} \Delta T_{reac}(t) \; + \; \frac{\partial f_2}{\partial C_{res}} \Delta C_{res}(t) \; + \; \frac{\partial f_2}{\partial m_{res}} \Delta m_{res}(t) \; + \; \frac{\partial f_2}{\partial m_{am}} \Delta m_{am}(t) \\ & + \frac{\partial f_2}{\partial m_{form}} \Delta m_{form}(t) \; + \; \frac{\partial f_2}{\partial m_{prep}} \Delta m_{prep}(t) \; + \; \frac{\partial f_2}{\partial x_{res}} \Delta x_{res}(t) \\ & + \frac{\partial f_2}{\partial x_{am}} \Delta x_{am}(t) \; + \; \frac{\partial f_2}{\partial x_{form}} \Delta x_{form}(t)\end{aligned}$$

Heating Element ($\mathbf{x}_3$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\frac{dT_h(t)}{dt} = \frac{1}{m_h \cdot c_{p,h}} \cdot \left(60 \cdot P(t) - h_h \cdot A_h \big(T_h(t) - T_{reac}(t)\big)\right)$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{T}_h(t) = f_3\Big(T_{reac}(t),\ T_h(t),\ P(t)\Big) \tag{C.20}$$

The function f_3 is partial derivated in regards to all variables mentioned in Equation C.20.

$$\frac{\partial f_3}{\partial T_{reac}} = \frac{h_h \cdot A_h}{m_h \cdot c_{p,h}}$$

$$\frac{\partial f_3}{\partial T_h} = -\frac{h_h \cdot A_h}{m_h \cdot c_{p,h}}$$

$$\frac{\partial f_3}{\partial P} = \frac{60}{m_h \cdot c_{p,h}}$$

The linearized equation of Equation C.20 is the following:

$$\Delta \dot{T}_h(t) = \frac{h_h \cdot A_h}{m_h \cdot c_{p,h}} \Delta T_{reac}(t) - \frac{h_h \cdot A_h}{m_h \cdot c_{p,h}} \Delta T_h(t) + \frac{1}{m_h \cdot c_{p,h}} \Delta P(t) \tag{C.21}$$

Mass of Resorcinol ($\mathbf{x}_4$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\frac{dm_{res}(t)}{dt} = \frac{\rho_{res} \cdot K_{v,res} \cdot x_{res}(t)}{60} \cdot \sqrt{\frac{\rho_{res} \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}}$$
$$- \frac{m_{res}(t) \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t)) \cdot K_{v,out} \cdot x_{out}(t)}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i(t))} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}}$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{m}_{res}(t) = f_4\Big(m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{out}(t),\ x_{res}(t)\Big) \tag{C.22}$$

The function f_4 is partial derivated in regards to all variables mentioned in Equation C.22.

$$\frac{\partial f_4}{\partial m_{res}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot \Big(\rho_{res} \cdot {m_{res}}^2 + 3 \cdot \rho_{res} \cdot m_{res} \cdot \sum_{i=1}^{3}(m_i) + 2 \cdot \big(\sum_{i=1}^{3} m_i\big) \cdot \big(\sum_{i=1}^{3} \rho_i \cdot m_i\big)\Big)}{120 \cdot r \cdot 100000 \cdot \pi\big(\sum_{i=1}^{4}(m_i)\big)^2 \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}$$
$$\cdot\sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_4}{\partial m_{am}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{res} \cdot \Big(\rho_{am} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{am}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000 \cdot \pi\big(\sum_{i=1}^{4}(m_i)\big)^2 \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}$$
$$\cdot\sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_4}{\partial m_{form}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{res} \cdot \Big(\rho_{form} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{form}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000 \cdot \pi\big(\sum_{i=1}^{4}(m_i)\big)^2 \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}$$
$$\cdot\sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_4}{\partial m_{prep}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{res} \cdot \Big(\rho_{prep} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{prep}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}$$
$$\cdot\sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_4}{\partial x_{out}} = -\frac{m_{res} \cdot K_{v,out} \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i)} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}} \tag{C.23}$$

$$\frac{\partial f_4}{\partial x_{res}} = \frac{\rho_{res} \cdot K_{v,res}}{60} \cdot \sqrt{\frac{\rho_{res} \cdot g \cdot h_{res}}{100000 \cdot SG_{res}}} \tag{C.24}$$

The linearized equation of Equation C.22 is the following:

$$\begin{aligned}\Delta \dot{m}_{res}(t) \;=\; & \frac{\partial f_4}{\partial m_{res}}\Delta m_{res}(t) \;+\; \frac{\partial f_4}{\partial m_{am}}\Delta m_{am}(t) \;+\; \frac{\partial f_4}{\partial m_{form}}\Delta m_{form}(t) \\ & +\; \frac{\partial f_4}{\partial m_{prep}}\Delta m_{prep}(t) \;+\; \frac{\partial f_4}{\partial x_{out}}\Delta x_{out}(t) \;+\; \frac{\partial f_4}{\partial x_{res}}\Delta x_{res}(t)\end{aligned}$$

Mass of Ammonia ($\mathbf{x}_5$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\begin{aligned}\frac{dm_{am}(t)}{dt} = & \frac{\rho_{am} \cdot K_{v,am} \cdot x_{am}(t)}{60} \cdot \sqrt{\frac{\rho_{am} \cdot g \cdot h_{am}}{100000 \cdot SG_{am}}} \\ -& \frac{m_{am}(t) \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t)) \cdot K_{v,out} \cdot x_{out}(t)}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i(t))} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}}\end{aligned}$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{m}_{am}(t) = f_5\Big(m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{out}(t),\ x_{am}(t)\Big) \tag{C.25}$$

The function f_5 is partial derivated in regards to all variables mentioned in Equation C.25.

$$\begin{aligned}\frac{\partial f_5}{\partial m_{res}} = -& \frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{am} \cdot \Big(\rho_{res} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{res}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \\ & \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}\end{aligned}$$

$$\begin{aligned}\frac{\partial f_5}{\partial m_{am}} = -& \frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot \Big(\rho_{am} \cdot {m_{am}}^2 + 2 \cdot \big(\sum_{i=1}^{3} m_i\big) \cdot \big(\sum_{i=1}^{3} \rho_i \cdot m_i\big) + 3 \cdot \rho_{am} \cdot m_{am} \cdot \sum_{i=1}^{3} m_i\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \\ & \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}\end{aligned}$$

$$\begin{aligned}\frac{\partial f_5}{\partial m_{form}} = -& \frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{am} \cdot \Big(\rho_{form} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{form}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \\ & \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}\end{aligned}$$

$$\frac{\partial f_5}{\partial m_{prep}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{am} \cdot \Big(\rho_{prep} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{prep}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_5}{\partial x_{out}} = -\frac{m_{am} \cdot K_{v,out} \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i)} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}} \tag{C.26}$$

$$\frac{\partial f_5}{\partial x_{am}} = \frac{\rho_{am} \cdot K_{v,am}}{60} \cdot \sqrt{\frac{\rho_{am} \cdot g \cdot h_{am}}{100000 \cdot SG_{am}}} \tag{C.27}$$

The linearized equation of Equation C.25 is the following:

$$\Delta\dot{m}_{am}(t) = \frac{\partial f_5}{\partial m_{res}}\Delta m_{res}(t) + \frac{\partial f_5}{\partial m_{am}}\Delta m_{am}(t) + \frac{\partial f_5}{\partial m_{form}}\Delta m_{form}(t) + \frac{\partial f_5}{\partial m_{prep}}\Delta m_{prep}(t) + \frac{\partial f_5}{\partial x_{out}}\Delta x_{out}(t) + \frac{\partial f_5}{\partial x_{am}}\Delta x_{am}(t)$$

Mass of Formaldehyde ($\mathbf{x}_6$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\frac{dm_{form}(t)}{dt} = \frac{\rho_{form} \cdot K_{v,form} \cdot x_{am}(t)}{60} \cdot \sqrt{\frac{\rho_{form} \cdot g \cdot h_{form}}{100000 \cdot SG_{form}}} - \frac{m_{form}(t) \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t)) \cdot K_{v,out} \cdot x_{out}(t)}{60 \cdot r \cdot \sum_{i=1}^{4} m_i(t)} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}}$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{m}_{form}(t) = f_6\Big(m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{out}(t),\ x_{form}(t)\Big) \tag{C.28}$$

The function f_6 is partial derivated in regards to all variables mentioned in Equation C.28.

$$\frac{\partial f_6}{\partial m_{res}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{form} \cdot \Big(\rho_{res} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{res}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_6}{\partial m_{am}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{form} \cdot \Big(\rho_{am} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{am}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_6}{\partial m_{form}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot \Big(\rho_{form} \cdot {m_{form}}^2 + 2 \cdot \big(\sum_{i=1}^{3} m_i\big) \cdot \big(\sum_{i=1}^{3} \rho_i \cdot m_i\big) + 3 \cdot \rho_{form} \cdot m_{form} \cdot \sum_{i=1}^{3} m_i\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_6}{\partial m_{prep}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{form} \cdot \Big(\rho_{prep} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{prep}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big)} \cdot \sqrt{\frac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}$$

$$\frac{\partial f_6}{\partial x_{out}} = -\frac{m_{form} \cdot \sum_{i=1}^{5}(\rho_i \cdot m_i) \cdot K_{v,out}}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i)} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}} \tag{C.29}$$

$$\frac{\partial f_6}{\partial x_{form}} = \frac{\rho_{form} \cdot K_{v,form}}{60} \cdot \sqrt{\frac{\rho_{form} \cdot g \cdot h_{form}}{100000 \cdot SG_{form}}} \tag{C.30}$$

The linearized equation of Equation C.25 is the following:

$$\begin{aligned}\Delta \dot{m}_{form}(t) \;=\; & \frac{\partial f_6}{\partial m_{res}}\Delta m_{res}(t) \;+\; \frac{\partial f_6}{\partial m_{am}}\Delta m_{am}(t) \;+\; \frac{\partial f_6}{\partial m_{form}}\Delta m_{form}(t) \;+\; \frac{\partial f_6}{\partial m_{prep}}\Delta m_{prep}(t) \\ & + \frac{\partial f_6}{\partial x_{out}}\Delta x_{out}(t) \;+\; \frac{\partial f_6}{\partial x_{form}}\Delta x_{form}(t)\end{aligned}$$

Mass of Preparation ($\mathbf{x_7}$):
By replacing the mentioned variables in the Equations C.10 - C.16, with their state or input defined counterpart, the equation that represents the change of reactor temperature depending on the states, inputs and disturbances defined becomes the following:

$$\frac{dm_{prep}(t)}{dt} = -\frac{m_{prep}(t) \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t)) \cdot K_{v,out} \cdot x_{out}(t)}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i(t))} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i(t))}}$$

A simplification of the equation above can be made, to make the different variables apparent in this state more clear.

$$\dot{m}_{prep}(t) = f_7\Big(m_{res}(t),\ m_{am}(t),\ m_{form}(t),\ m_{prep}(t),\ x_{out}(t)\Big) \tag{C.31}$$

The function f_7 is partial derivated in regards to all variables mentioned in Equation C.31.

$$\frac{\partial f_7}{\partial m_{res}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{prep} \cdot \Big(\rho_{res} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{res}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big) \cdot \sqrt{\dfrac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}}$$

$$\frac{\partial f_7}{\partial m_{am}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{prep} \cdot \Big(\rho_{am} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{am}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big) \cdot \sqrt{\dfrac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}}$$

$$\frac{\partial f_7}{\partial m_{form}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot m_{prep} \cdot \Big(\rho_{form} \cdot \big(\sum_{i=1}^{3}(m_i) - m_{form}\big) - 2 \cdot \sum_{i=1}^{3}(\rho_i \cdot m_i)\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big) \cdot \sqrt{\dfrac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}}$$

$$\frac{\partial f_7}{\partial m_{prep}} = -\frac{K_{v,out} \cdot x_{out} \cdot g \cdot \rho_{H2O} \cdot \Big(\rho_{prep} \cdot {m_{prep}}^2 + 2 \cdot \big(\sum_{i=1}^{3} m_i\big) \cdot \big(\sum_{i=1}^{3} \rho_i \cdot m_i\big) + 3 \cdot \rho_{prep} \cdot m_{prep} \cdot \sum_{i=1}^{3} m_i\Big)}{120 \cdot r \cdot 100000\pi\big(\sum_{i=1}^{4} m_i\big)^2 \cdot \big(\sum_{i=1}^{4} \rho_i \cdot m_i\big) \cdot \sqrt{\dfrac{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}{g \cdot \rho_{H2O}}}}$$

$$\frac{\partial f_7}{\partial x_{out}} = -\frac{m_{prep} \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i) \cdot K_{v,out}}{60 \cdot r \cdot \sum_{i=1}^{4}(m_i)} \cdot \sqrt{\frac{g \cdot \rho_{H2O}}{100000 \cdot \pi \cdot \sum_{i=1}^{4}(\rho_i \cdot m_i)}} \tag{C.32}$$

$$\begin{aligned}\Delta\dot{m}_{prep}(t) \ = \ & \frac{\partial f_7}{\partial m_{res}}\Delta m_{res}(t) \ + \ \frac{\partial f_7}{\partial m_{am}}\Delta m_{am}(t) \ + \ \frac{\partial f_7}{\partial m_{form}}\Delta m_{form}(t) \\ & + \ \frac{\partial f_7}{\partial m_{prep}}\Delta m_{prep}(t) \ + \ \frac{\partial f_7}{\partial x_{out}}\Delta x_{out}(t)\end{aligned}$$

State-space representation

The general state-space representation of the system then becomes the following:

$$\begin{bmatrix} \Delta\dot{T}_{reac}(t) \\ \Delta\dot{C}_{res}(t) \\ \Delta\dot{T}_h(t) \\ \Delta\dot{m}_{res}(t) \\ \Delta\dot{m}_{am}(t) \\ \Delta\dot{m}_{form}(t) \\ \Delta\dot{m}_{prep}(t) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial T_{reac}} & 0 & \frac{\partial f_1}{\partial T_h} & \frac{\partial f_1}{\partial m_{res}} & \frac{\partial f_1}{\partial m_{am}} & \frac{\partial f_1}{\partial m_{form}} & \frac{\partial f_1}{\partial m_{prep}} \\ \frac{\partial f_2}{\partial T_{reac}} & \frac{\partial f_2}{\partial C_{res}} & 0 & \frac{\partial f_2}{\partial m_{res}} & \frac{\partial f_2}{\partial m_{am}} & \frac{\partial f_2}{\partial m_{form}} & \frac{\partial f_2}{\partial m_{prep}} \\ \frac{\partial f_3}{\partial T_{reac}} & 0 & \frac{\partial f_3}{\partial T_h} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial f_4}{\partial m_{res}} & \frac{\partial f_4}{\partial m_{am}} & \frac{\partial f_4}{\partial m_{form}} & \frac{\partial f_4}{\partial m_{prep}} \\ 0 & 0 & 0 & \frac{\partial f_5}{\partial m_{res}} & \frac{\partial f_5}{\partial m_{am}} & \frac{\partial f_5}{\partial m_{form}} & \frac{\partial f_5}{\partial m_{prep}} \\ 0 & 0 & 0 & \frac{\partial f_6}{\partial m_{res}} & \frac{\partial f_6}{\partial m_{am}} & \frac{\partial f_6}{\partial m_{form}} & \frac{\partial f_6}{\partial m_{prep}} \\ 0 & 0 & 0 & \frac{\partial f_7}{\partial m_{res}} & \frac{\partial f_7}{\partial m_{am}} & \frac{\partial f_7}{\partial m_{form}} & \frac{\partial f_7}{\partial m_{prep}} \end{bmatrix} \cdot \begin{bmatrix} \Delta T_{reac}(t) \\ \Delta C_{res}(t) \\ \Delta T_h(t) \\ \Delta m_{res}(t) \\ \Delta m_{am}(t) \\ \Delta m_{form}(t) \\ \Delta m_{prep}(t) \end{bmatrix}$$
$$+ \begin{bmatrix} 0 & 0 & \frac{\partial f_1}{\partial x_{res}} & \frac{\partial f_1}{\partial x_{am}} & \frac{\partial f_1}{\partial x_{form}} \\ 0 & 0 & \frac{\partial f_2}{\partial x_{res}} & \frac{\partial f_2}{\partial x_{am}} & \frac{\partial f_2}{\partial x_{form}} \\ \frac{\partial f_3}{\partial P} & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial f_4}{\partial x_{out}} & \frac{\partial f_4}{\partial x_{res}} & 0 & 0 \\ 0 & \frac{\partial f_5}{\partial x_{out}} & 0 & \frac{\partial f_5}{\partial x_{am}} & 0 \\ 0 & \frac{\partial f_6}{\partial x_{out}} & 0 & 0 & \frac{\partial f_6}{\partial x_{form}} \\ 0 & \frac{\partial f_7}{\partial x_{out}} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta P(t) \\ \Delta x_{out}(t) \\ \Delta x_{res}(t) \\ \Delta x_{am}(t) \\ \Delta x_{form}(t) \end{bmatrix} \tag{C.33}$$
$$+ \begin{bmatrix} \frac{\partial f_1}{\partial T_{res,in}} & \frac{\partial f_1}{\partial T_{am,in}} & \frac{\partial f_1}{\partial T_{form,in}} & \frac{\partial f_1}{\partial T_{Ambient}} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta T_{res,in}(t) \\ \Delta T_{am,in}(t) \\ \Delta T_{form,in}(t) \\ \Delta T_{Ambient}(t) \end{bmatrix}$$

$$\begin{bmatrix} \Delta T_{reac}(t) \\ \Delta C_{res}(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta T_{reac}(t) \\ \Delta C_{res}(t) \\ \Delta T_h(t) \\ \Delta m_{res}(t) \\ \Delta m_{am}(t) \\ \Delta m_{form}(t) \\ \Delta m_{prep}(t) \end{bmatrix} \tag{C.34}$$

Appendix D

TIA Portal Programming

This appendix shows the program code from the TIA portal for the PLC control.

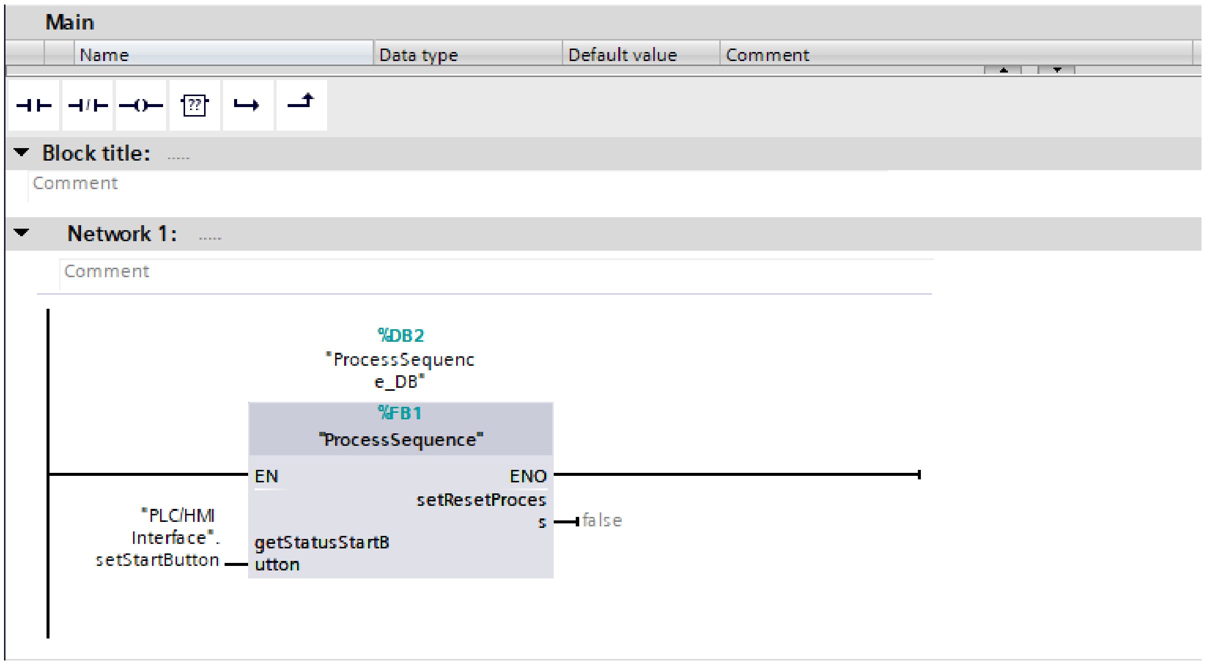
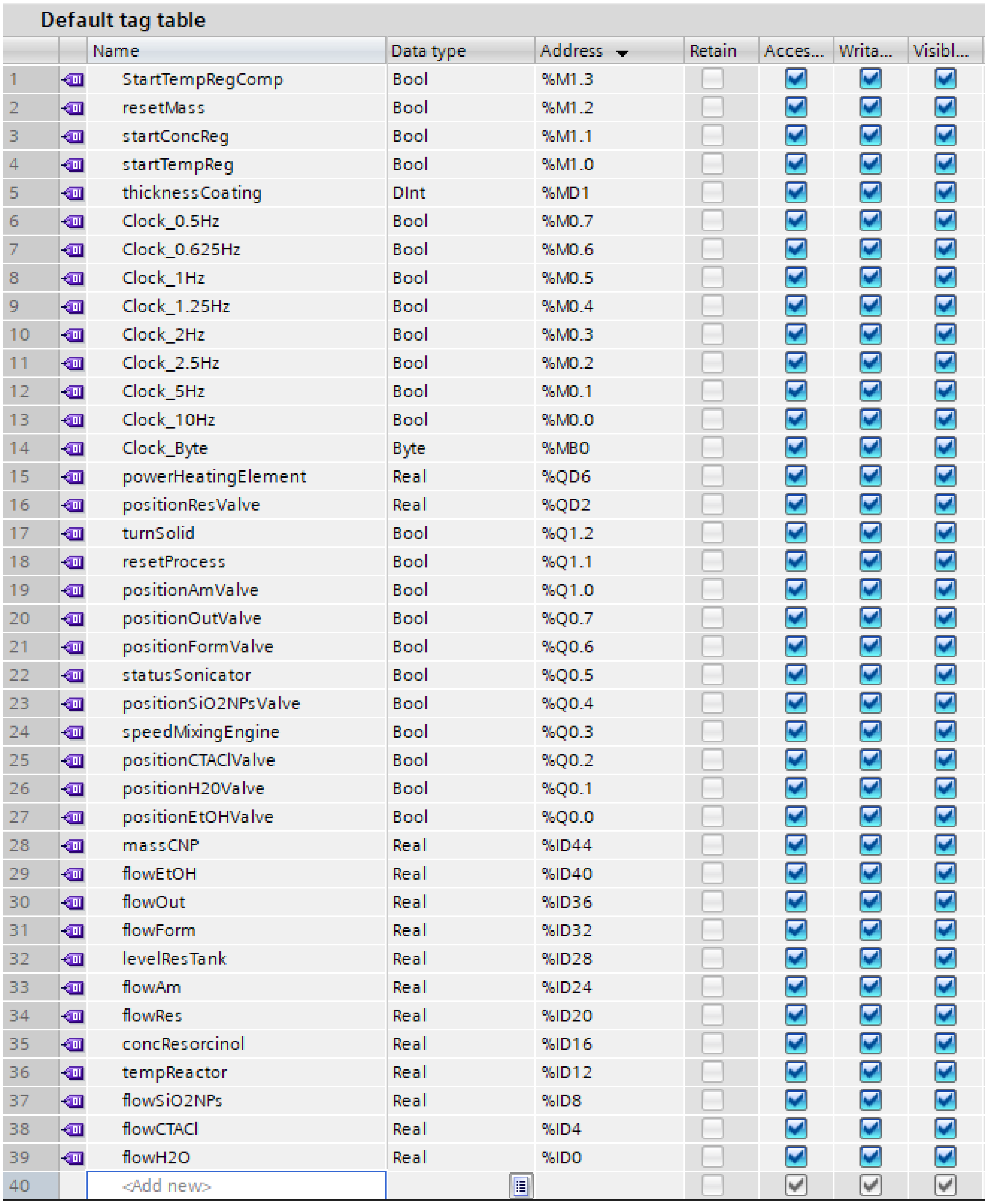


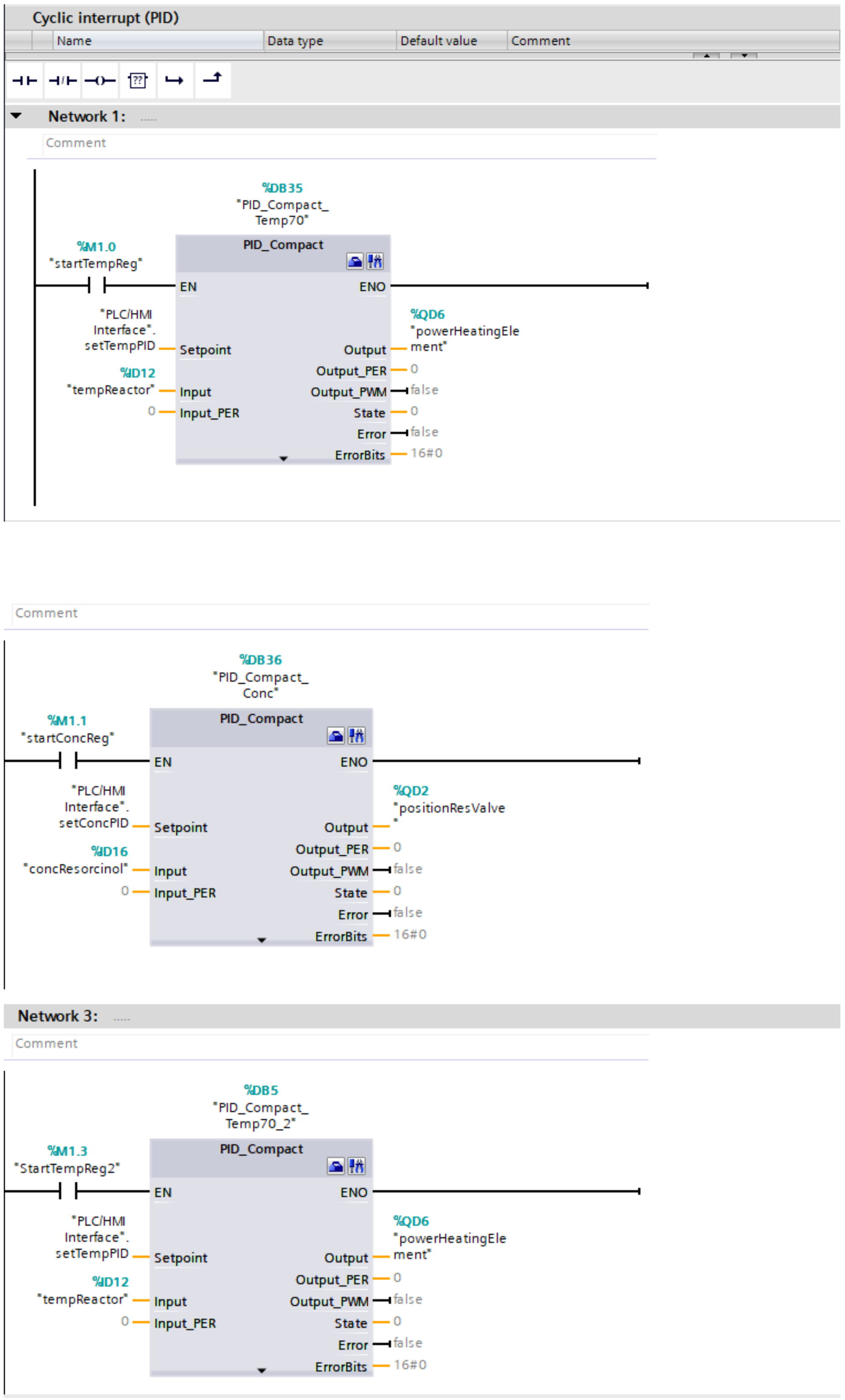
Figure D.0.1: Main



Default tag table

		Name	Data type	Address	Retain	Acces...	Writa...	Visibl...
1		StartTempRegComp	Bool	%M1.3	☐	✓	✓	✓
2		resetMass	Bool	%M1.2	☐	✓	✓	✓
3		startConcReg	Bool	%M1.1	☐	✓	✓	✓
4		startTempReg	Bool	%M1.0	☐	✓	✓	✓
5		thicknessCoating	DInt	%MD1	☐	✓	✓	✓
6		Clock_0.5Hz	Bool	%M0.7	☐	✓	✓	✓
7		Clock_0.625Hz	Bool	%M0.6	☐	✓	✓	✓
8		Clock_1Hz	Bool	%M0.5	☐	✓	✓	✓
9		Clock_1.25Hz	Bool	%M0.4	☐	✓	✓	✓
10		Clock_2Hz	Bool	%M0.3	☐	✓	✓	✓
11		Clock_2.5Hz	Bool	%M0.2	☐	✓	✓	✓
12		Clock_5Hz	Bool	%M0.1	☐	✓	✓	✓
13		Clock_10Hz	Bool	%M0.0	☐	✓	✓	✓
14		Clock_Byte	Byte	%MB0	☐	✓	✓	✓
15		powerHeatingElement	Real	%QD6	☐	✓	✓	✓
16		positionResValve	Real	%QD2	☐	✓	✓	✓
17		turnSolid	Bool	%Q1.2	☐	✓	✓	✓
18		resetProcess	Bool	%Q1.1	☐	✓	✓	✓
19		positionAmValve	Bool	%Q1.0	☐	✓	✓	✓
20		positionOutValve	Bool	%Q0.7	☐	✓	✓	✓
21		positionFormValve	Bool	%Q0.6	☐	✓	✓	✓
22		statusSonicator	Bool	%Q0.5	☐	✓	✓	✓
23		positionSiO2NPsValve	Bool	%Q0.4	☐	✓	✓	✓
24		speedMixingEngine	Bool	%Q0.3	☐	✓	✓	✓
25		positionCTAClValve	Bool	%Q0.2	☐	✓	✓	✓
26		positionH2OValve	Bool	%Q0.1	☐	✓	✓	✓
27		positionEtOHValve	Bool	%Q0.0	☐	✓	✓	✓
28		massCNP	Real	%ID44	☐	✓	✓	✓
29		flowEtOH	Real	%ID40	☐	✓	✓	✓
30		flowOut	Real	%ID36	☐	✓	✓	✓
31		flowForm	Real	%ID32	☐	✓	✓	✓
32		levelResTank	Real	%ID28	☐	✓	✓	✓
33		flowAm	Real	%ID24	☐	✓	✓	✓
34		flowRes	Real	%ID20	☐	✓	✓	✓
35		concResorcinol	Real	%ID16	☐	✓	✓	✓
36		tempReactor	Real	%ID12	☐	✓	✓	✓
37		flowSiO2NPs	Real	%ID8	☐	✓	✓	✓
38		flowCTACl	Real	%ID4	☐	✓	✓	✓
39		flowH2O	Real	%ID0	☐	✓	✓	✓
40		<Add new>			☐	✓	✓	✓

Figure D.0.2: Tag Table


Cyclic interrupt (PID)

Name	Data type	Default value	Comment

Network 1:

Comment

%DB35 "PID_Compact_Temp70" — PID_Compact

Input	Operand	Output	Operand
EN	%M1.0 "startTempReg"	ENO	
Setpoint	"PLC/HMI Interface".setTempPID	Output	%QD6 "powerHeatingElement"
Input	%ID12 "tempReactor"	Output_PER	0
Input_PER	0	Output_PWM	false
		State	0
		Error	false
		ErrorBits	16#0

Comment

%DB36 "PID_Compact_Conc" — PID_Compact

Input	Operand	Output	Operand
EN	%M1.1 "startConcReg"	ENO	
Setpoint	"PLC/HMI Interface".setConcPID	Output	%QD2 "positionResValve"
Input	%ID16 "concResorcinol"	Output_PER	0
Input_PER	0	Output_PWM	false
		State	0
		Error	false
		ErrorBits	16#0

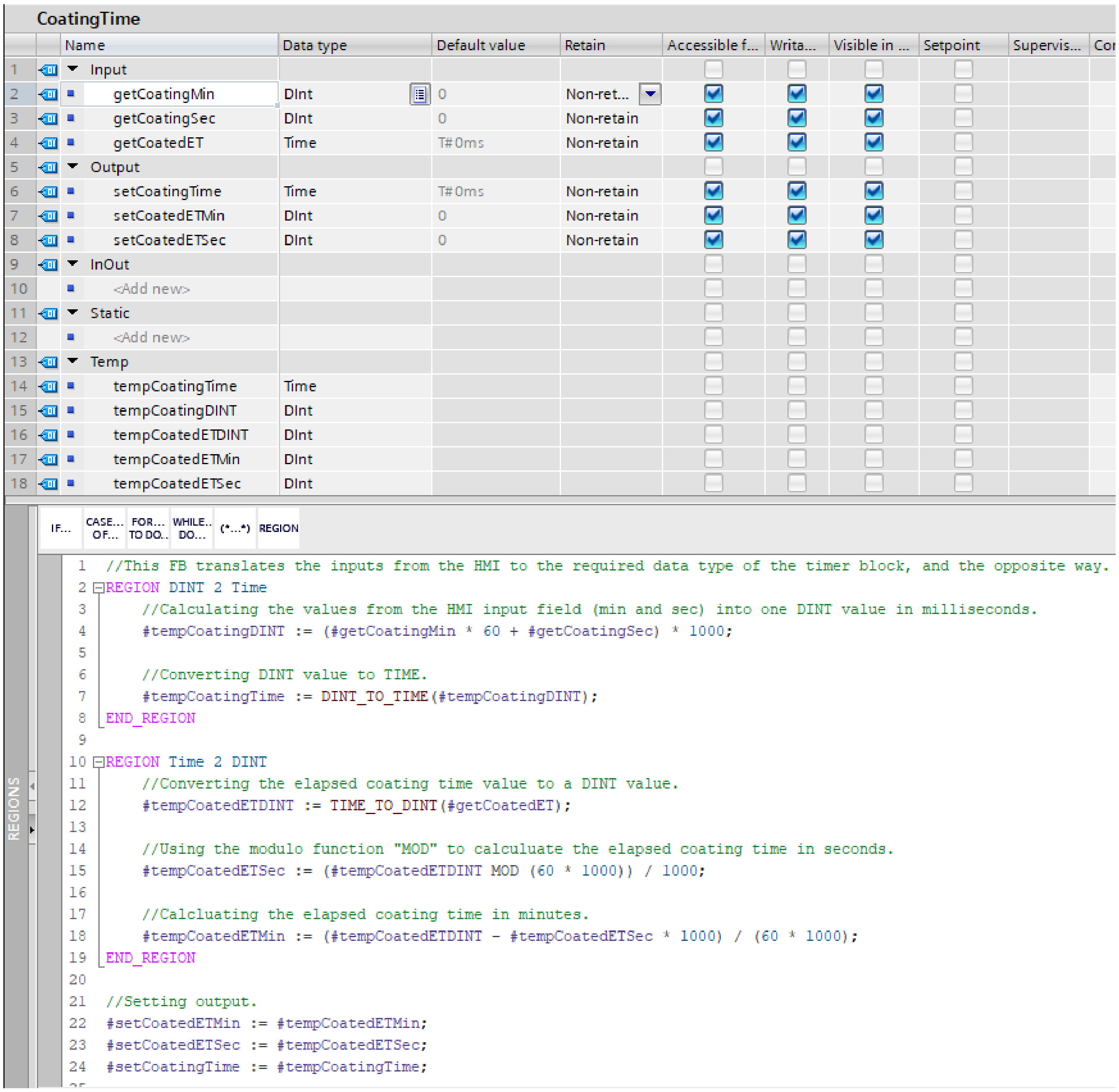
Network 3:

Comment

%DB5 "PID_Compact_Temp70_2" — PID_Compact

Input	Operand	Output	Operand
EN	%M1.3 "StartTempReg2"	ENO	
Setpoint	"PLC/HMI Interface".setTempPID	Output	%QD6 "powerHeatingElement"
Input	%ID12 "tempReactor"	Output_PER	0
Input_PER	0	Output_PWM	false
		State	0
		Error	false
		ErrorBits	16#0

Figure D.0.3: Cyclic interrupt (PID)

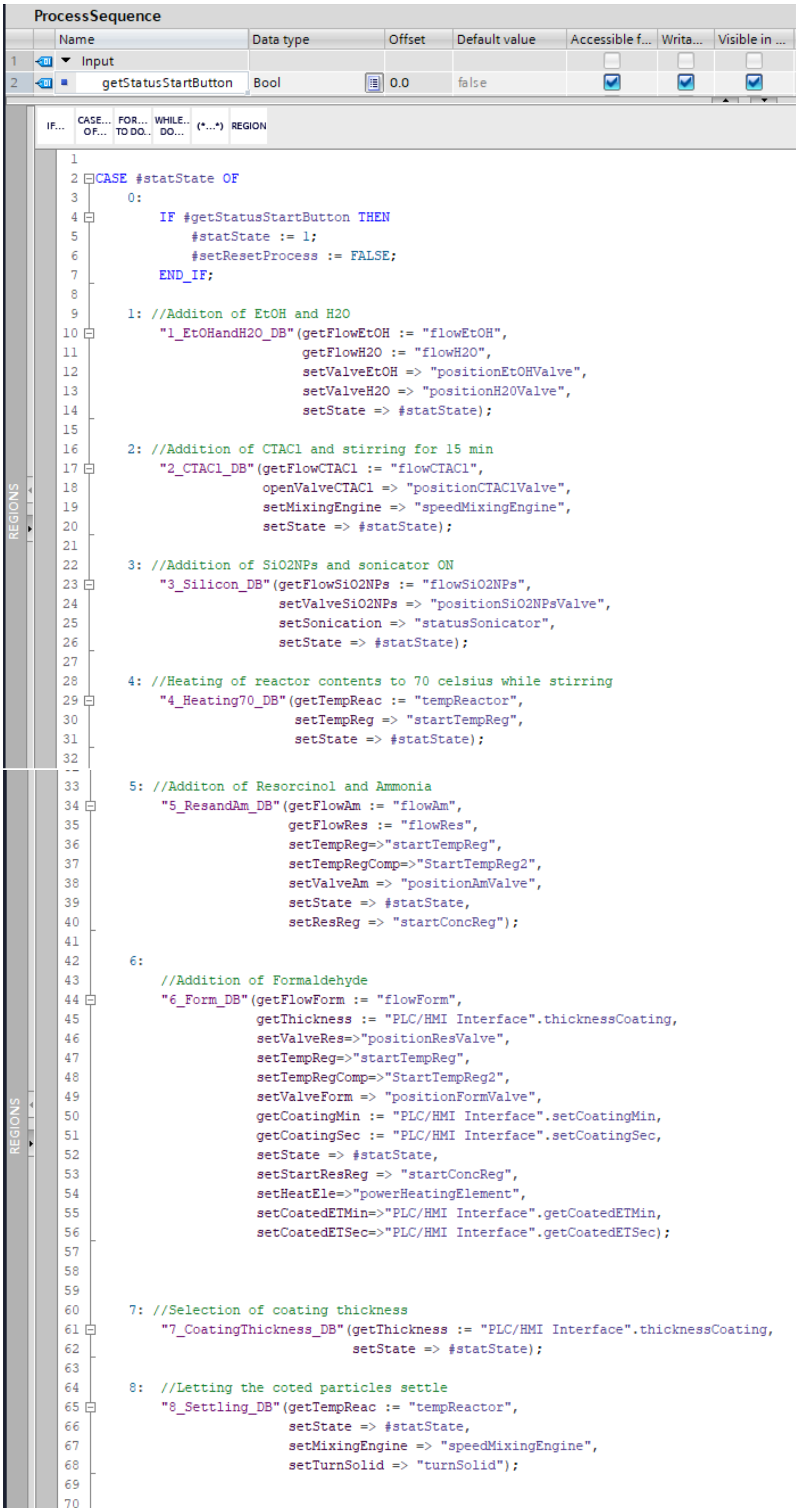


CoatingTime

	Name	Data type	Default value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Cor
1	▼ Input				☐	☐	☐	☐		
2	getCoatingMin	DInt	0	Non-ret...	✓	✓	✓	☐		
3	getCoatingSec	DInt	0	Non-retain	✓	✓	✓	☐		
4	getCoatedET	Time	T#0ms	Non-retain	✓	✓	✓	☐		
5	▼ Output				☐	☐	☐	☐		
6	setCoatingTime	Time	T#0ms	Non-retain	✓	✓	✓	☐		
7	setCoatedETMin	DInt	0	Non-retain	✓	✓	✓	☐		
8	setCoatedETSec	DInt	0	Non-retain	✓	✓	✓	☐		
9	▼ InOut				☐	☐	☐	☐		
10	<Add new>				☐	☐	☐	☐		
11	▼ Static				☐	☐	☐	☐		
12	<Add new>				☐	☐	☐	☐		
13	▼ Temp				☐	☐	☐	☐		
14	tempCoatingTime	Time			☐	☐	☐	☐		
15	tempCoatingDINT	DInt			☐	☐	☐	☐		
16	tempCoatedETDINT	DInt			☐	☐	☐	☐		
17	tempCoatedETMin	DInt			☐	☐	☐	☐		
18	tempCoatedETSec	DInt			☐	☐	☐	☐		

```
//This FB translates the inputs from the HMI to the required data type of the timer block, and the opposite way.
REGION DINT 2 Time
    //Calculating the values from the HMI input field (min and sec) into one DINT value in milliseconds.
    #tempCoatingDINT := (#getCoatingMin * 60 + #getCoatingSec) * 1000;

    //Converting DINT value to TIME.
    #tempCoatingTime := DINT_TO_TIME(#tempCoatingDINT);
END_REGION

REGION Time 2 DINT
    //Converting the elapsed coating time value to a DINT value.
    #tempCoatedETDINT := TIME_TO_DINT(#getCoatedET);

    //Using the modulo function "MOD" to calculuate the elapsed coating time in seconds.
    #tempCoatedETSec := (#tempCoatedETDINT MOD (60 * 1000)) / 1000;

    //Calcluating the elapsed coating time in minutes.
    #tempCoatedETMin := (#tempCoatedETDINT - #tempCoatedETSec * 1000) / (60 * 1000);
END_REGION

//Setting output.
#setCoatedETMin := #tempCoatedETMin;
#setCoatedETSec := #tempCoatedETSec;
#setCoatingTime := #tempCoatingTime;
```

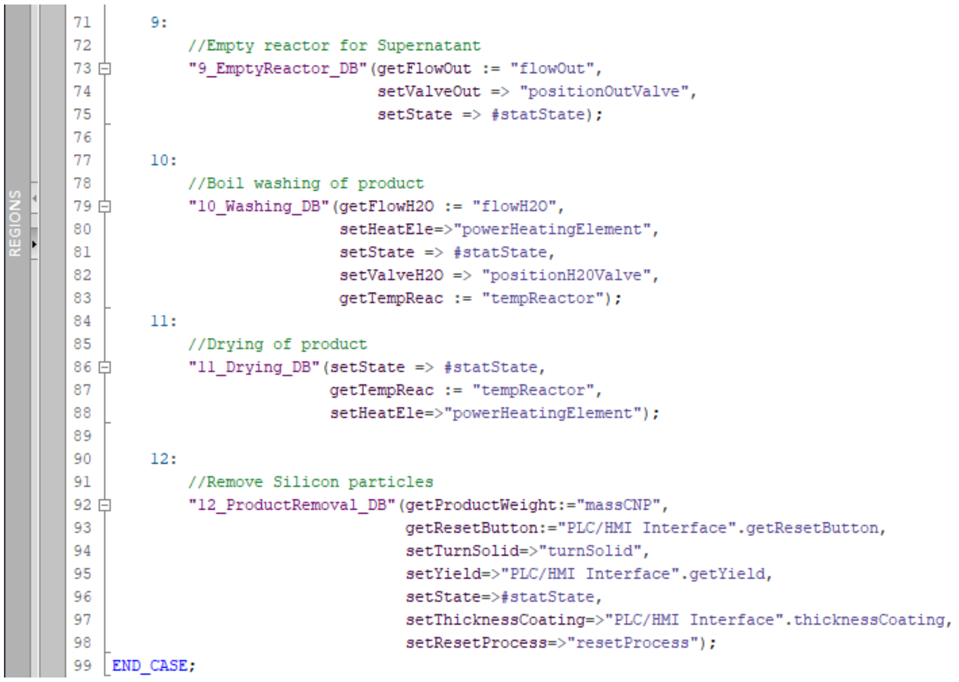
Figure D.0.4: Coating Time


ProcessSequence

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...
1	Input				☐	☐	☐
2	getStatusStartButton	Bool	0.0	false	✓	✓	✓

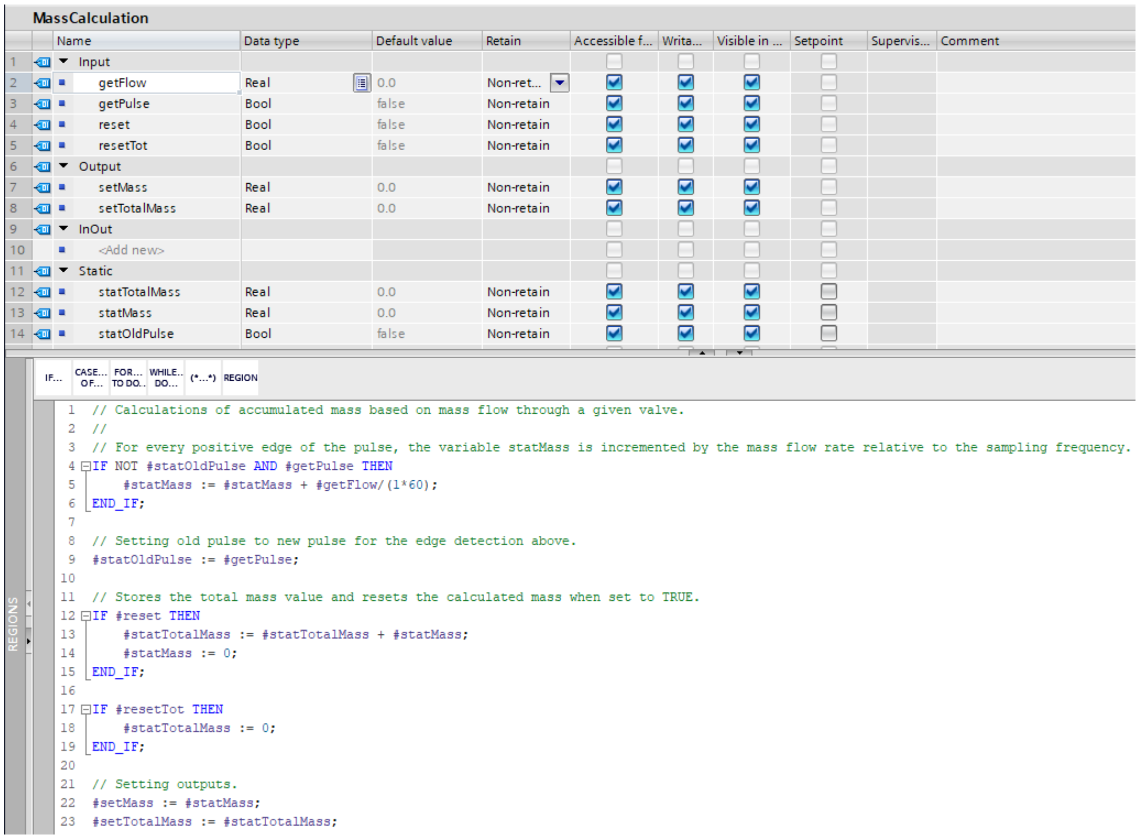
```

CASE #statState OF
    0:
        IF #getStatusStartButton THEN
            #statState := 1;
            #setResetProcess := FALSE;
        END_IF;

    1: //Additon of EtOH and H2O
        "1_EtOHandH2O_DB"(getFlowEtOH := "flowEtOH",
                          getFlowH2O := "flowH2O",
                          setValveEtOH => "positionEtOHValve",
                          setValveH2O => "positionH2OValve",
                          setState => #statState);

    2: //Addition of CTACl and stirring for 15 min
        "2_CTACl_DB"(getFlowCTACl := "flowCTACl",
                     openValveCTACl => "positionCTAClValve",
                     setMixingEngine => "speedMixingEngine",
                     setState => #statState);

    3: //Addition of SiO2NPs and sonicator ON
        "3_Silicon_DB"(getFlowSiO2NPs := "flowSiO2NPs",
                       setValveSiO2NPs => "positionSiO2NPsValve",
                       setSonication => "statusSonicator",
                       setState => #statState);

    4: //Heating of reactor contents to 70 celsius while stirring
        "4_Heating70_DB"(getTempReac := "tempReactor",
                         setTempReg => "startTempReg",
                         setState => #statState);

    5: //Additon of Resorcinol and Ammonia
        "5_ResandAm_DB"(getFlowAm := "flowAm",
                        getFlowRes := "flowRes",
                        setTempReg=>"startTempReg",
                        setTempRegComp=>"StartTempReg2",
                        setValveAm => "positionAmValve",
                        setState => #statState,
                        setResReg => "startConcReg");

    6:
        //Addition of Formaldehyde
        "6_Form_DB"(getFlowForm := "flowForm",
                    getThickness := "PLC/HMI Interface".thicknessCoating,
                    setValveRes=>"positionResValve",
                    setTempReg=>"startTempReg",
                    setTempRegComp=>"StartTempReg2",
                    setValveForm => "positionFormValve",
                    getCoatingMin := "PLC/HMI Interface".setCoatingMin,
                    getCoatingSec := "PLC/HMI Interface".setCoatingSec,
                    setState => #statState,
                    setStartResReg => "startConcReg",
                    setHeatEle=>"powerHeatingElement",
                    setCoatedETMin=>"PLC/HMI Interface".getCoatedETMin,
                    setCoatedETSec=>"PLC/HMI Interface".getCoatedETSec);



    7: //Selection of coating thickness
        "7_CoatingThickness_DB"(getThickness := "PLC/HMI Interface".thicknessCoating,
                                setState => #statState);

    8:  //Letting the coted particles settle
        "8_Settling_DB"(getTempReac := "tempReactor",
                        setState => #statState,
                        setMixingEngine => "speedMixingEngine",
                        setTurnSolid => "turnSolid");


```

(a) ProcessSequence



```
    9:
        //Empty reactor for Supernatant
        "9_EmptyReactor_DB"(getFlowOut := "flowOut",
                            setValveOut => "positionOutValve",
                            setState => #statState);
        
    10:
        //Boil washing of product
        "10_Washing_DB"(getFlowH2O := "flowH2O",
                        setHeatEle=>"powerHeatingElement",
                        setState => #statState,
                        setValveH2O => "positionH2OValve",
                        getTempReac := "tempReactor");
    11:
        //Drying of product
        "11_Drying_DB"(setState => #statState,
                       getTempReac := "tempReactor",
                       setHeatEle=>"powerHeatingElement");
        
    12:
        //Remove Silicon particles
        "12_ProductRemoval_DB"(getProductWeight:="massCNP",
                               getResetButton:="PLC/HMI Interface".getResetButton,
                               setTurnSolid=>"turnSolid",
                               setYield=>"PLC/HMI Interface".getYield,
                               setState=>#statState,
                               setThicknessCoating=>"PLC/HMI Interface".thicknessCoating,
                               setResetProcess=>"resetProcess");
END_CASE;
```

Figure D.0.6: ProcessSequence

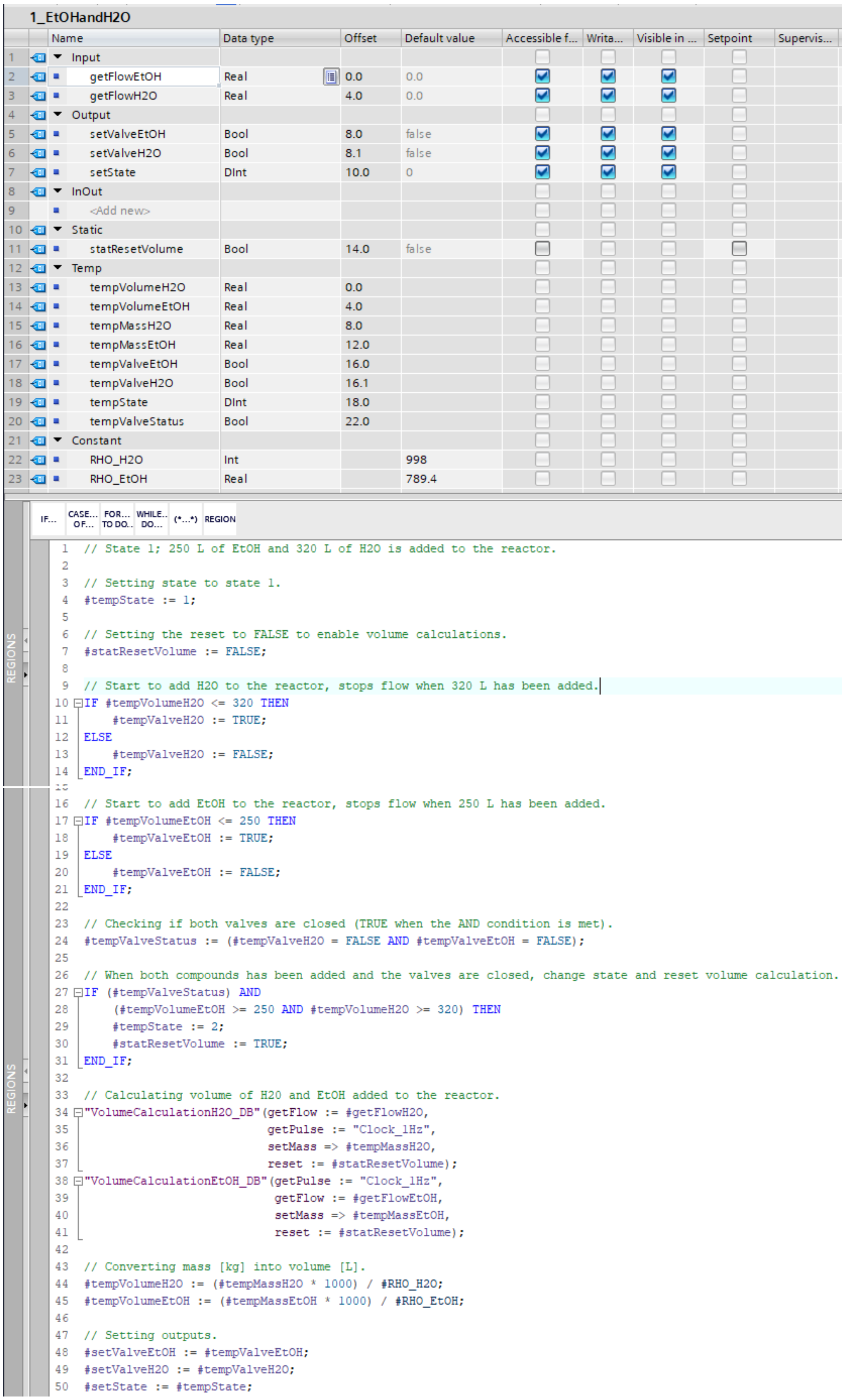


MassCalculation

	Name	Data type	Default value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	Input				☐	☐	☐	☐		
2	getFlow	Real	0.0	Non-ret...	✓	✓	✓	☐		
3	getPulse	Bool	false	Non-retain	✓	✓	✓	☐		
4	reset	Bool	false	Non-retain	✓	✓	✓	☐		
5	resetTot	Bool	false	Non-retain	✓	✓	✓	☐		
6	Output				☐	☐	☐	☐		
7	setMass	Real	0.0	Non-retain	✓	✓	✓	☐		
8	setTotalMass	Real	0.0	Non-retain	✓	✓	✓	☐		
9	InOut				☐	☐	☐	☐		
10	<Add new>				☐	☐	☐	☐		
11	Static				☐	☐	☐	☐		
12	statTotalMass	Real	0.0	Non-retain	✓	✓	✓	☐		
13	statMass	Real	0.0	Non-retain	✓	✓	✓	☐		
14	statOldPulse	Bool	false	Non-retain	✓	✓	✓	☐		

```
// Calculations of accumulated mass based on mass flow through a given valve.
//
// For every positive edge of the pulse, the variable statMass is incremented by the mass flow rate relative to the sampling frequency.
IF NOT #statOldPulse AND #getPulse THEN
    #statMass := #statMass + #getFlow/(1*60);
END_IF;

// Setting old pulse to new pulse for the edge detection above.
#statOldPulse := #getPulse;

// Stores the total mass value and resets the calculated mass when set to TRUE.
IF #reset THEN
    #statTotalMass := #statTotalMass + #statMass;
    #statMass := 0;
END_IF;

IF #resetTot THEN
    #statTotalMass := 0;
END_IF;

// Setting outputs.
#setMass := #statMass;
#setTotalMass := #statTotalMass;
```

Figure D.0.7: MassCalculation

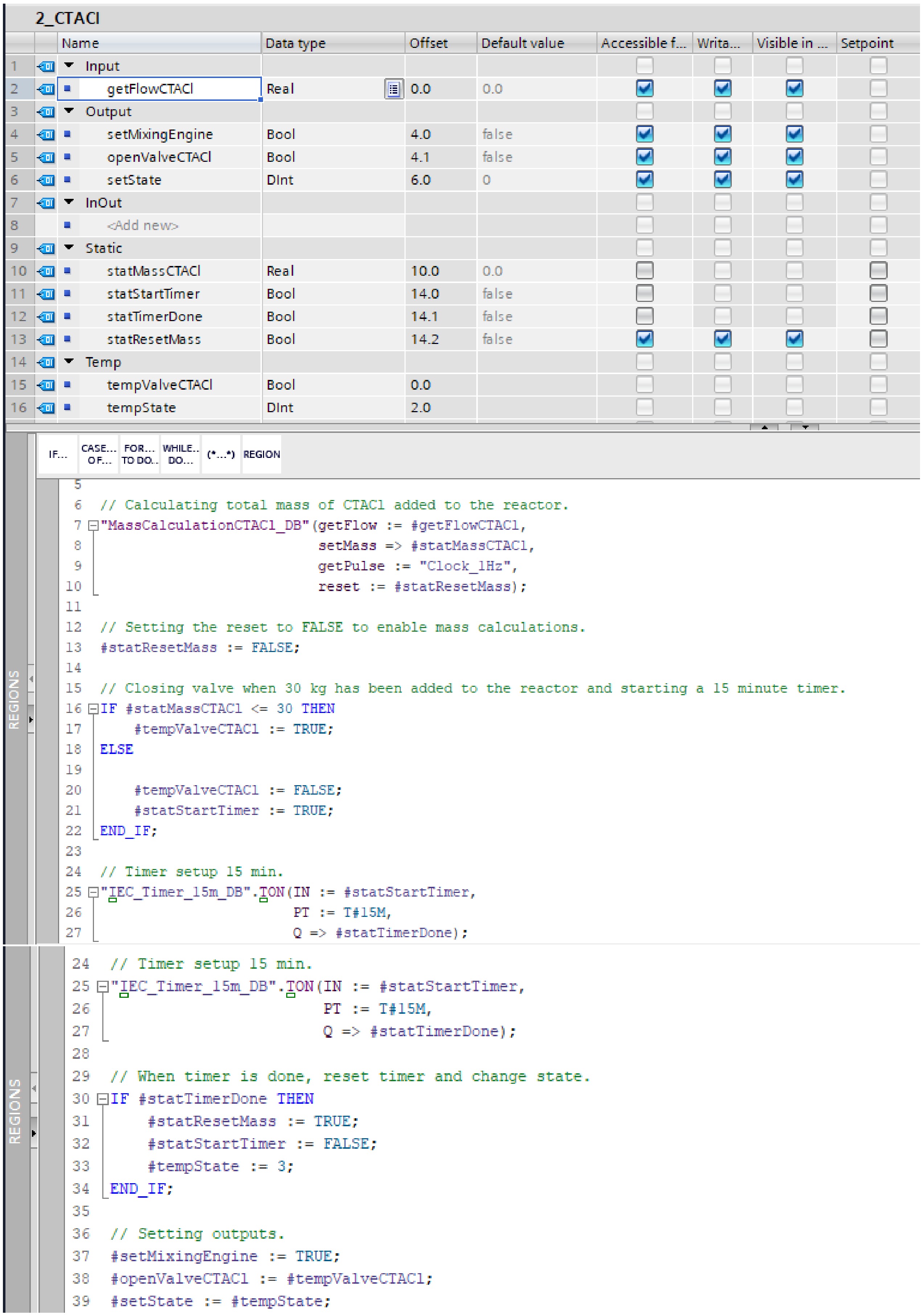


1_EtOHandH2O

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...
1	Input				☐	☐	☐	☐	
2	getFlowEtOH	Real	0.0	0.0	✓	✓	✓	☐	
3	getFlowH2O	Real	4.0	0.0	✓	✓	✓	☐	
4	Output				☐	☐	☐	☐	
5	setValveEtOH	Bool	8.0	false	✓	✓	✓	☐	
6	setValveH2O	Bool	8.1	false	✓	✓	✓	☐	
7	setState	DInt	10.0	0	✓	✓	✓	☐	
8	InOut				☐	☐	☐	☐	
9	<Add new>				☐	☐	☐	☐	
10	Static				☐	☐	☐	☐	
11	statResetVolume	Bool	14.0	false	☐	☐	☐	☐	
12	Temp				☐	☐	☐	☐	
13	tempVolumeH2O	Real	0.0		☐	☐	☐	☐	
14	tempVolumeEtOH	Real	4.0		☐	☐	☐	☐	
15	tempMassH2O	Real	8.0		☐	☐	☐	☐	
16	tempMassEtOH	Real	12.0		☐	☐	☐	☐	
17	tempValveEtOH	Bool	16.0		☐	☐	☐	☐	
18	tempValveH2O	Bool	16.1		☐	☐	☐	☐	
19	tempState	DInt	18.0		☐	☐	☐	☐	
20	tempValveStatus	Bool	22.0		☐	☐	☐	☐	
21	Constant				☐	☐	☐	☐	
22	RHO_H2O	Int		998	☐	☐	☐	☐	
23	RHO_EtOH	Real		789.4	☐	☐	☐	☐	

```
// State 1; 250 L of EtOH and 320 L of H2O is added to the reactor.

// Setting state to state 1.
#tempState := 1;

// Setting the reset to FALSE to enable volume calculations.
#statResetVolume := FALSE;

// Start to add H2O to the reactor, stops flow when 320 L has been added.
IF #tempVolumeH2O <= 320 THEN
    #tempValveH2O := TRUE;
ELSE
    #tempValveH2O := FALSE;
END_IF;

// Start to add EtOH to the reactor, stops flow when 250 L has been added.
IF #tempVolumeEtOH <= 250 THEN
    #tempValveEtOH := TRUE;
ELSE
    #tempValveEtOH := FALSE;
END_IF;

// Checking if both valves are closed (TRUE when the AND condition is met).
#tempValveStatus := (#tempValveH2O = FALSE AND #tempValveEtOH = FALSE);

// When both compounds has been added and the valves are closed, change state and reset volume calculation.
IF (#tempValveStatus) AND
    (#tempVolumeEtOH >= 250 AND #tempVolumeH2O >= 320) THEN
    #tempState := 2;
    #statResetVolume := TRUE;
END_IF;

// Calculating volume of H2O and EtOH added to the reactor.
"VolumeCalculationH2O_DB"(getFlow := #getFlowH2O,
                          getPulse := "Clock_1Hz",
                          setMass => #tempMassH2O,
                          reset := #statResetVolume);
"VolumeCalculationEtOH_DB"(getPulse := "Clock_1Hz",
                           getFlow := #getFlowEtOH,
                           setMass => #tempMassEtOH,
                           reset := #statResetVolume);

// Converting mass [kg] into volume [L].
#tempVolumeH2O := (#tempMassH2O * 1000) / #RHO_H2O;
#tempVolumeEtOH := (#tempMassEtOH * 1000) / #RHO_EtOH;

// Setting outputs.
#setValveEtOH := #tempValveEtOH;
#setValveH2O := #tempValveH2O;
#setState := #tempState;
```

(a) 1_EtOHandH2O



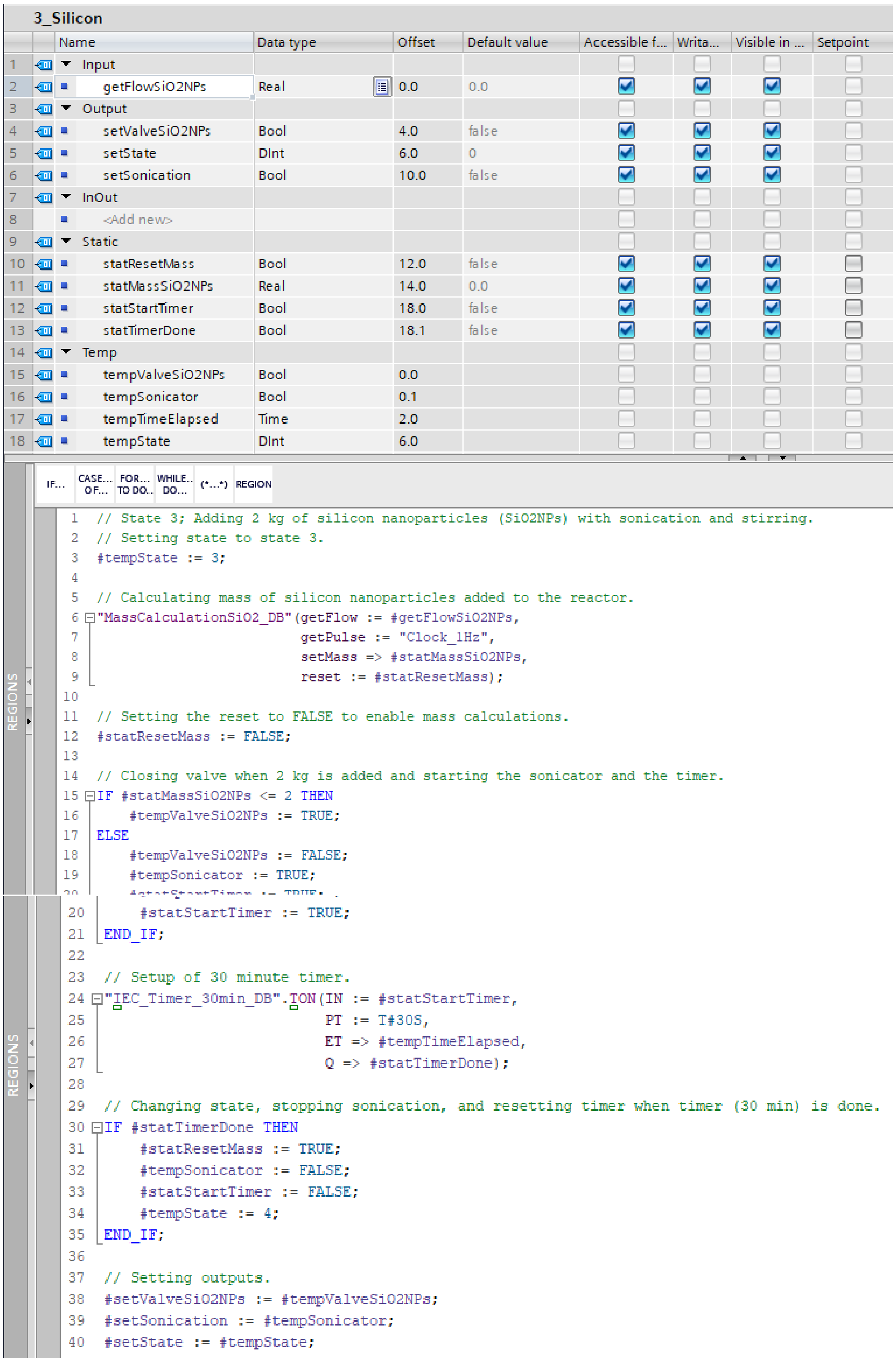
2_CTACl

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input							
2	getFlowCTACl	Real	0.0	0.0	✓	✓	✓	
3	Output							
4	setMixingEngine	Bool	4.0	false	✓	✓	✓	
5	openValveCTACl	Bool	4.1	false	✓	✓	✓	
6	setState	DInt	6.0	0	✓	✓	✓	
7	InOut							
8	<Add new>							
9	Static							
10	statMassCTACl	Real	10.0	0.0				
11	statStartTimer	Bool	14.0	false				
12	statTimerDone	Bool	14.1	false				
13	statResetMass	Bool	14.2	false	✓	✓	✓	
14	Temp							
15	tempValveCTACl	Bool	0.0					
16	tempState	DInt	2.0					

```
// Calculating total mass of CTACl added to the reactor.
"MassCalculationCTACl_DB"(getFlow := #getFlowCTACl,
                          setMass => #statMassCTACl,
                          getPulse := "Clock_1Hz",
                          reset := #statResetMass);

// Setting the reset to FALSE to enable mass calculations.
#statResetMass := FALSE;

// Closing valve when 30 kg has been added to the reactor and starting a 15 minute timer.
IF #statMassCTACl <= 30 THEN
    #tempValveCTACl := TRUE;
ELSE

    #tempValveCTACl := FALSE;
    #statStartTimer := TRUE;
END_IF;

// Timer setup 15 min.
"IEC_Timer_15m_DB".TON(IN := #statStartTimer,
                       PT := T#15M,
                       Q => #statTimerDone);
```

```
// Timer setup 15 min.
"IEC_Timer_15m_DB".TON(IN := #statStartTimer,
                       PT := T#15M,
                       Q => #statTimerDone);

// When timer is done, reset timer and change state.
IF #statTimerDone THEN
    #statResetMass := TRUE;
    #statStartTimer := FALSE;
    #tempState := 3;
END_IF;

// Setting outputs.
#setMixingEngine := TRUE;
#openValveCTACl := #tempValveCTACl;
#setState := #tempState;
```

(a) 2_CTACl

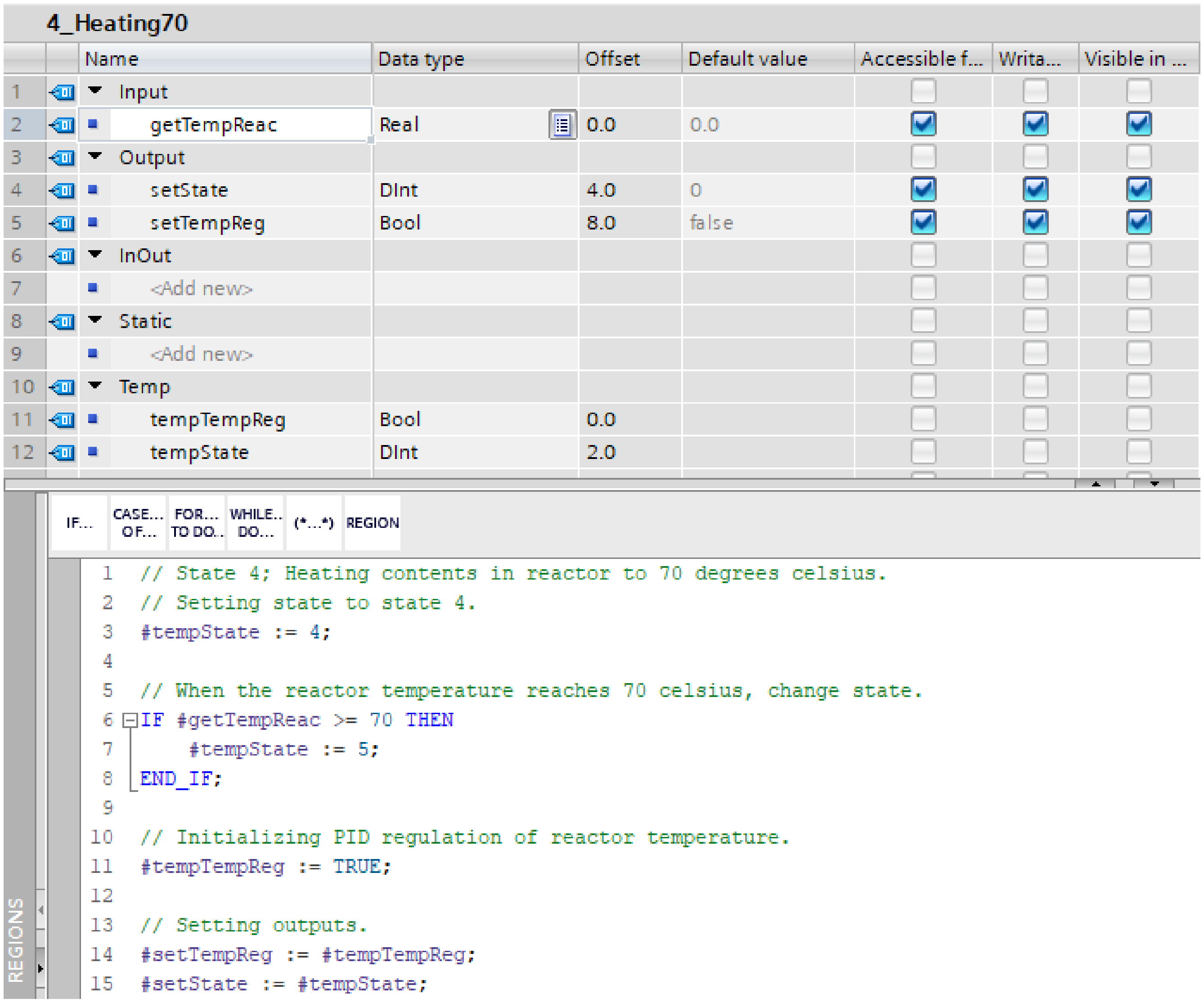


3_Silicon

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input				☐	☐	☐	☐
2	getFlowSiO2NPs	Real	0.0	0.0	✓	✓	✓	☐
3	Output				☐	☐	☐	☐
4	setValveSiO2NPs	Bool	4.0	false	✓	✓	✓	☐
5	setState	DInt	6.0	0	✓	✓	✓	☐
6	setSonication	Bool	10.0	false	✓	✓	✓	☐
7	InOut				☐	☐	☐	☐
8	<Add new>				☐	☐	☐	☐
9	Static				☐	☐	☐	☐
10	statResetMass	Bool	12.0	false	✓	✓	✓	☐
11	statMassSiO2NPs	Real	14.0	0.0	✓	✓	✓	☐
12	statStartTimer	Bool	18.0	false	✓	✓	✓	☐
13	statTimerDone	Bool	18.1	false	✓	✓	✓	☐
14	Temp				☐	☐	☐	☐
15	tempValveSiO2NPs	Bool	0.0		☐	☐	☐	☐
16	tempSonicator	Bool	0.1		☐	☐	☐	☐
17	tempTimeElapsed	Time	2.0		☐	☐	☐	☐
18	tempState	DInt	6.0		☐	☐	☐	☐

```
// State 3; Adding 2 kg of silicon nanoparticles (SiO2NPs) with sonication and stirring.
// Setting state to state 3.
#tempState := 3;

// Calculating mass of silicon nanoparticles added to the reactor.
"MassCalculationSiO2_DB"(getFlow := #getFlowSiO2NPs,
                         getPulse := "Clock_1Hz",
                         setMass => #statMassSiO2NPs,
                         reset := #statResetMass);

// Setting the reset to FALSE to enable mass calculations.
#statResetMass := FALSE;

// Closing valve when 2 kg is added and starting the sonicator and the timer.
IF #statMassSiO2NPs <= 2 THEN
    #tempValveSiO2NPs := TRUE;
ELSE
    #tempValveSiO2NPs := FALSE;
    #tempSonicator := TRUE;
    #statStartTimer := TRUE;
END_IF;

// Setup of 30 minute timer.
"IEC_Timer_30min_DB".TON(IN := #statStartTimer,
                         PT := T#30S,
                         ET => #tempTimeElapsed,
                         Q => #statTimerDone);

// Changing state, stopping sonication, and resetting timer when timer (30 min) is done.
IF #statTimerDone THEN
    #statResetMass := TRUE;
    #tempSonicator := FALSE;
    #statStartTimer := FALSE;
    #tempState := 4;
END_IF;

// Setting outputs.
#setValveSiO2NPs := #tempValveSiO2NPs;
#setSonication := #tempSonicator;
#setState := #tempState;
```

(a) 3_Silicon

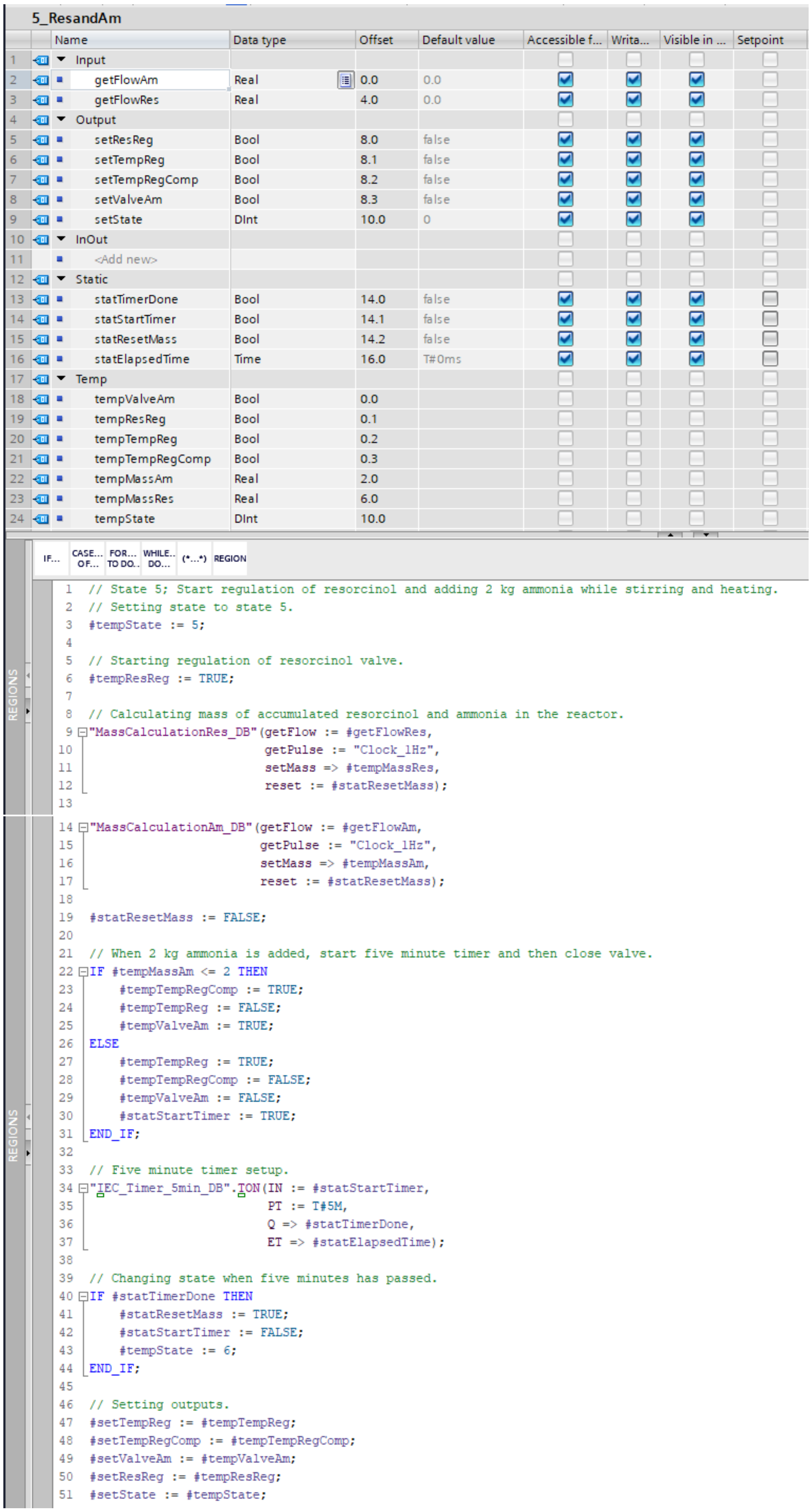


4_Heating70

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...
1	Input						
2	getTempReac	Real	0.0	0.0	✓	✓	✓
3	Output						
4	setState	DInt	4.0	0	✓	✓	✓
5	setTempReg	Bool	8.0	false	✓	✓	✓
6	InOut						
7	<Add new>						
8	Static						
9	<Add new>						
10	Temp						
11	tempTempReg	Bool	0.0				
12	tempState	DInt	2.0				

```
// State 4; Heating contents in reactor to 70 degrees celsius.
// Setting state to state 4.
#tempState := 4;

// When the reactor temperature reaches 70 celsius, change state.
IF #getTempReac >= 70 THEN
    #tempState := 5;
END_IF;

// Initializing PID regulation of reactor temperature.
#tempTempReg := TRUE;

// Setting outputs.
#setTempReg := #tempTempReg;
#setState := #tempState;
```

Figure D.0.11: 4_Heating70

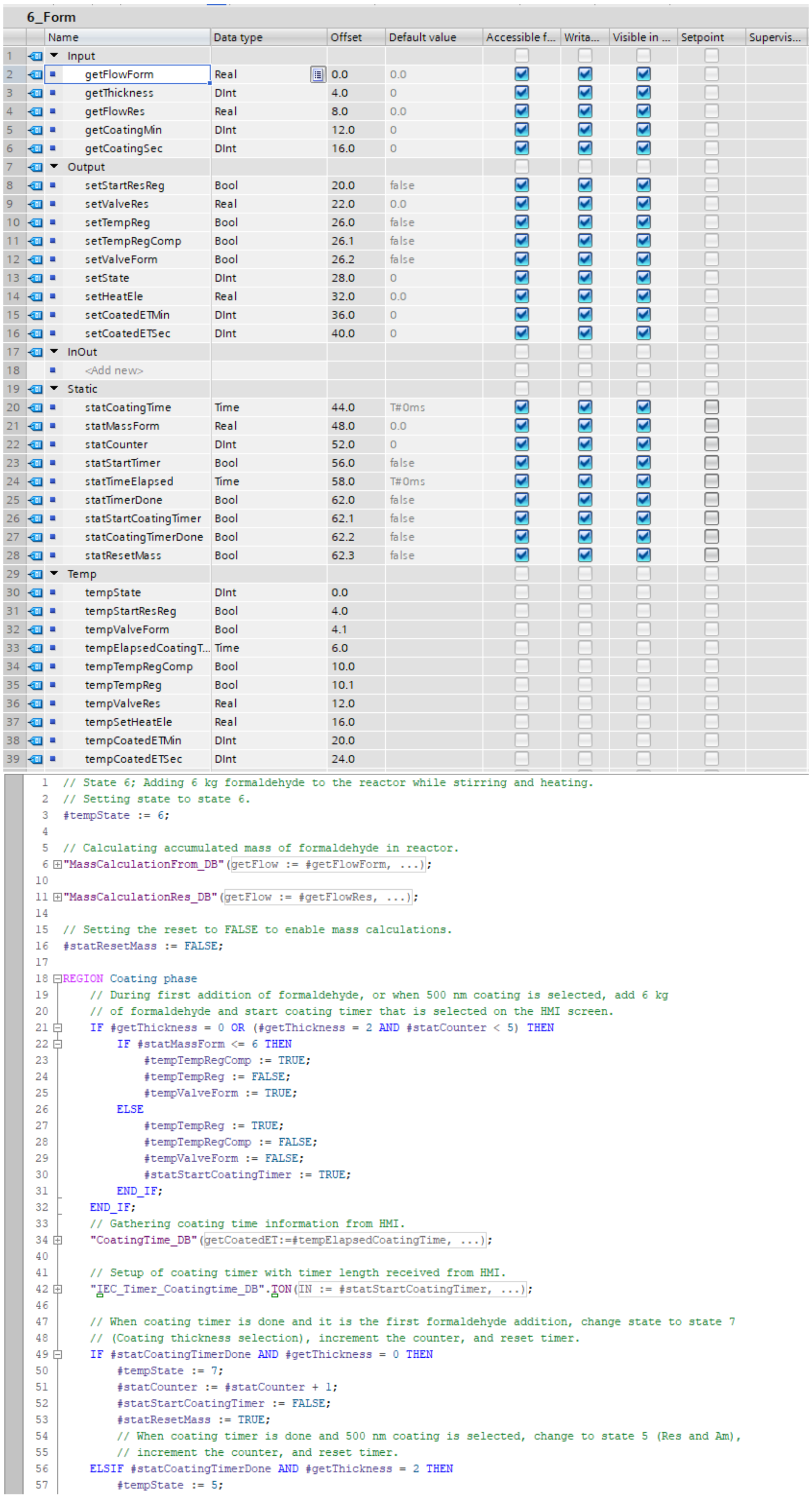


5_ResandAm

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	▼ Input				☐	☐	☐	☐
2	getFlowAm	Real	0.0	0.0	✓	✓	✓	☐
3	getFlowRes	Real	4.0	0.0	✓	✓	✓	☐
4	▼ Output				☐	☐	☐	☐
5	setResReg	Bool	8.0	false	✓	✓	✓	☐
6	setTempReg	Bool	8.1	false	✓	✓	✓	☐
7	setTempRegComp	Bool	8.2	false	✓	✓	✓	☐
8	setValveAm	Bool	8.3	false	✓	✓	✓	☐
9	setState	DInt	10.0	0	✓	✓	✓	☐
10	▼ InOut				☐	☐	☐	☐
11	<Add new>				☐	☐	☐	☐
12	▼ Static				☐	☐	☐	☐
13	statTimerDone	Bool	14.0	false	✓	✓	✓	☐
14	statStartTimer	Bool	14.1	false	✓	✓	✓	☐
15	statResetMass	Bool	14.2	false	✓	✓	✓	☐
16	statElapsedTime	Time	16.0	T#0ms	✓	✓	✓	☐
17	▼ Temp				☐	☐	☐	☐
18	tempValveAm	Bool	0.0		☐	☐	☐	☐
19	tempResReg	Bool	0.1		☐	☐	☐	☐
20	tempTempReg	Bool	0.2		☐	☐	☐	☐
21	tempTempRegComp	Bool	0.3		☐	☐	☐	☐
22	tempMassAm	Real	2.0		☐	☐	☐	☐
23	tempMassRes	Real	6.0		☐	☐	☐	☐
24	tempState	DInt	10.0		☐	☐	☐	☐

```
// State 5; Start regulation of resorcinol and adding 2 kg ammonia while stirring and heating.
// Setting state to state 5.
#tempState := 5;

// Starting regulation of resorcinol valve.
#tempResReg := TRUE;

// Calculating mass of accumulated resorcinol and ammonia in the reactor.
"MassCalculationRes_DB"(getFlow := #getFlowRes,
                        getPulse := "Clock_1Hz",
                        setMass => #tempMassRes,
                        reset := #statResetMass);

"MassCalculationAm_DB"(getFlow := #getFlowAm,
                       getPulse := "Clock_1Hz",
                       setMass => #tempMassAm,
                       reset := #statResetMass);

#statResetMass := FALSE;

// When 2 kg ammonia is added, start five minute timer and then close valve.
IF #tempMassAm <= 2 THEN
    #tempTempRegComp := TRUE;
    #tempTempReg := FALSE;
    #tempValveAm := TRUE;
ELSE
    #tempTempReg := TRUE;
    #tempTempRegComp := FALSE;
    #tempValveAm := FALSE;
    #statStartTimer := TRUE;
END_IF;

// Five minute timer setup.
"IEC_Timer_5min_DB".TON(IN := #statStartTimer,
                        PT := T#5M,
                        Q => #statTimerDone,
                        ET => #statElapsedTime);

// Changing state when five minutes has passed.
IF #statTimerDone THEN
    #statResetMass := TRUE;
    #statStartTimer := FALSE;
    #tempState := 6;
END_IF;

// Setting outputs.
#setTempReg := #tempTempReg;
#setTempRegComp := #tempTempRegComp;
#setValveAm := #tempValveAm;
#setResReg := #tempResReg;
#setState := #tempState;
```

(a) 5_ResandAm

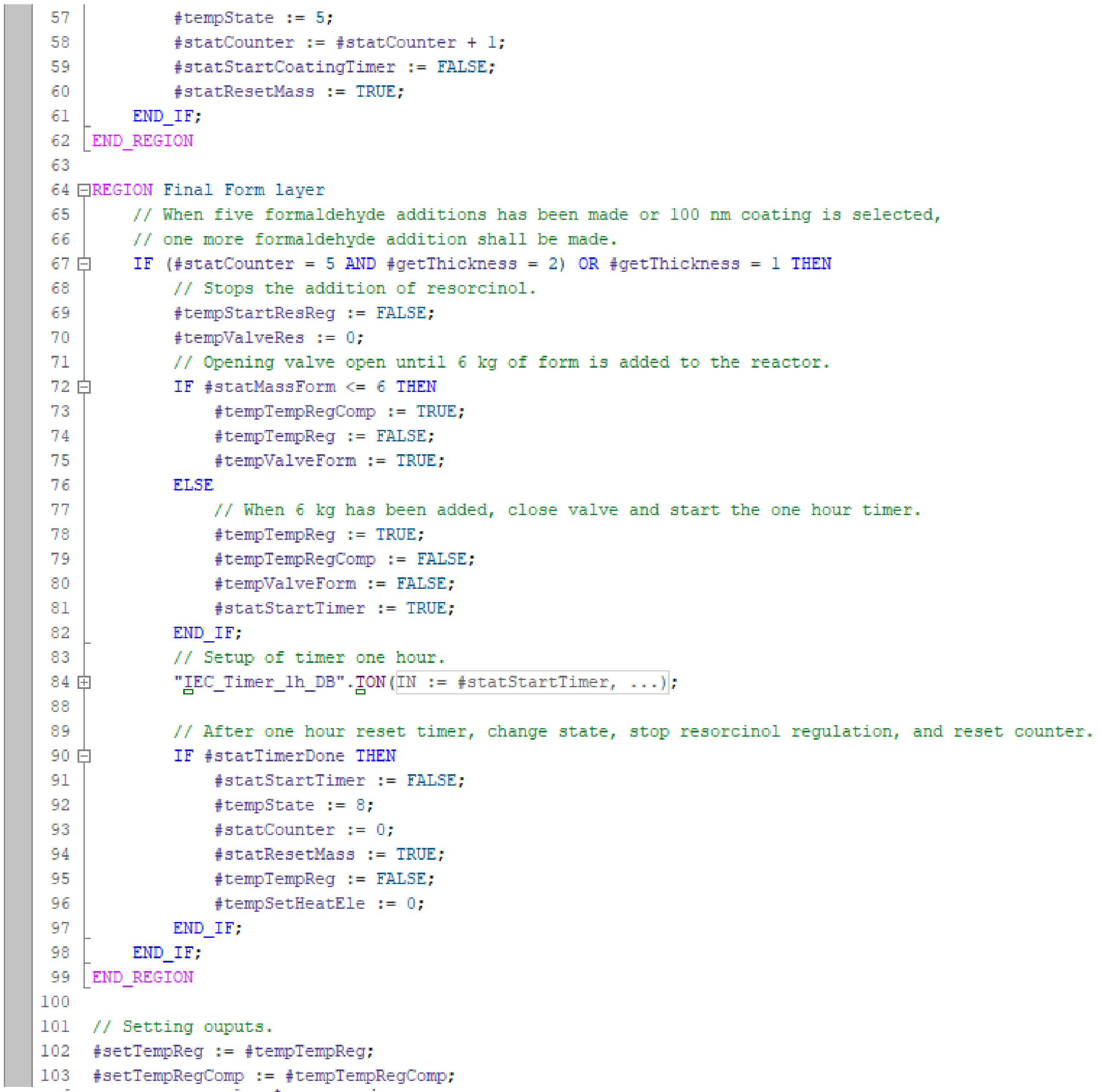


6_Form

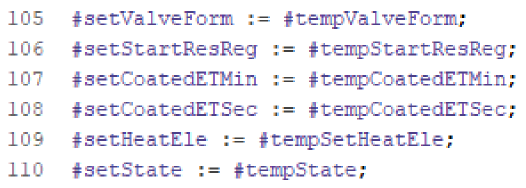
	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...
1	▼ Input				☐	☐	☐	☐	
2	getFlowForm	Real	0.0	0.0	✓	✓	✓	☐	
3	getThickness	DInt	4.0	0	✓	✓	✓	☐	
4	getFlowRes	Real	8.0	0.0	✓	✓	✓	☐	
5	getCoatingMin	DInt	12.0	0	✓	✓	✓	☐	
6	getCoatingSec	DInt	16.0	0	✓	✓	✓	☐	
7	▼ Output				☐	☐	☐	☐	
8	setStartResReg	Bool	20.0	false	✓	✓	✓	☐	
9	setValveRes	Real	22.0	0.0	✓	✓	✓	☐	
10	setTempReg	Bool	26.0	false	✓	✓	✓	☐	
11	setTempRegComp	Bool	26.1	false	✓	✓	✓	☐	
12	setValveForm	Bool	26.2	false	✓	✓	✓	☐	
13	setState	DInt	28.0	0	✓	✓	✓	☐	
14	setHeatEle	Real	32.0	0.0	✓	✓	✓	☐	
15	setCoatedETMin	DInt	36.0	0	✓	✓	✓	☐	
16	setCoatedETSec	DInt	40.0	0	✓	✓	✓	☐	
17	▼ InOut				☐	☐	☐	☐	
18	<Add new>				☐	☐	☐	☐	
19	▼ Static				☐	☐	☐	☐	
20	statCoatingTime	Time	44.0	T#0ms	✓	✓	✓	☐	
21	statMassForm	Real	48.0	0.0	✓	✓	✓	☐	
22	statCounter	DInt	52.0	0	✓	✓	✓	☐	
23	statStartTimer	Bool	56.0	false	✓	✓	✓	☐	
24	statTimeElapsed	Time	58.0	T#0ms	✓	✓	✓	☐	
25	statTimerDone	Bool	62.0	false	✓	✓	✓	☐	
26	statStartCoatingTimer	Bool	62.1	false	✓	✓	✓	☐	
27	statCoatingTimerDone	Bool	62.2	false	✓	✓	✓	☐	
28	statResetMass	Bool	62.3	false	✓	✓	✓	☐	
29	▼ Temp				☐	☐	☐	☐	
30	tempState	DInt	0.0		☐	☐	☐	☐	
31	tempStartResReg	Bool	4.0		☐	☐	☐	☐	
32	tempValveForm	Bool	4.1		☐	☐	☐	☐	
33	tempElapsedCoatingT...	Time	6.0		☐	☐	☐	☐	
34	tempTempRegComp	Bool	10.0		☐	☐	☐	☐	
35	tempTempReg	Bool	10.1		☐	☐	☐	☐	
36	tempValveRes	Real	12.0		☐	☐	☐	☐	
37	tempSetHeatEle	Real	16.0		☐	☐	☐	☐	
38	tempCoatedETMin	DInt	20.0		☐	☐	☐	☐	
39	tempCoatedETSec	DInt	24.0		☐	☐	☐	☐	

```
// State 6; Adding 6 kg formaldehyde to the reactor while stirring and heating.
// Setting state to state 6.
#tempState := 6;

// Calculating accumulated mass of formaldehyde in reactor.
"MassCalculationFrom_DB"(getFlow := #getFlowForm, ...);

"MassCalculationRes_DB"(getFlow := #getFlowRes, ...);

// Setting the reset to FALSE to enable mass calculations.
#statResetMass := FALSE;

REGION Coating phase
    // During first addition of formaldehyde, or when 500 nm coating is selected, add 6 kg
    // of formaldehyde and start coating timer that is selected on the HMI screen.
    IF #getThickness = 0 OR (#getThickness = 2 AND #statCounter < 5) THEN
        IF #statMassForm <= 6 THEN
            #tempTempRegComp := TRUE;
            #tempTempReg := FALSE;
            #tempValveForm := TRUE;
        ELSE
            #tempTempReg := TRUE;
            #tempTempRegComp := FALSE;
            #tempValveForm := FALSE;
            #statStartCoatingTimer := TRUE;
        END_IF;
    END_IF;
    // Gathering coating time information from HMI.
    "CoatingTime_DB"(getCoatedET:=#tempElapsedCoatingTime, ...);

    // Setup of coating timer with timer length received from HMI.
    "IEC_Timer_Coatingtime_DB".TON(IN := #statStartCoatingTimer, ...);

    // When coating timer is done and it is the first formaldehyde addition, change state to state 7
    // (Coating thickness selection), increment the counter, and reset timer.
    IF #statCoatingTimerDone AND #getThickness = 0 THEN
        #tempState := 7;
        #statCounter := #statCounter + 1;
        #statStartCoatingTimer := FALSE;
        #statResetMass := TRUE;
        // When coating timer is done and 500 nm coating is selected, change to state 5 (Res and Am),
        // increment the counter, and reset timer.
    ELSIF #statCoatingTimerDone AND #getThickness = 2 THEN
        #tempState := 5;
```

(a) 6_Form

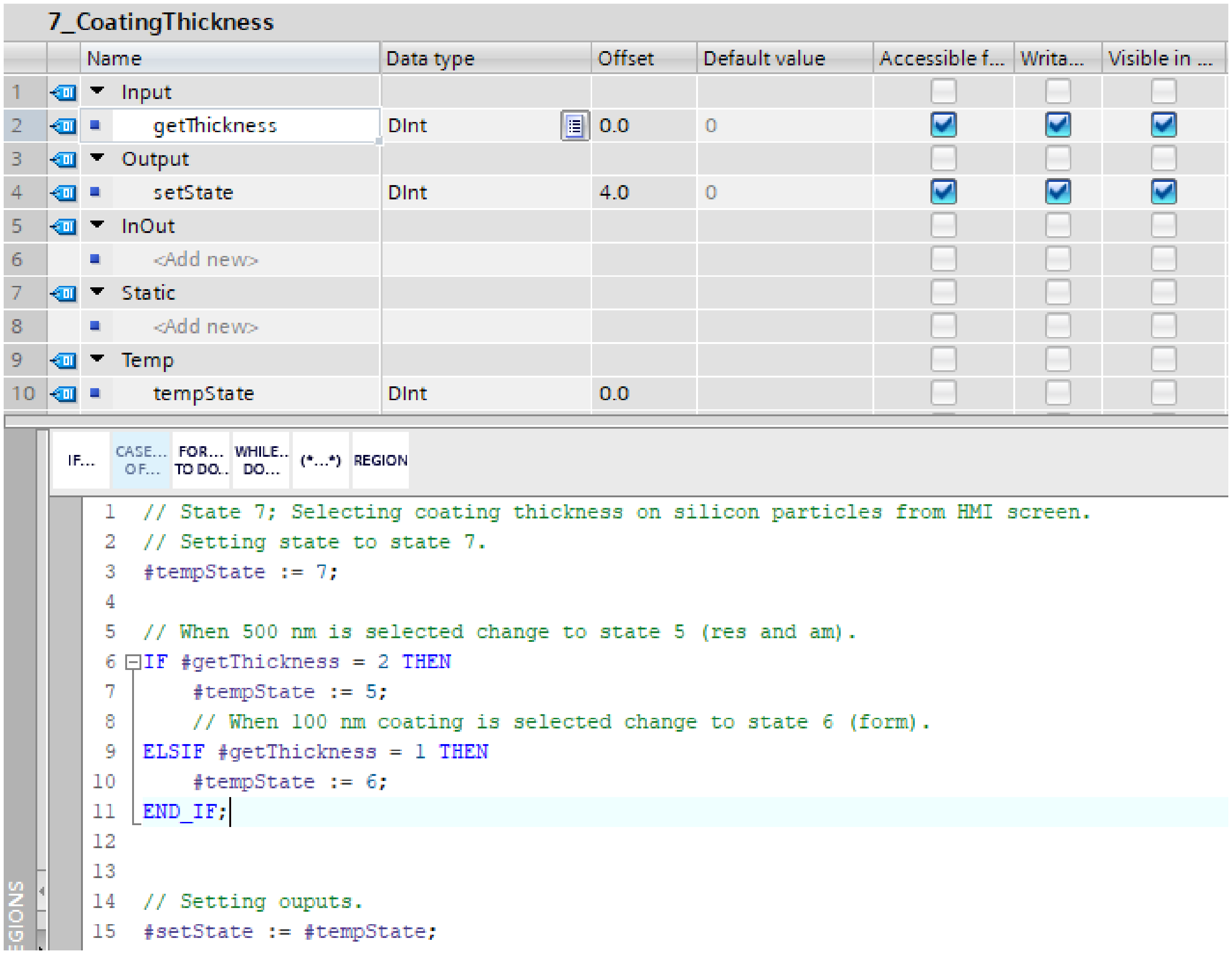


```
        #tempState := 5;
        #statCounter := #statCounter + 1;
        #statStartCoatingTimer := FALSE;
        #statResetMass := TRUE;
    END_IF;
END_REGION

REGION Final Form layer
    // When five formaldehyde additions has been made or 100 nm coating is selected,
    // one more formaldehyde addition shall be made.
    IF (#statCounter = 5 AND #getThickness = 2) OR #getThickness = 1 THEN
        // Stops the addition of resorcinol.
        #tempStartResReg := FALSE;
        #tempValveRes := 0;
        // Opening valve open until 6 kg of form is added to the reactor.
        IF #statMassForm <= 6 THEN
            #tempTempRegComp := TRUE;
            #tempTempReg := FALSE;
            #tempValveForm := TRUE;
        ELSE
            // When 6 kg has been added, close valve and start the one hour timer.
            #tempTempReg := TRUE;
            #tempTempRegComp := FALSE;
            #tempValveForm := FALSE;
            #statStartTimer := TRUE;
        END_IF;
        // Setup of timer one hour.
        "IEC_Timer_1h_DB".TON(IN := #statStartTimer, ...);

        // After one hour reset timer, change state, stop resorcinol regulation, and reset counter.
        IF #statTimerDone THEN
            #statStartTimer := FALSE;
            #tempState := 8;
            #statCounter := 0;
            #statResetMass := TRUE;
            #tempTempReg := FALSE;
            #tempSetHeatEle := 0;
        END_IF;
    END_IF;
END_REGION

// Setting ouputs.
#setTempReg := #tempTempReg;
#setTempRegComp := #tempTempRegComp;
```



```
#setValveForm := #tempValveForm;
#setStartResReg := #tempStartResReg;
#setCoatedETMin := #tempCoatedETMin;
#setCoatedETSec := #tempCoatedETSec;
#setHeatEle := #tempSetHeatEle;
#setState := #tempState;
```

(a) 6_Form

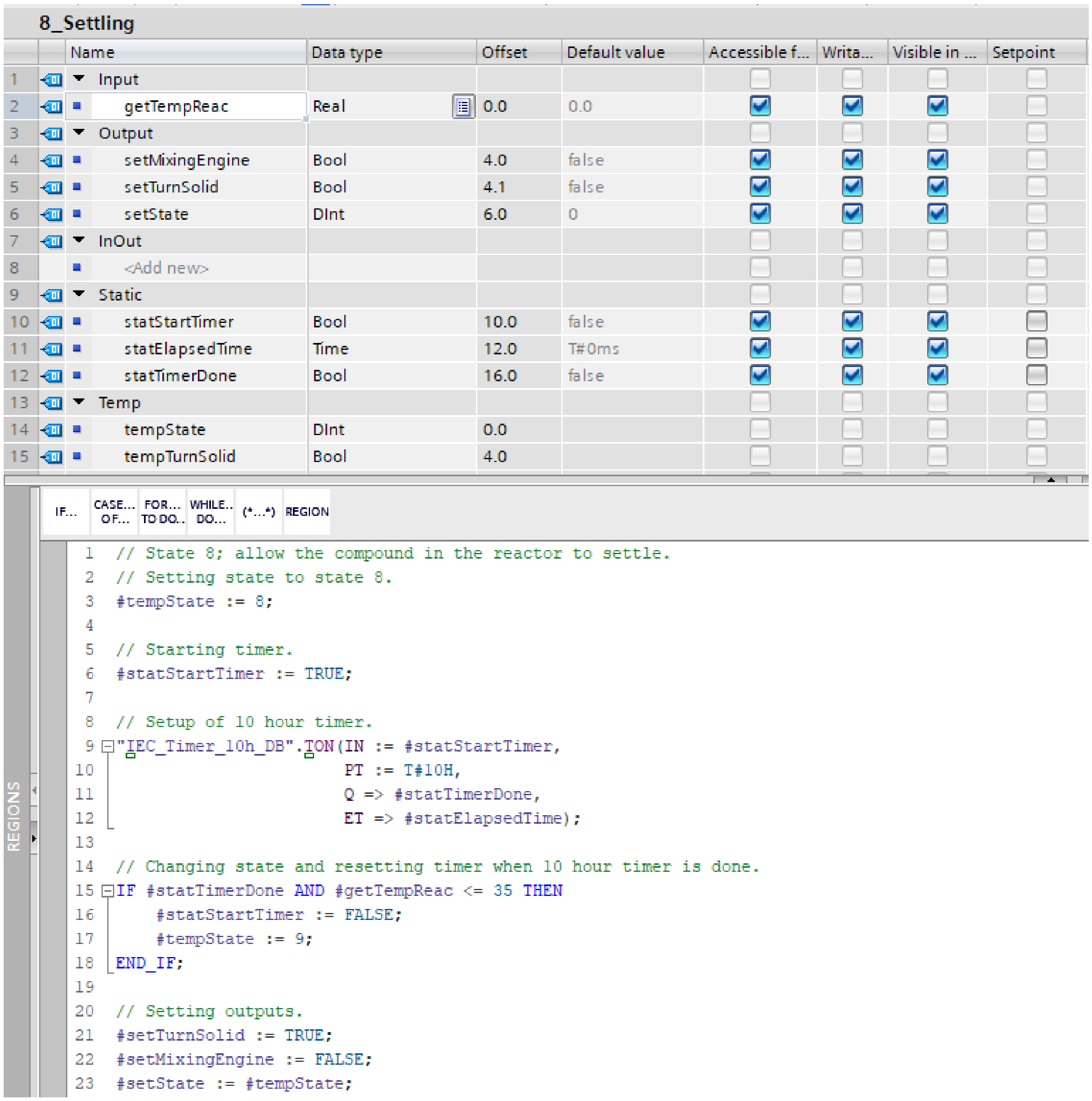


7_CoatingThickness

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...
1	Input				☐	☐	☐
2	getThickness	DInt	0.0	0	✓	✓	✓
3	Output				☐	☐	☐
4	setState	DInt	4.0	0	✓	✓	✓
5	InOut				☐	☐	☐
6	<Add new>				☐	☐	☐
7	Static				☐	☐	☐
8	<Add new>				☐	☐	☐
9	Temp				☐	☐	☐
10	tempState	DInt	0.0		☐	☐	☐

```
// State 7; Selecting coating thickness on silicon particles from HMI screen.
// Setting state to state 7.
#tempState := 7;

// When 500 nm is selected change to state 5 (res and am).
IF #getThickness = 2 THEN
    #tempState := 5;
    // When 100 nm coating is selected change to state 6 (form).
ELSIF #getThickness = 1 THEN
    #tempState := 6;
END_IF;


// Setting ouputs.
#setState := #tempState;
```

Figure D.0.15: 7_CoatingThickness

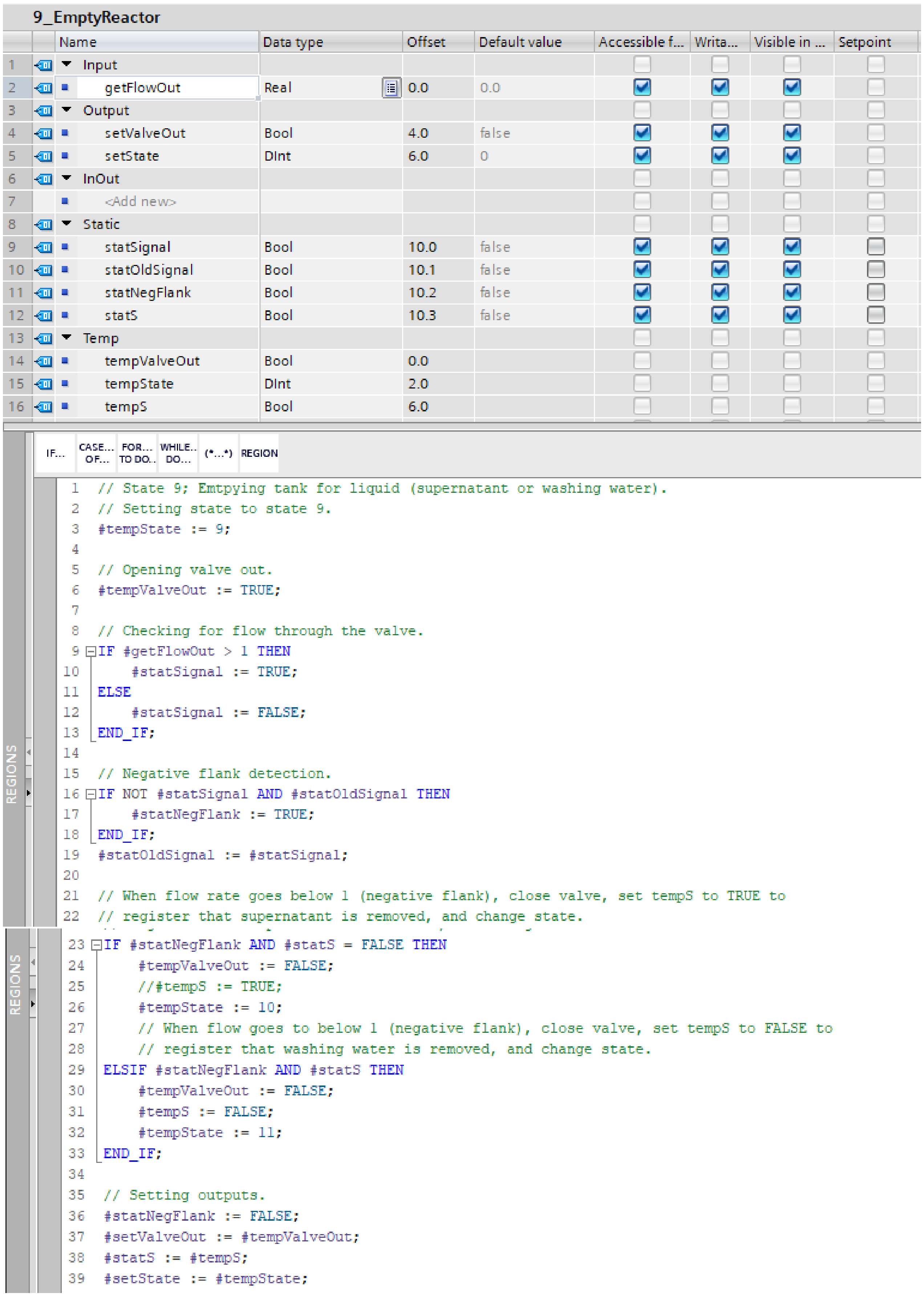


8_Settling

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input				☐	☐	☐	☐
2	getTempReac	Real	0.0	0.0	✓	✓	✓	☐
3	Output				☐	☐	☐	☐
4	setMixingEngine	Bool	4.0	false	✓	✓	✓	☐
5	setTurnSolid	Bool	4.1	false	✓	✓	✓	☐
6	setState	DInt	6.0	0	✓	✓	✓	☐
7	InOut				☐	☐	☐	☐
8	<Add new>				☐	☐	☐	☐
9	Static				☐	☐	☐	☐
10	statStartTimer	Bool	10.0	false	✓	✓	✓	☐
11	statElapsedTime	Time	12.0	T#0ms	✓	✓	✓	☐
12	statTimerDone	Bool	16.0	false	✓	✓	✓	☐
13	Temp				☐	☐	☐	☐
14	tempState	DInt	0.0		☐	☐	☐	☐
15	tempTurnSolid	Bool	4.0		☐	☐	☐	☐

```
// State 8; allow the compound in the reactor to settle.
// Setting state to state 8.
#tempState := 8;

// Starting timer.
#statStartTimer := TRUE;

// Setup of 10 hour timer.
"IEC_Timer_10h_DB".TON(IN := #statStartTimer,
                       PT := T#10H,
                       Q => #statTimerDone,
                       ET => #statElapsedTime);

// Changing state and resetting timer when 10 hour timer is done.
IF #statTimerDone AND #getTempReac <= 35 THEN
    #statStartTimer := FALSE;
    #tempState := 9;
END_IF;

// Setting outputs.
#setTurnSolid := TRUE;
#setMixingEngine := FALSE;
#setState := #tempState;
```

Figure D.0.16: 8_Settling

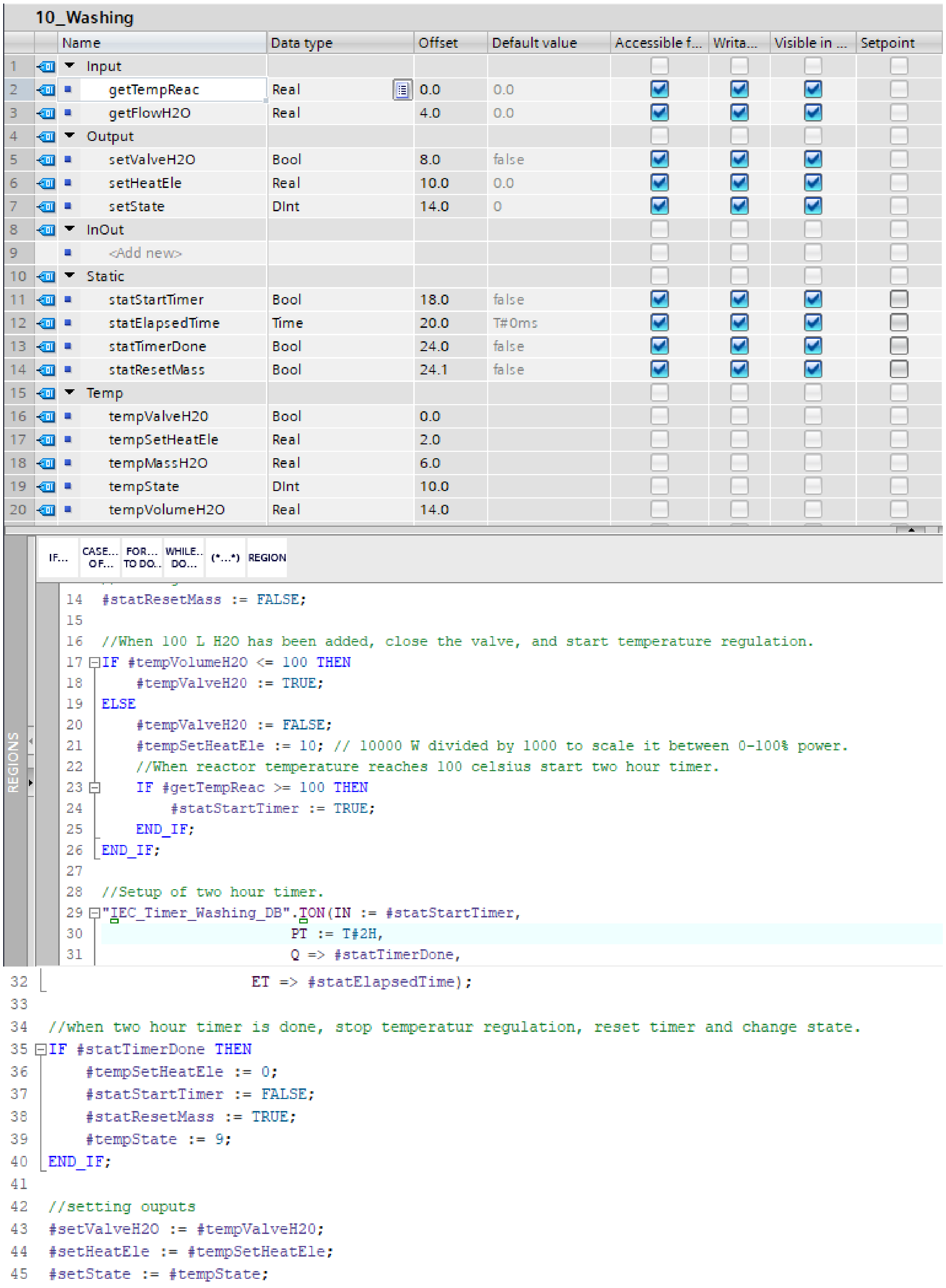


9_EmptyReactor

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input							
2	getFlowOut	Real	0.0	0.0	✓	✓	✓	
3	Output							
4	setValveOut	Bool	4.0	false	✓	✓	✓	
5	setState	DInt	6.0	0	✓	✓	✓	
6	InOut							
7	<Add new>							
8	Static							
9	statSignal	Bool	10.0	false	✓	✓	✓	
10	statOldSignal	Bool	10.1	false	✓	✓	✓	
11	statNegFlank	Bool	10.2	false	✓	✓	✓	
12	statS	Bool	10.3	false	✓	✓	✓	
13	Temp							
14	tempValveOut	Bool	0.0					
15	tempState	DInt	2.0					
16	tempS	Bool	6.0					

```
// State 9; Emtpying tank for liquid (supernatant or washing water).
// Setting state to state 9.
#tempState := 9;

// Opening valve out.
#tempValveOut := TRUE;

// Checking for flow through the valve.
IF #getFlowOut > 1 THEN
    #statSignal := TRUE;
ELSE
    #statSignal := FALSE;
END_IF;

// Negative flank detection.
IF NOT #statSignal AND #statOldSignal THEN
    #statNegFlank := TRUE;
END_IF;
#statOldSignal := #statSignal;

// When flow rate goes below 1 (negative flank), close valve, set tempS to TRUE to
// register that supernatant is removed, and change state.
IF #statNegFlank AND #statS = FALSE THEN
    #tempValveOut := FALSE;
    //#tempS := TRUE;
    #tempState := 10;
    // When flow goes to below 1 (negative flank), close valve, set tempS to FALSE to
    // register that washing water is removed, and change state.
ELSIF #statNegFlank AND #statS THEN
    #tempValveOut := FALSE;
    #tempS := FALSE;
    #tempState := 11;
END_IF;

// Setting outputs.
#statNegFlank := FALSE;
#setValveOut := #tempValveOut;
#statS := #tempS;
#setState := #tempState;
```

(a) 9_EmptyReactor

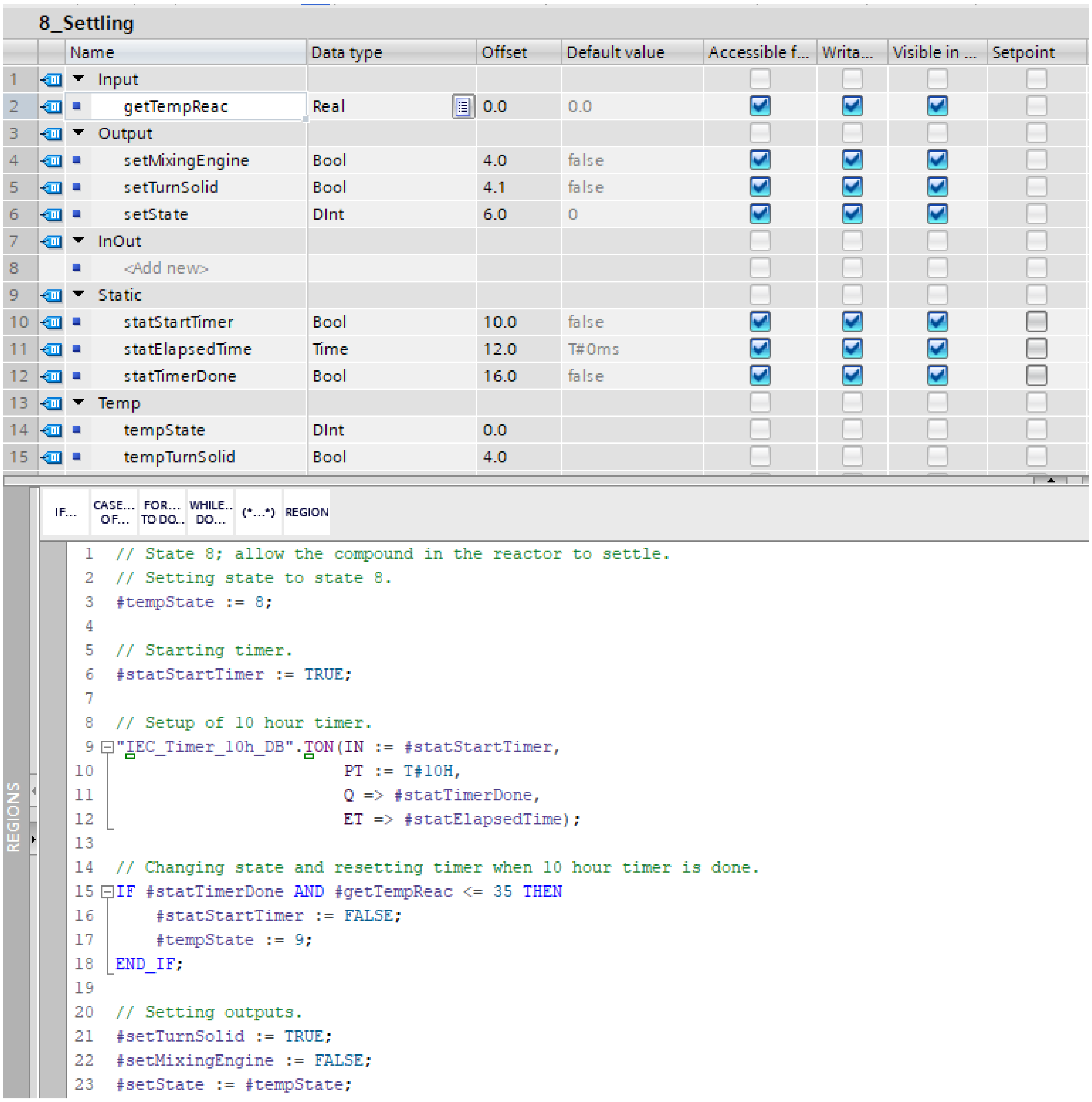


10_Washing

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input							
2	getTempReac	Real	0.0	0.0	✓	✓	✓	
3	getFlowH2O	Real	4.0	0.0	✓	✓	✓	
4	Output							
5	setValveH2O	Bool	8.0	false	✓	✓	✓	
6	setHeatEle	Real	10.0	0.0	✓	✓	✓	
7	setState	DInt	14.0	0	✓	✓	✓	
8	InOut							
9	<Add new>							
10	Static							
11	statStartTimer	Bool	18.0	false	✓	✓	✓	
12	statElapsedTime	Time	20.0	T#0ms	✓	✓	✓	
13	statTimerDone	Bool	24.0	false	✓	✓	✓	
14	statResetMass	Bool	24.1	false	✓	✓	✓	
15	Temp							
16	tempValveH20	Bool	0.0					
17	tempSetHeatEle	Real	2.0					
18	tempMassH2O	Real	6.0					
19	tempState	DInt	10.0					
20	tempVolumeH2O	Real	14.0					

```
#statResetMass := FALSE;

//When 100 L H2O has been added, close the valve, and start temperature regulation.
IF #tempVolumeH2O <= 100 THEN
    #tempValveH20 := TRUE;
ELSE
    #tempValveH20 := FALSE;
    #tempSetHeatEle := 10; // 10000 W divided by 1000 to scale it between 0-100% power.
    //When reactor temperature reaches 100 celsius start two hour timer.
    IF #getTempReac >= 100 THEN
        #statStartTimer := TRUE;
    END_IF;
END_IF;

//Setup of two hour timer.
"IEC_Timer_Washing_DB".TON(IN := #statStartTimer,
                          PT := T#2H,
                          Q => #statTimerDone,
                          ET => #statElapsedTime);

//when two hour timer is done, stop temperatur regulation, reset timer and change state.
IF #statTimerDone THEN
    #tempSetHeatEle := 0;
    #statStartTimer := FALSE;
    #statResetMass := TRUE;
    #tempState := 9;
END_IF;

//setting ouputs
#setValveH2O := #tempValveH20;
#setHeatEle := #tempSetHeatEle;
#setState := #tempState;
```

(a) 10_Washing

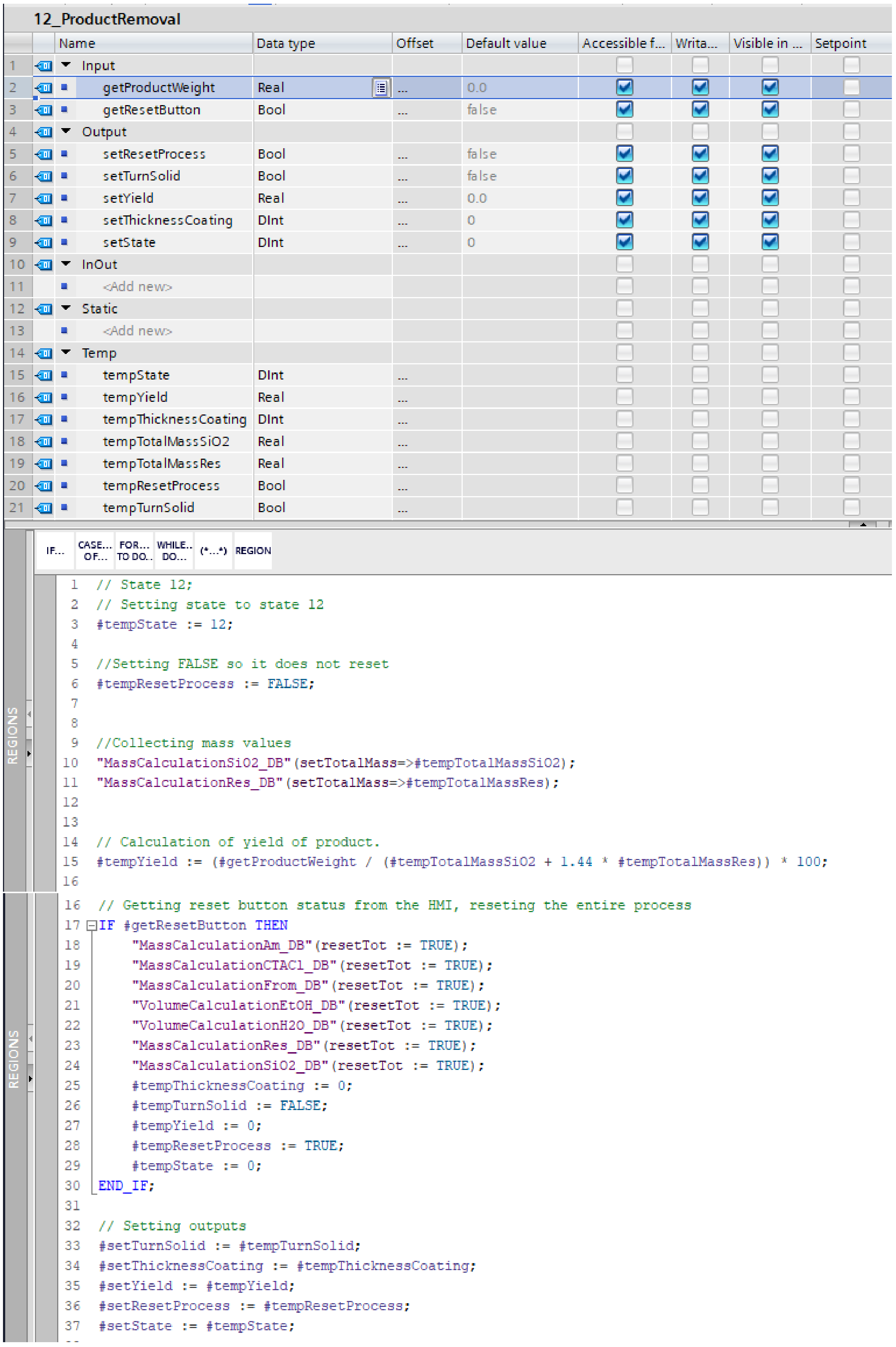


8_Settling

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Input				☐	☐	☐	☐
2	getTempReac	Real	0.0	0.0	✓	✓	✓	☐
3	Output				☐	☐	☐	☐
4	setMixingEngine	Bool	4.0	false	✓	✓	✓	☐
5	setTurnSolid	Bool	4.1	false	✓	✓	✓	☐
6	setState	DInt	6.0	0	✓	✓	✓	☐
7	InOut				☐	☐	☐	☐
8	<Add new>				☐	☐	☐	☐
9	Static				☐	☐	☐	☐
10	statStartTimer	Bool	10.0	false	✓	✓	✓	☐
11	statElapsedTime	Time	12.0	T#0ms	✓	✓	✓	☐
12	statTimerDone	Bool	16.0	false	✓	✓	✓	☐
13	Temp				☐	☐	☐	☐
14	tempState	DInt	0.0		☐	☐	☐	☐
15	tempTurnSolid	Bool	4.0		☐	☐	☐	☐

```
// State 8; allow the compound in the reactor to settle.
// Setting state to state 8.
#tempState := 8;

// Starting timer.
#statStartTimer := TRUE;

// Setup of 10 hour timer.
"IEC_Timer_10h_DB".TON(IN := #statStartTimer,
                       PT := T#10H,
                       Q => #statTimerDone,
                       ET => #statElapsedTime);

// Changing state and resetting timer when 10 hour timer is done.
IF #statTimerDone AND #getTempReac <= 35 THEN
    #statStartTimer := FALSE;
    #tempState := 9;
END_IF;

// Setting outputs.
#setTurnSolid := TRUE;
#setMixingEngine := FALSE;
#setState := #tempState;
```

Figure D.0.19: 11_Drying



12_ProductRemoval

	Name	Data type	Offset	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	▼ Input				☐	☐	☐	☐
2	getProductWeight	Real	...	0.0	✓	✓	✓	☐
3	getResetButton	Bool	...	false	✓	✓	✓	☐
4	▼ Output				☐	☐	☐	☐
5	setResetProcess	Bool	...	false	✓	✓	✓	☐
6	setTurnSolid	Bool	...	false	✓	✓	✓	☐
7	setYield	Real	...	0.0	✓	✓	✓	☐
8	setThicknessCoating	DInt	...	0	✓	✓	✓	☐
9	setState	DInt	...	0	✓	✓	✓	☐
10	▼ InOut				☐	☐	☐	☐
11	<Add new>				☐	☐	☐	☐
12	▼ Static				☐	☐	☐	☐
13	<Add new>				☐	☐	☐	☐
14	▼ Temp				☐	☐	☐	☐
15	tempState	DInt	...		☐	☐	☐	☐
16	tempYield	Real	...		☐	☐	☐	☐
17	tempThicknessCoating	DInt	...		☐	☐	☐	☐
18	tempTotalMassSiO2	Real	...		☐	☐	☐	☐
19	tempTotalMassRes	Real	...		☐	☐	☐	☐
20	tempResetProcess	Bool	...		☐	☐	☐	☐
21	tempTurnSolid	Bool	...		☐	☐	☐	☐

```
// State 12;
// Setting state to state 12
#tempState := 12;

//Setting FALSE so it does not reset
#tempResetProcess := FALSE;


//Collecting mass values
"MassCalculationSiO2_DB"(setTotalMass=>#tempTotalMassSiO2);
"MassCalculationRes_DB"(setTotalMass=>#tempTotalMassRes);


// Calculation of yield of product.
#tempYield := (#getProductWeight / (#tempTotalMassSiO2 + 1.44 * #tempTotalMassRes)) * 100;

// Getting reset button status from the HMI, reseting the entire process
IF #getResetButton THEN
    "MassCalculationAm_DB"(resetTot := TRUE);
    "MassCalculationCTACl_DB"(resetTot := TRUE);
    "MassCalculationFrom_DB"(resetTot := TRUE);
    "VolumeCalculationEtOH_DB"(resetTot := TRUE);
    "VolumeCalculationH2O_DB"(resetTot := TRUE);
    "MassCalculationRes_DB"(resetTot := TRUE);
    "MassCalculationSiO2_DB"(resetTot := TRUE);
    #tempThicknessCoating := 0;
    #tempTurnSolid := FALSE;
    #tempYield := 0;
    #tempResetProcess := TRUE;
    #tempState := 0;
END_IF;

// Setting outputs
#setTurnSolid := #tempTurnSolid;
#setThicknessCoating := #tempThicknessCoating;
#setYield := #tempYield;
#setResetProcess := #tempResetProcess;
#setState := #tempState;
```

(a) 12_ProductRemoval

Appendix E

SIMIT Charts

This appendix shows all the chart developed for the digital twin in SIMIT.

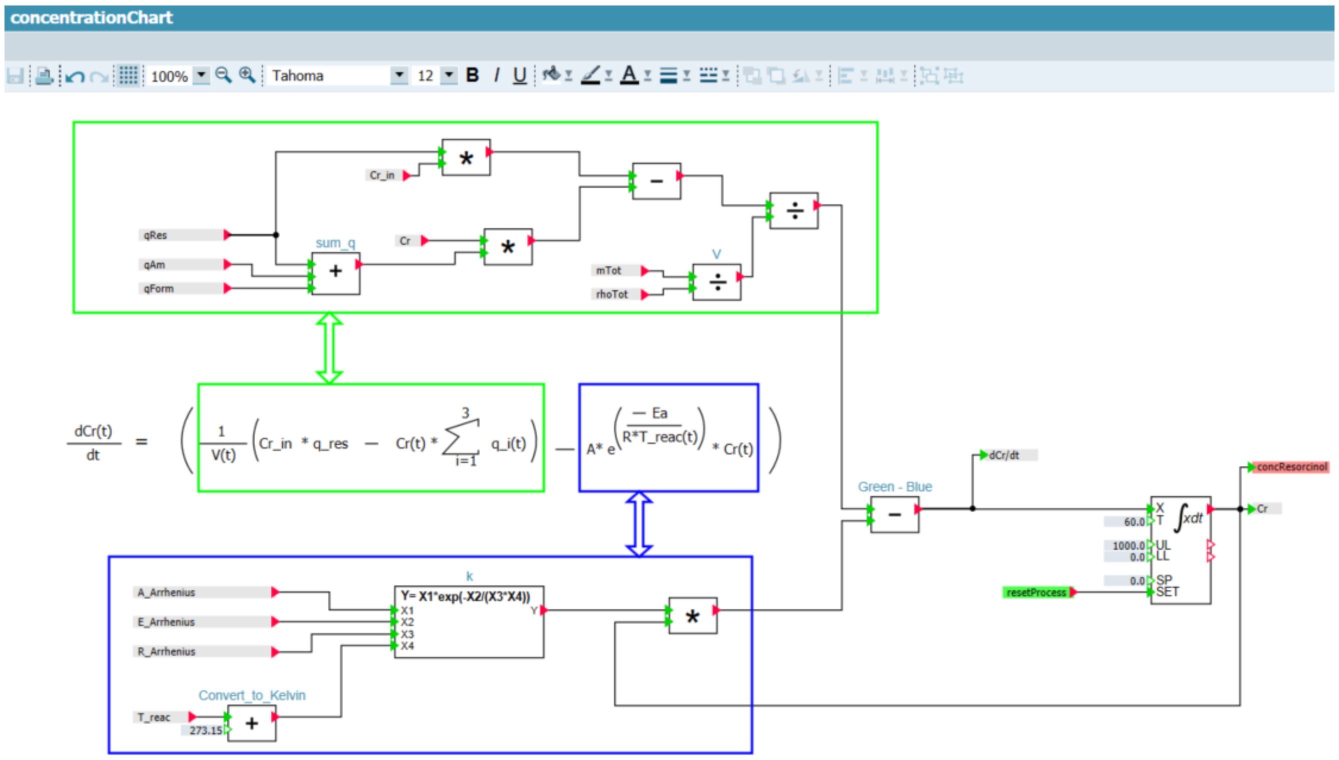


Figure E.0.1: Concentration Chart

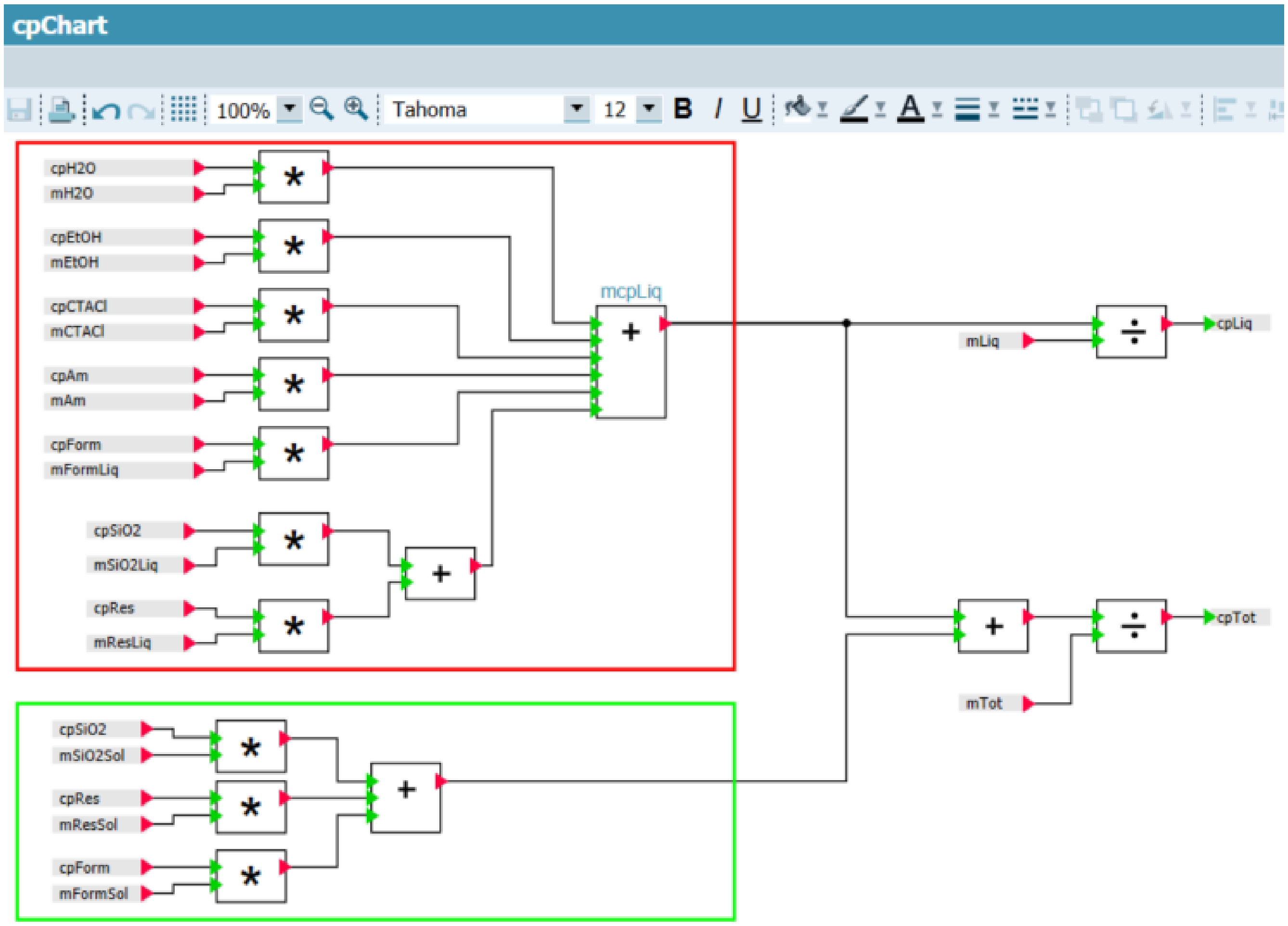


Figure E.0.2: Specific heat capacity chart

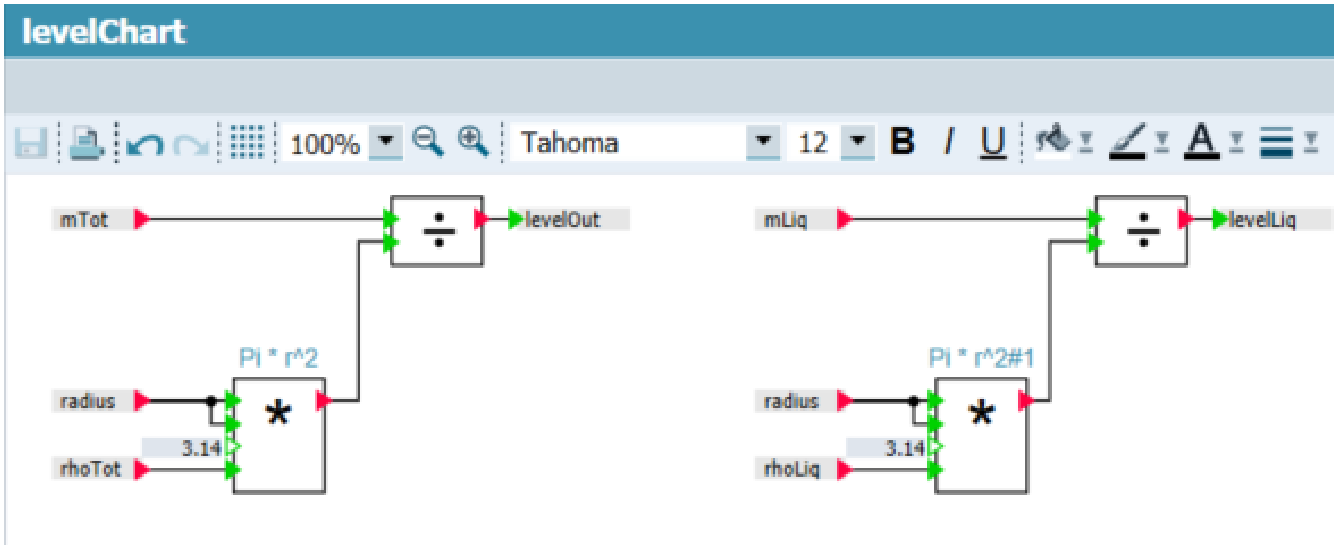


Figure E.0.3: Reactor level chart

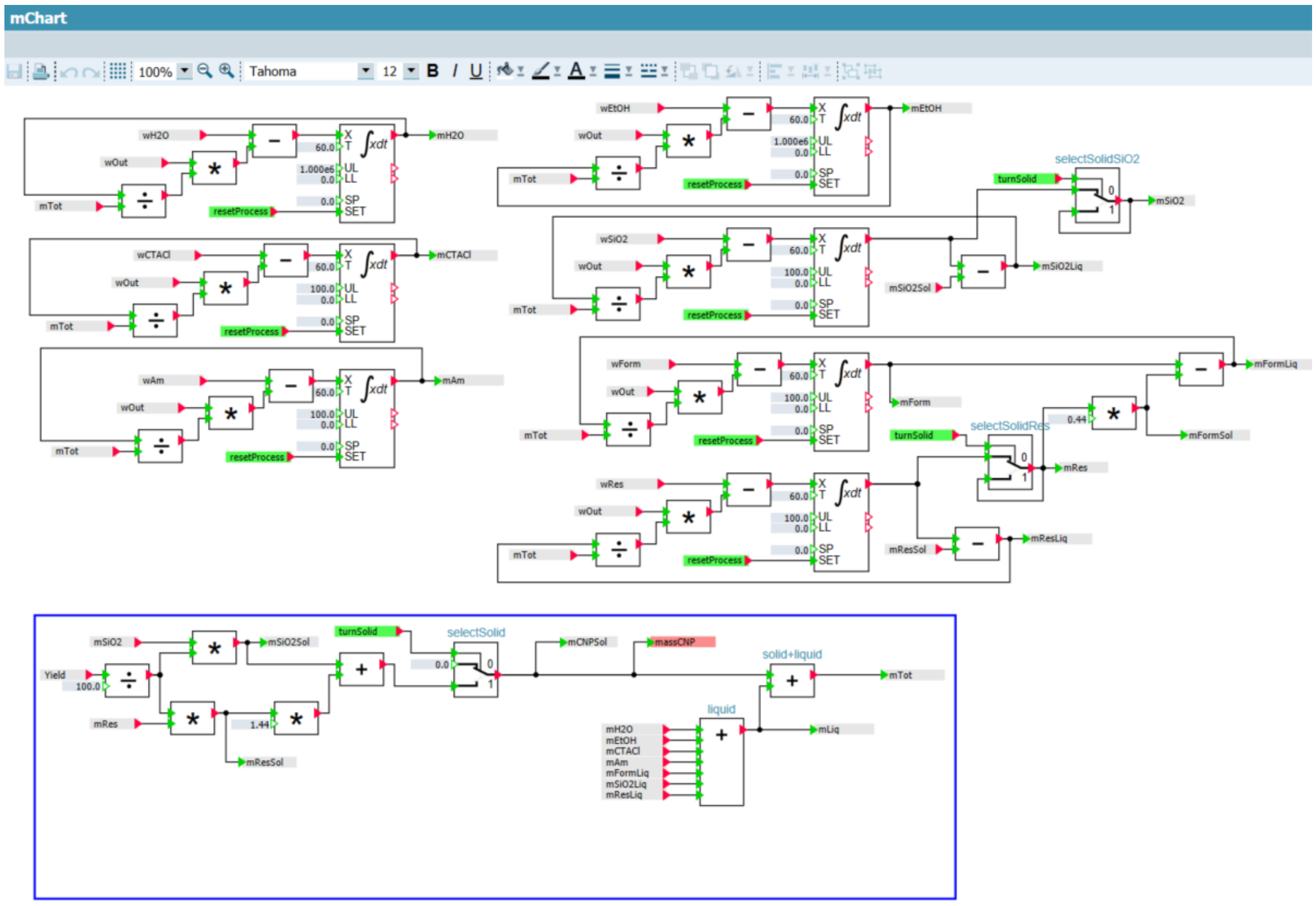


Figure E.0.4: Mass chart

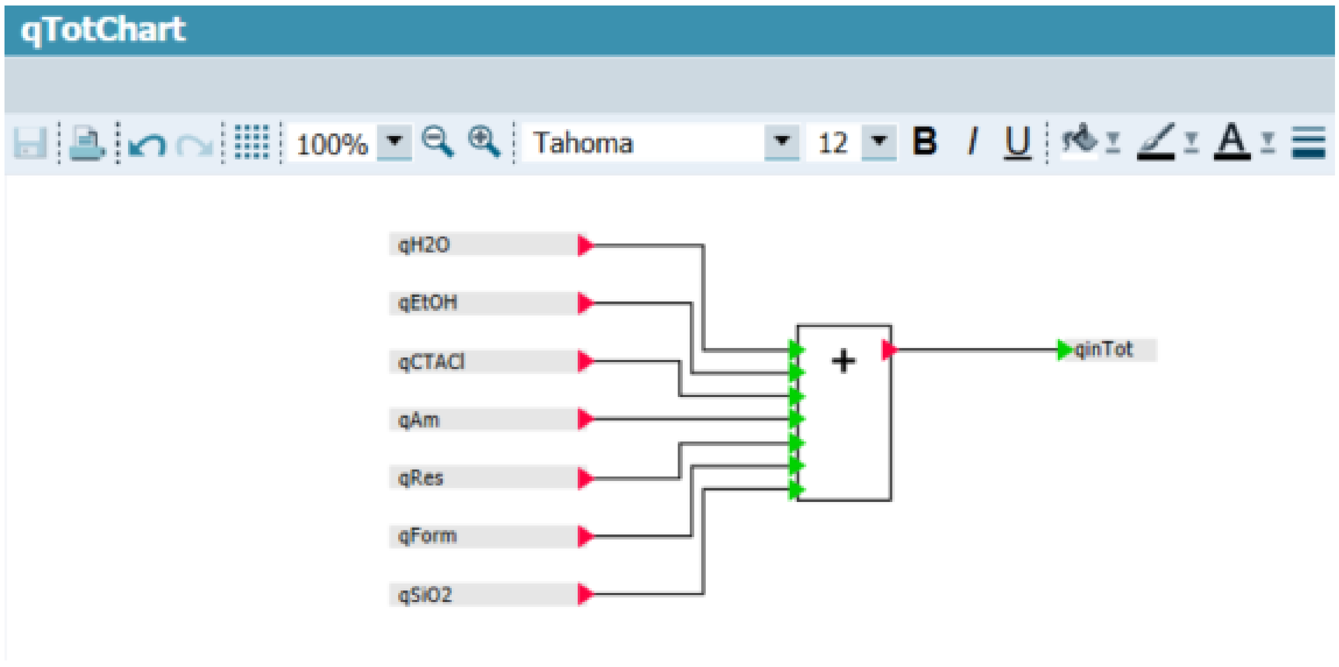


Figure E.0.5: Volumetric flow chart

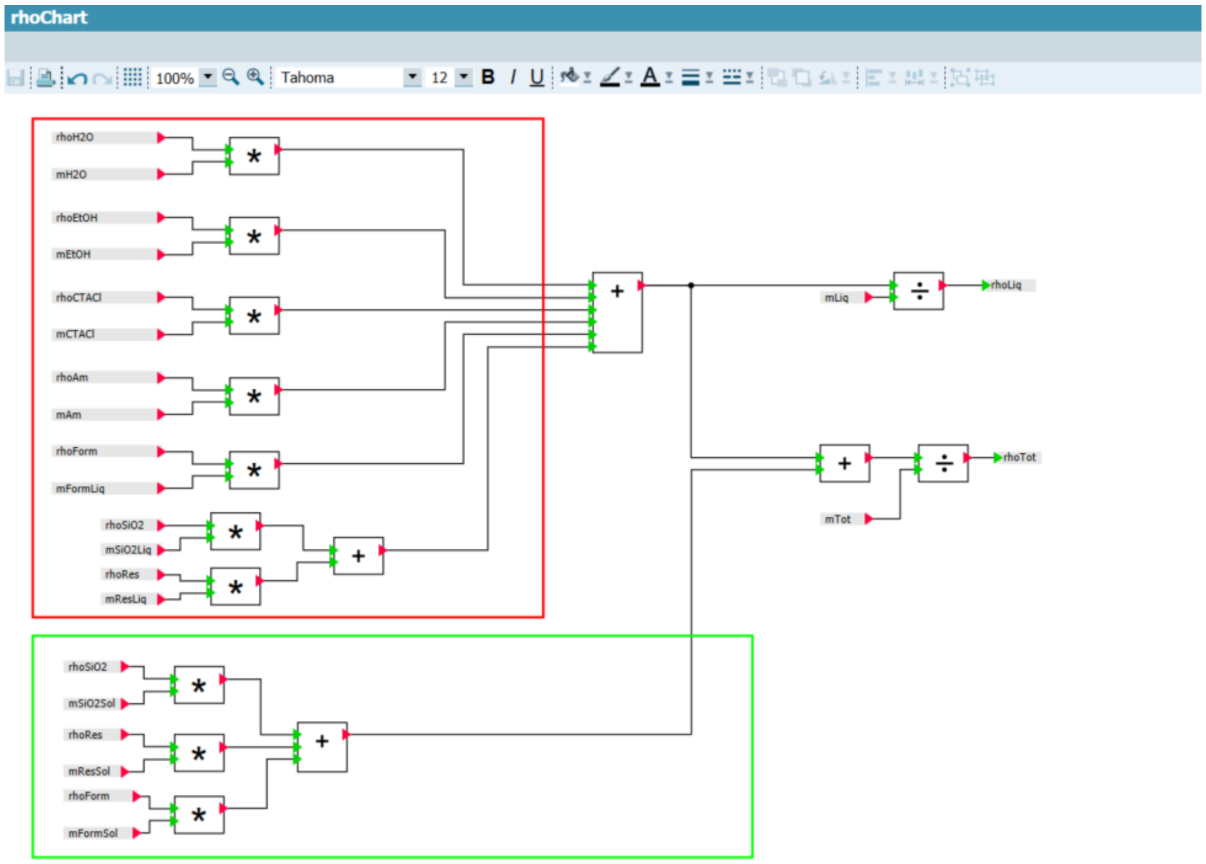
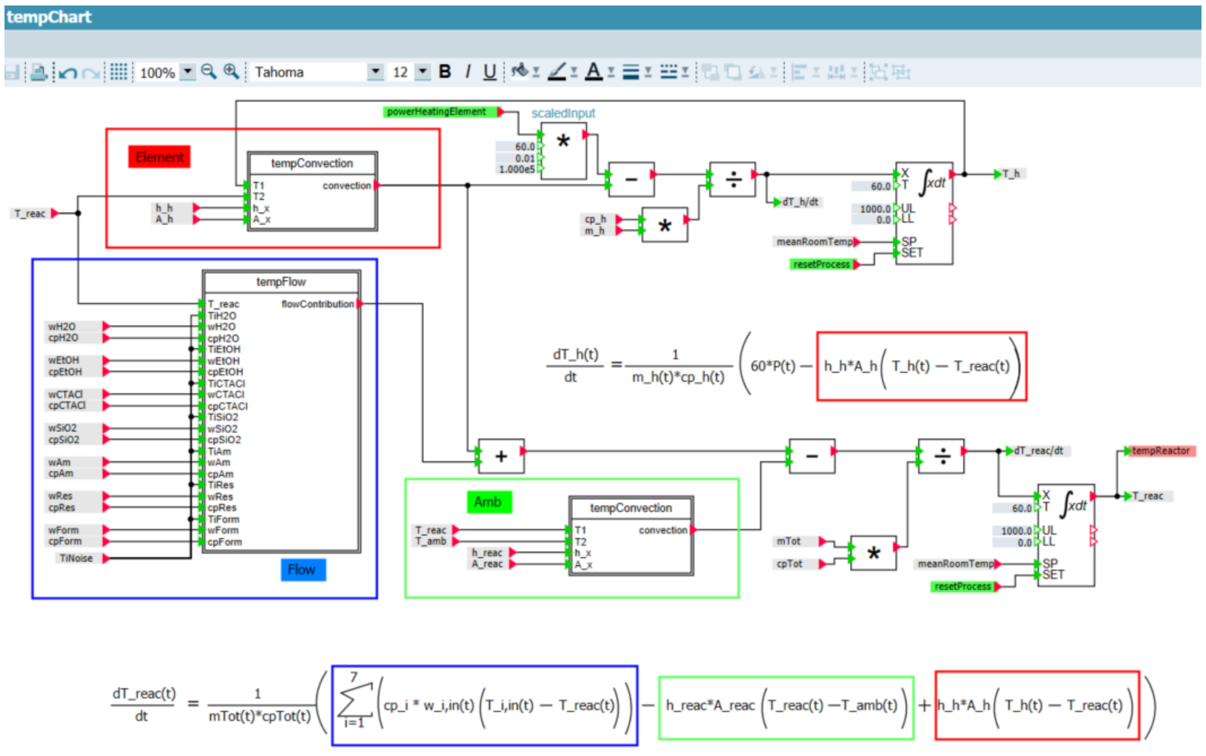


Figure E.0.6: Density chart



$$\frac{dT\_h(t)}{dt} = \frac{1}{m\_h(t)*cp\_h(t)}\left(60*P(t) - \boxed{h\_h*A\_h\left(T\_h(t) - T\_reac(t)\right)}\right)$$

$$\frac{dT\_reac(t)}{dt} = \frac{1}{mTot(t)*cpTot(t)}\left(\boxed{\sum_{i=1}^{7}\left(cp\_i * w\_i,in(t)\left(T\_i,in(t) - T\_reac(t)\right)\right)} - \boxed{h\_reac*A\_reac\left(T\_reac(t) - T\_amb(t)\right)} + \boxed{h\_h*A\_h\left(T\_h(t) - T\_reac(t)\right)}\right)$$

Figure E.0.7: Temperature chart

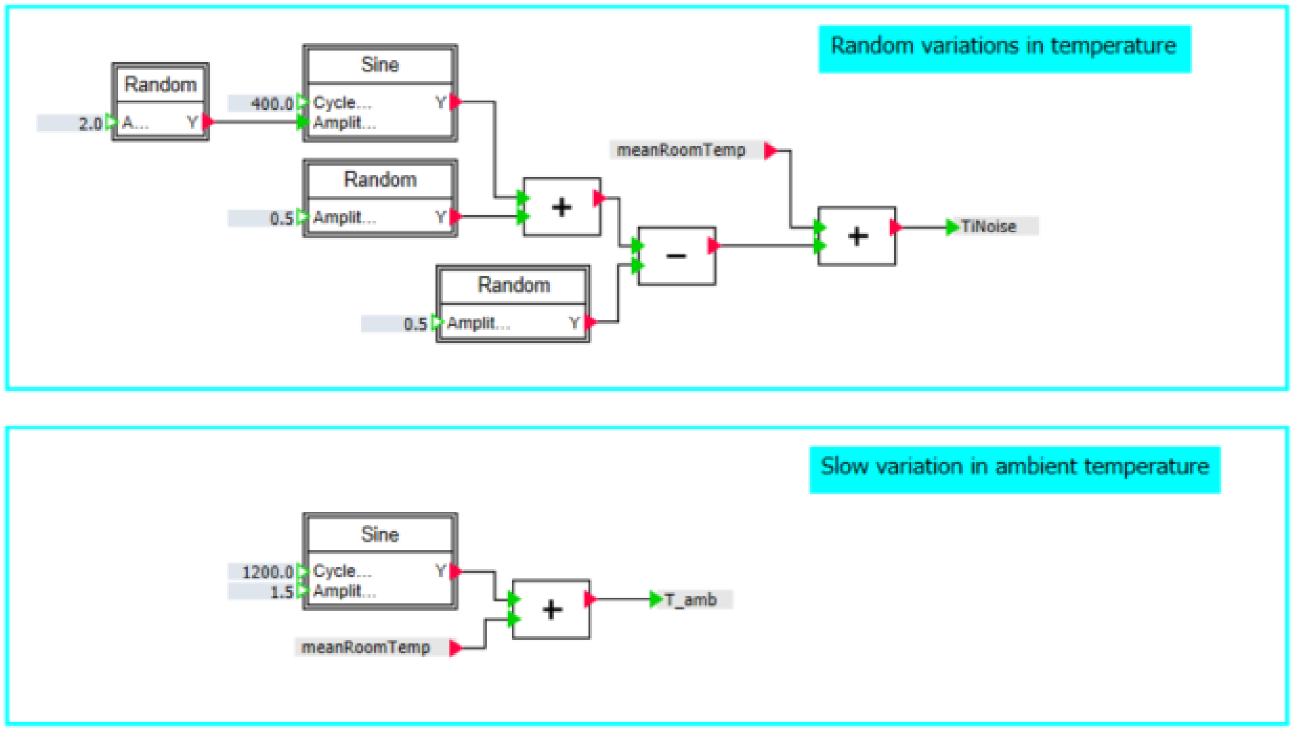
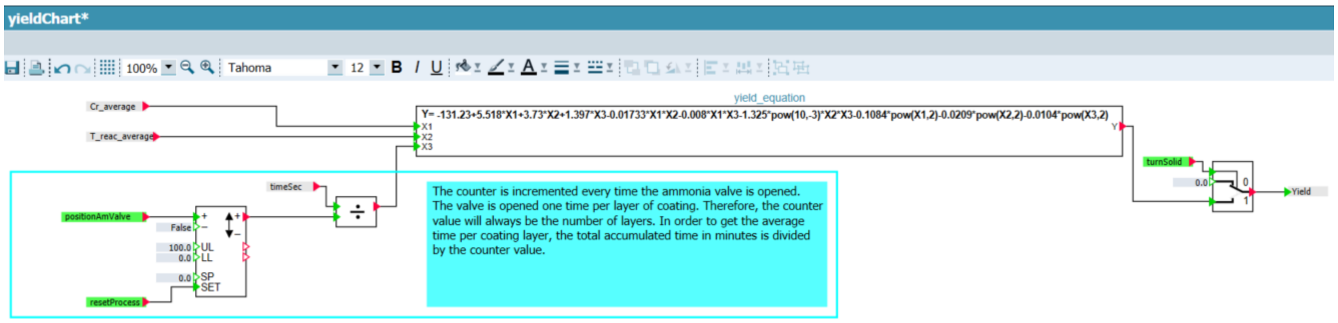


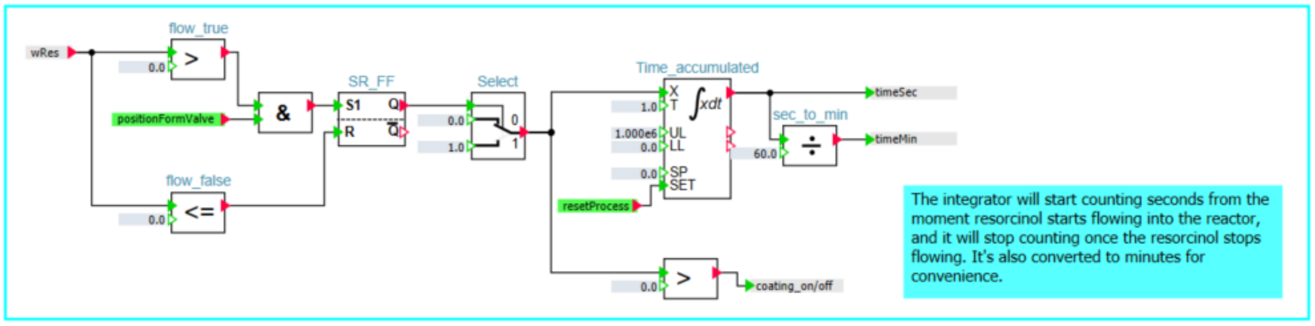
Figure E.0.8: Temperature noise chart



Y= -131.23+5.518*X1+3.73*X2+1.397*X3-0.01733*X1*X2-0.008*X1*X3-1.325*pow(10,-3)*X2*X3-0.1084*pow(X1,2)-0.0209*pow(X2,2)-0.0104*pow(X3,2)

The counter is incremented every time the ammonia valve is opened. The valve is opened one time per layer of coating. Therefore, the counter value will always be the number of layers. In order to get the average time per coating layer, the total accumulated time in minutes is divided by the counter value.

Figure E.0.9: Yield chart, part one



The integrator will start counting seconds from the moment resorcinol starts flowing into the reactor, and it will stop counting once the resorcinol stops flowing. It's also converted to minutes for convenience.

Figure E.0.10: Yield chart, part two

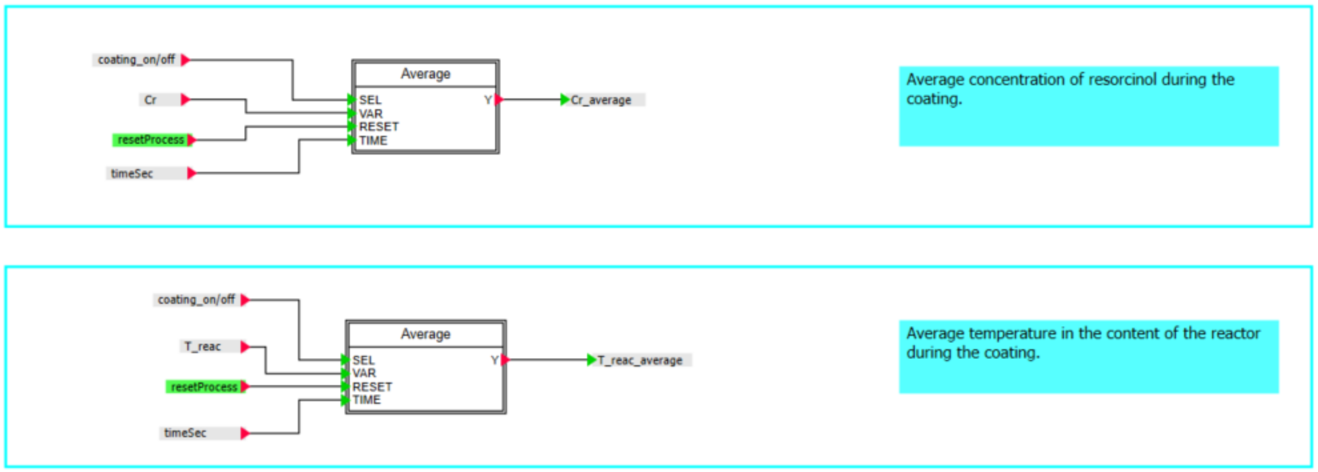


Figure E.0.11: Yield chart, part three

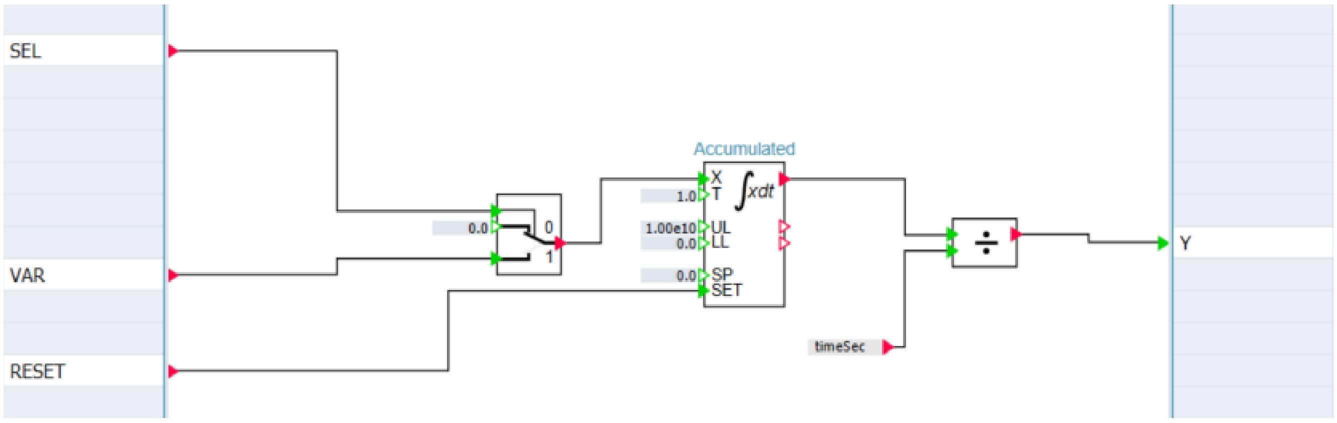


Figure E.0.12: Average calculation macro

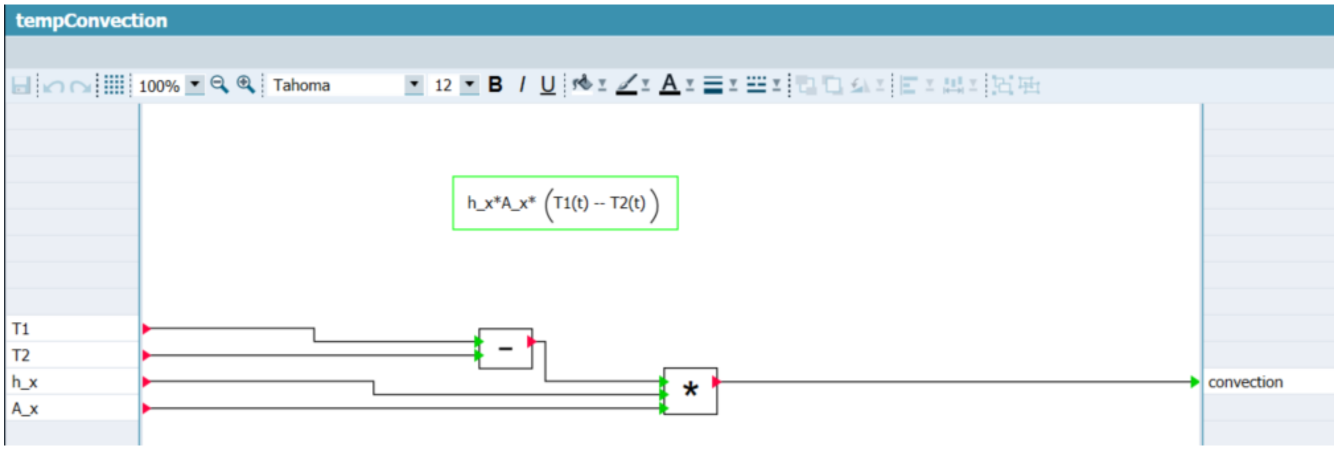


Figure E.0.13: Convection heat transfer macro

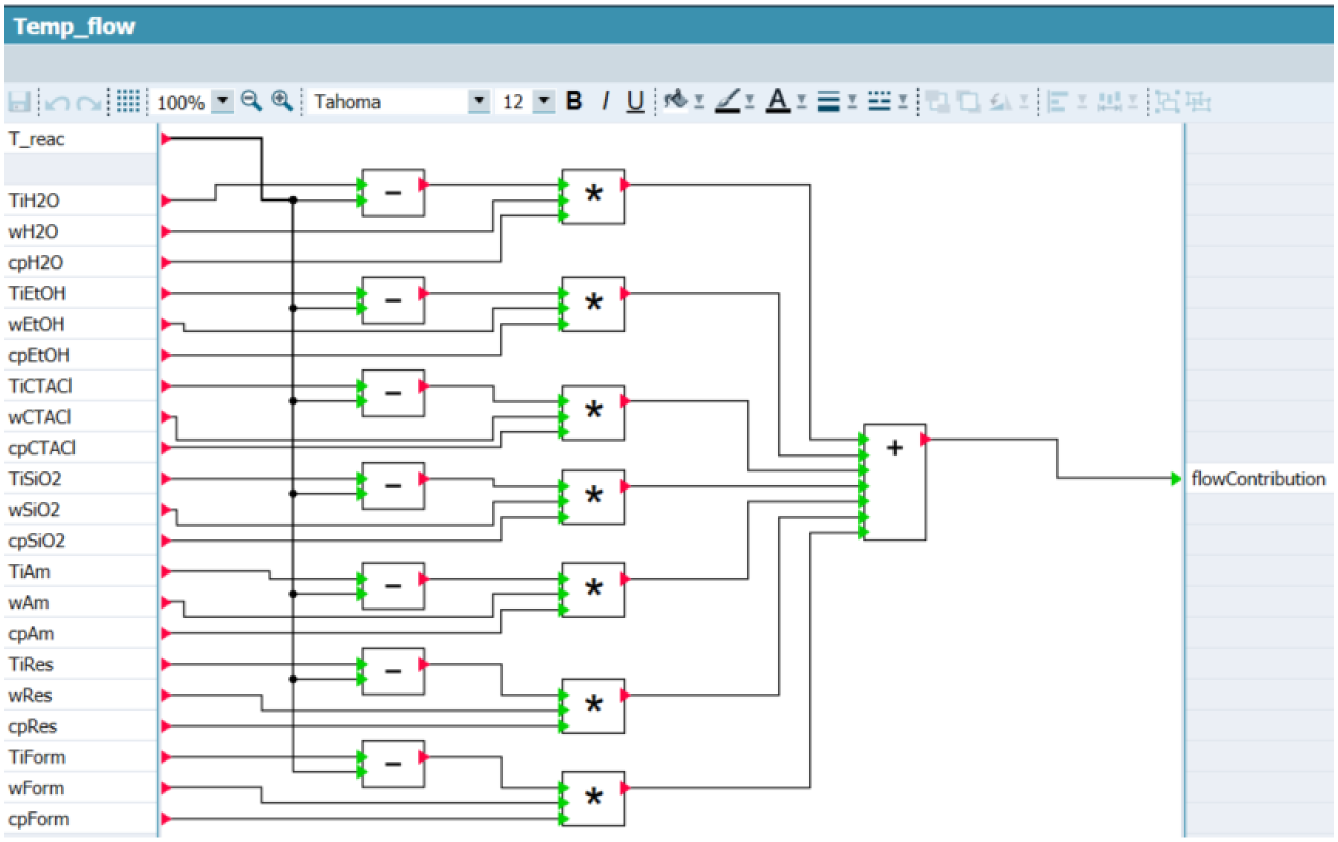


Figure E.0.14: Energy from total flow macro

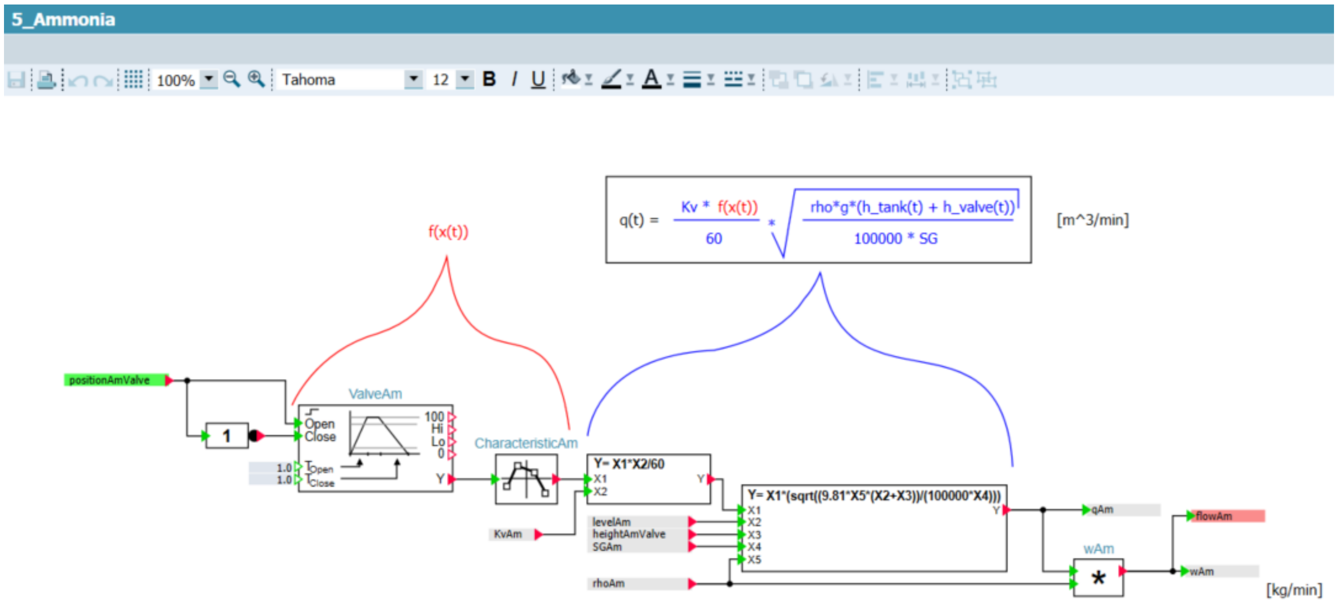


Figure E.0.15: Ammonia flow chart

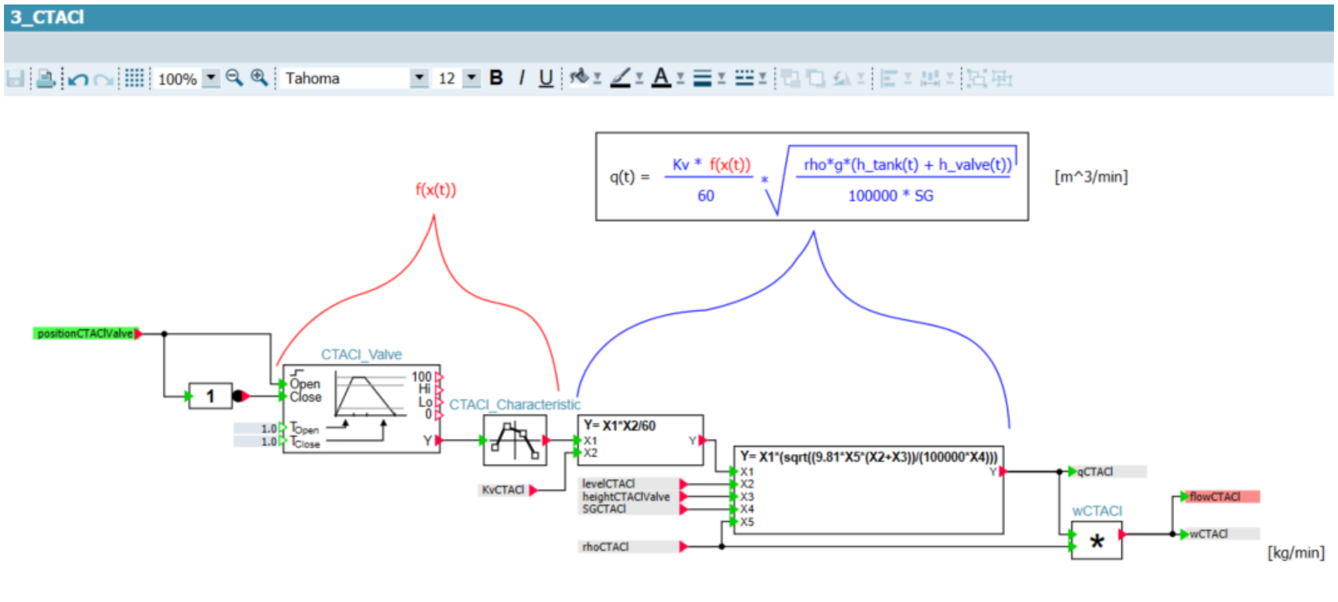


Figure E.0.16: CTACl flow chart

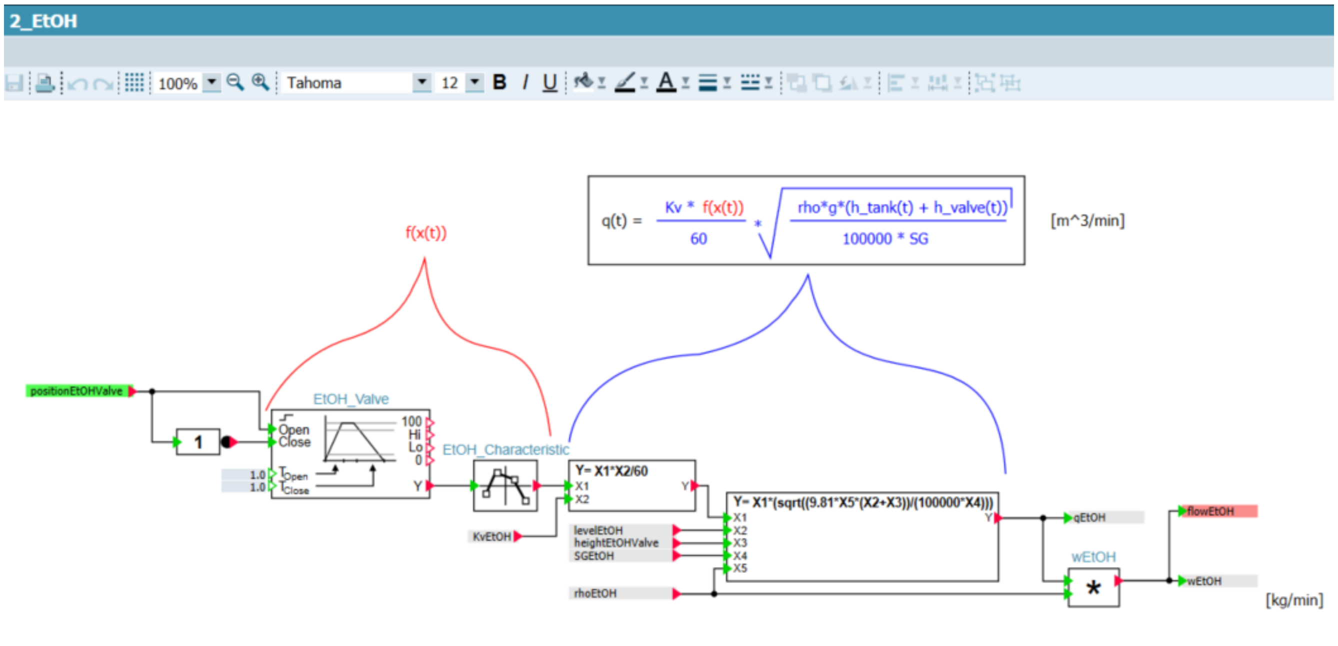


Figure E.0.17: EtOH flow chart

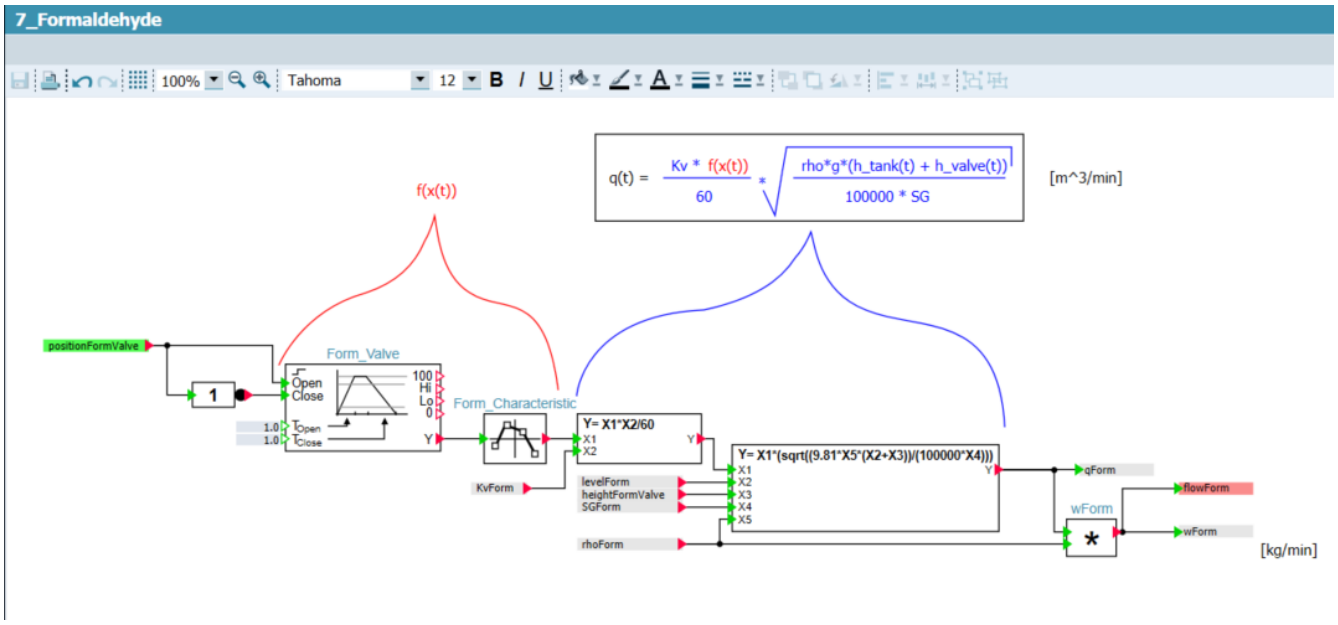


Figure E.0.18: Formaldehyde flow chart

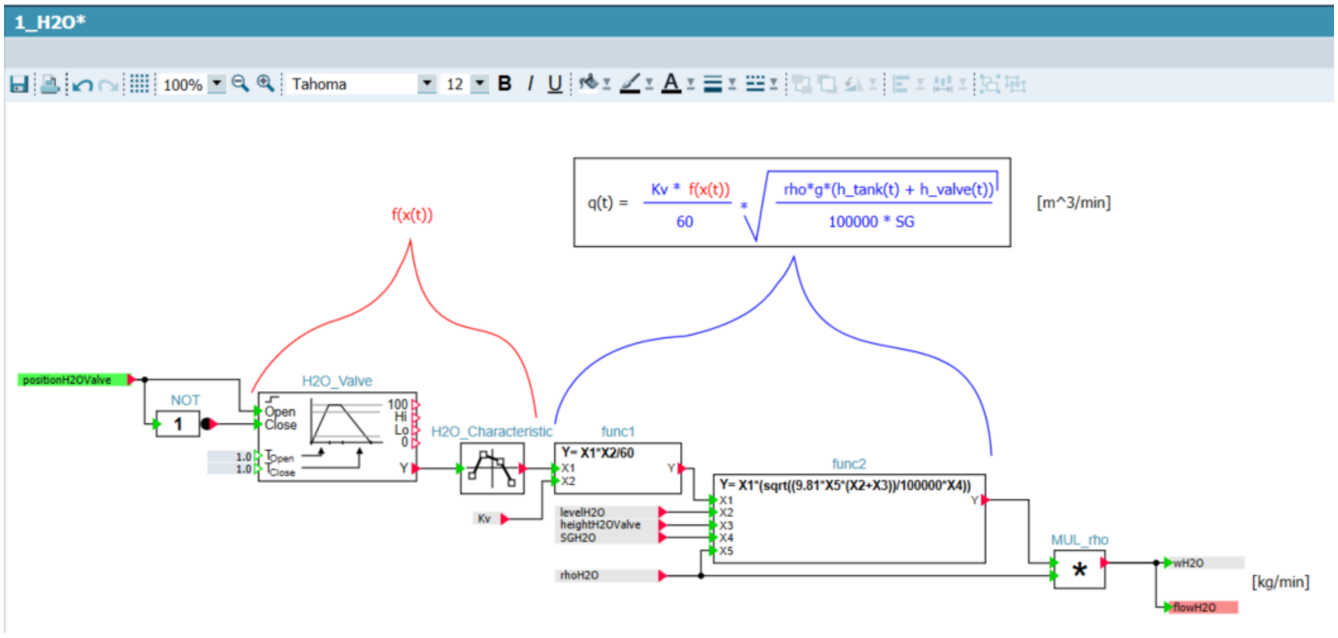


Figure E.0.19: H2O flow chart

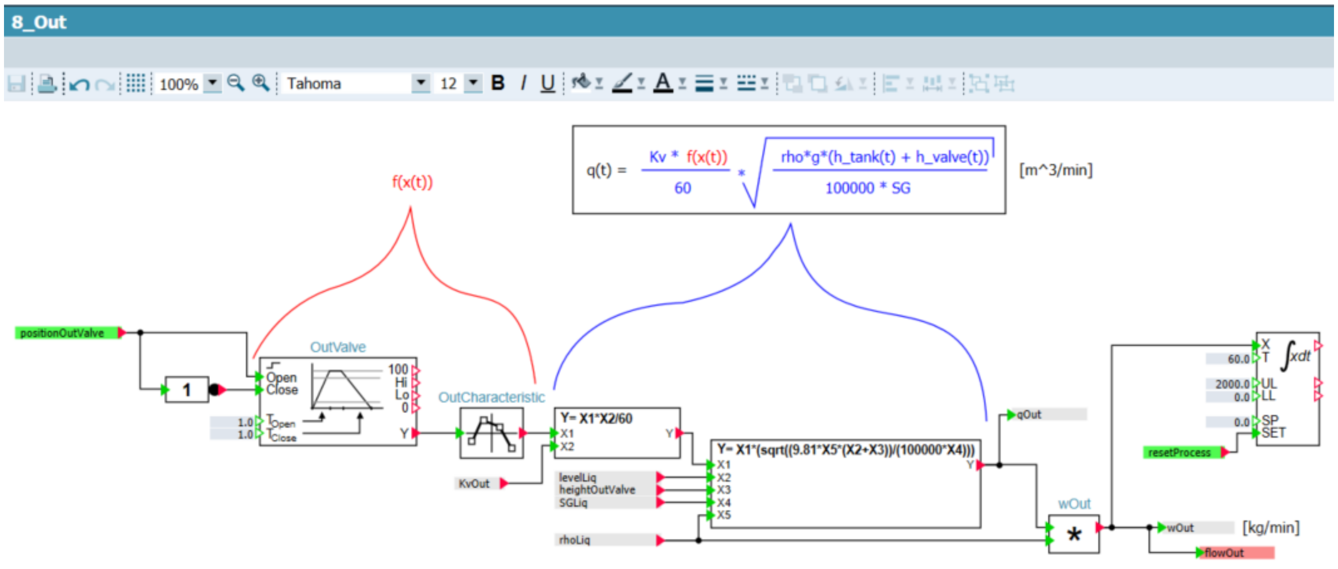


Figure E.0.20: Out flow chart

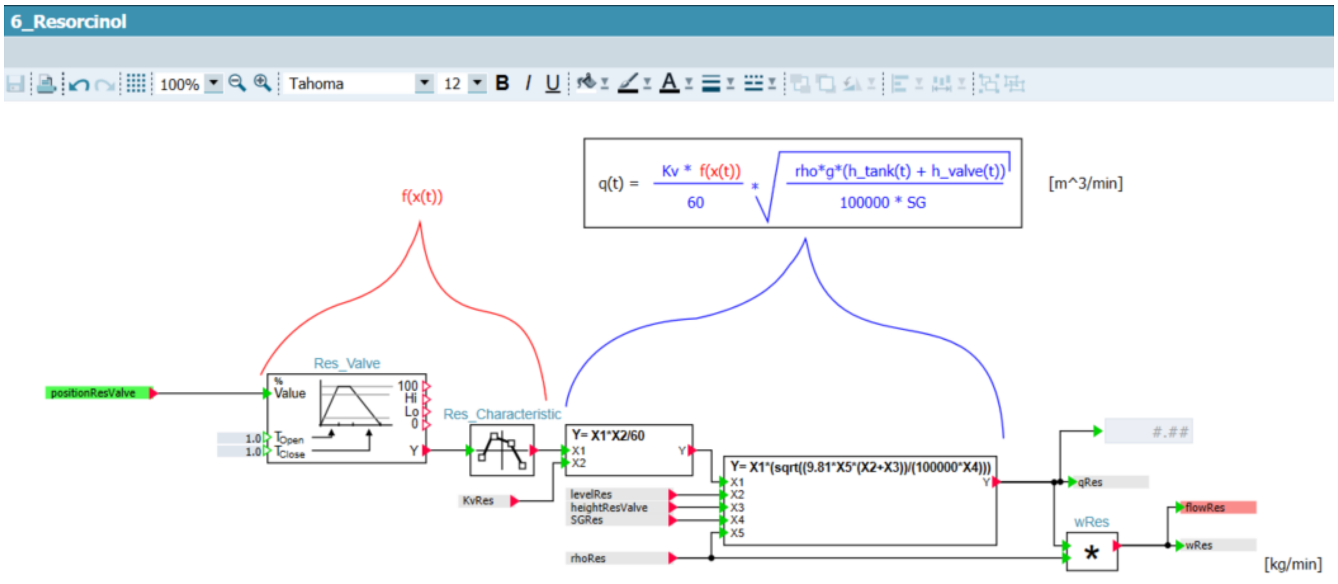


Figure E.0.21: Resorcinol flow chart

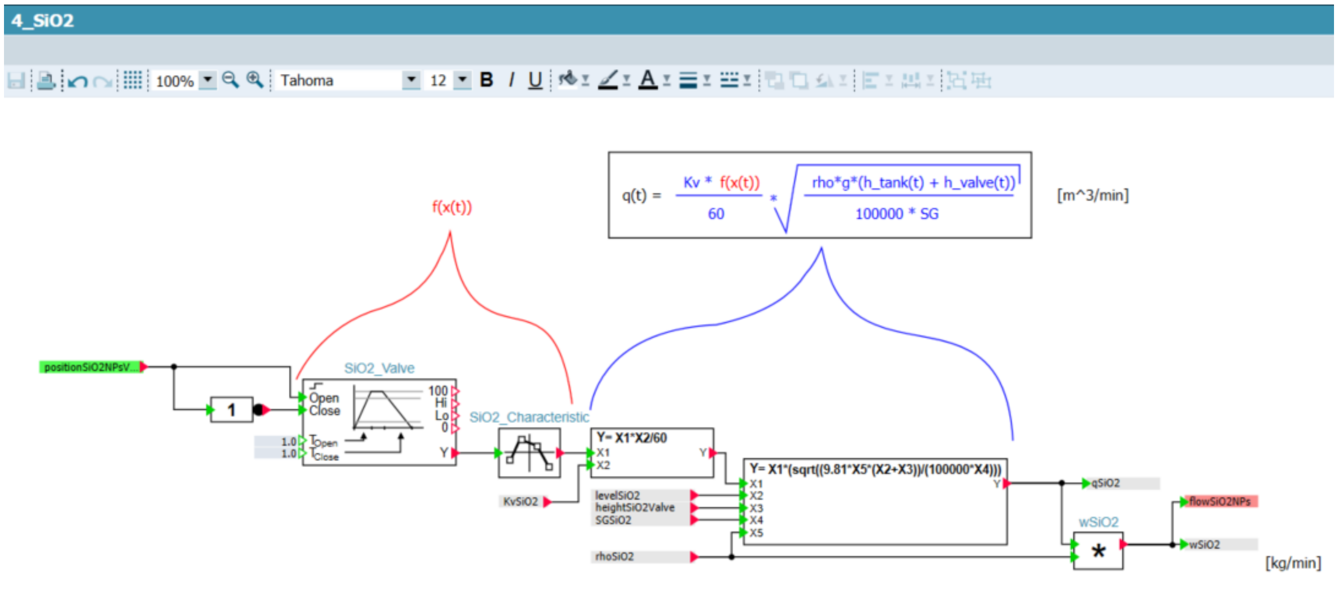


Figure E.0.22: SiO2 flow chart

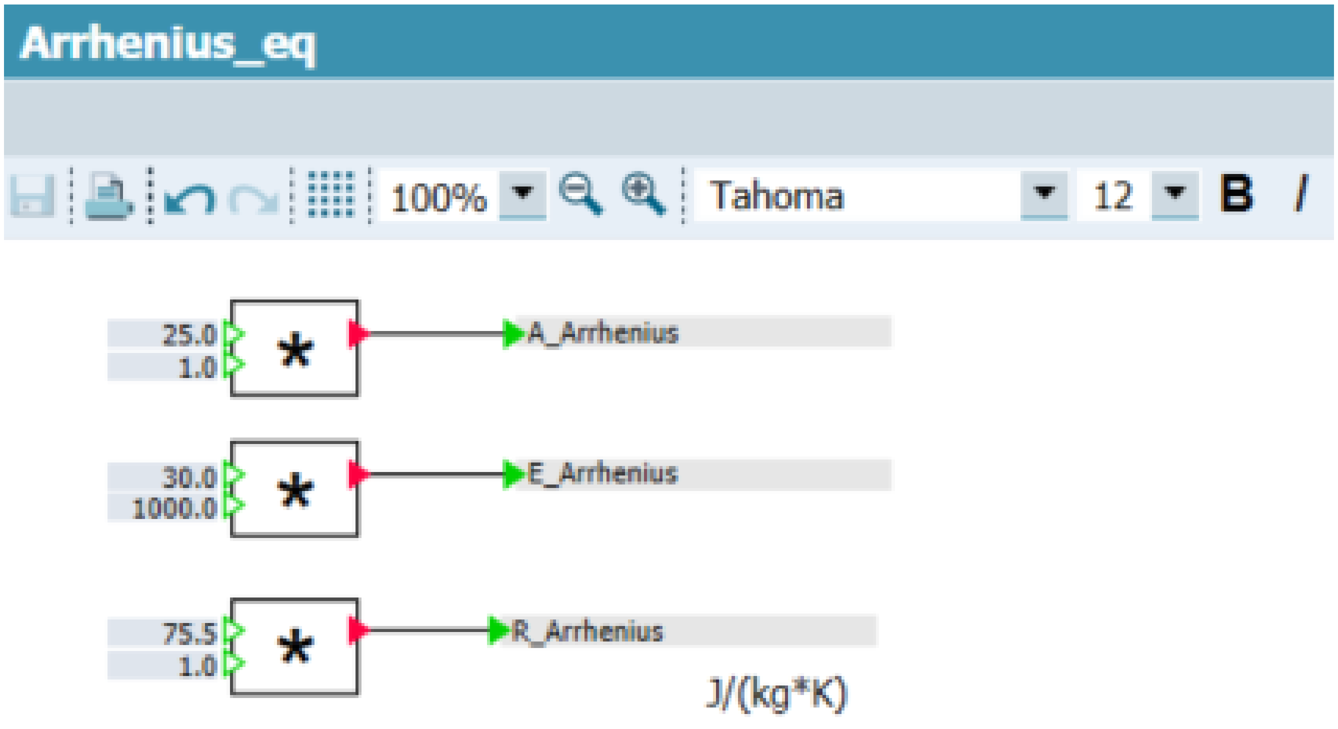


Figure E.0.23: Arrhenius variables chart

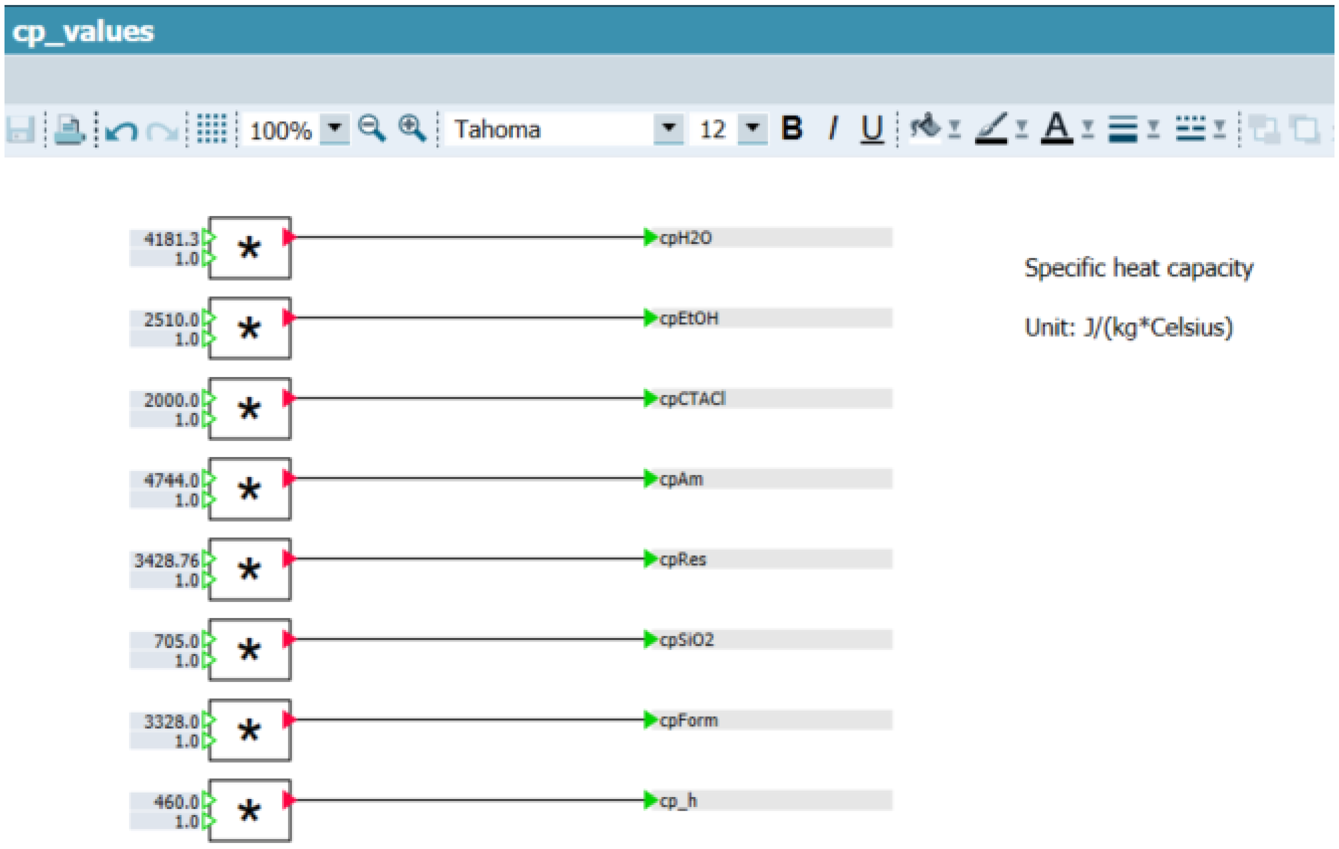


Figure E.0.24: C_p variables chart

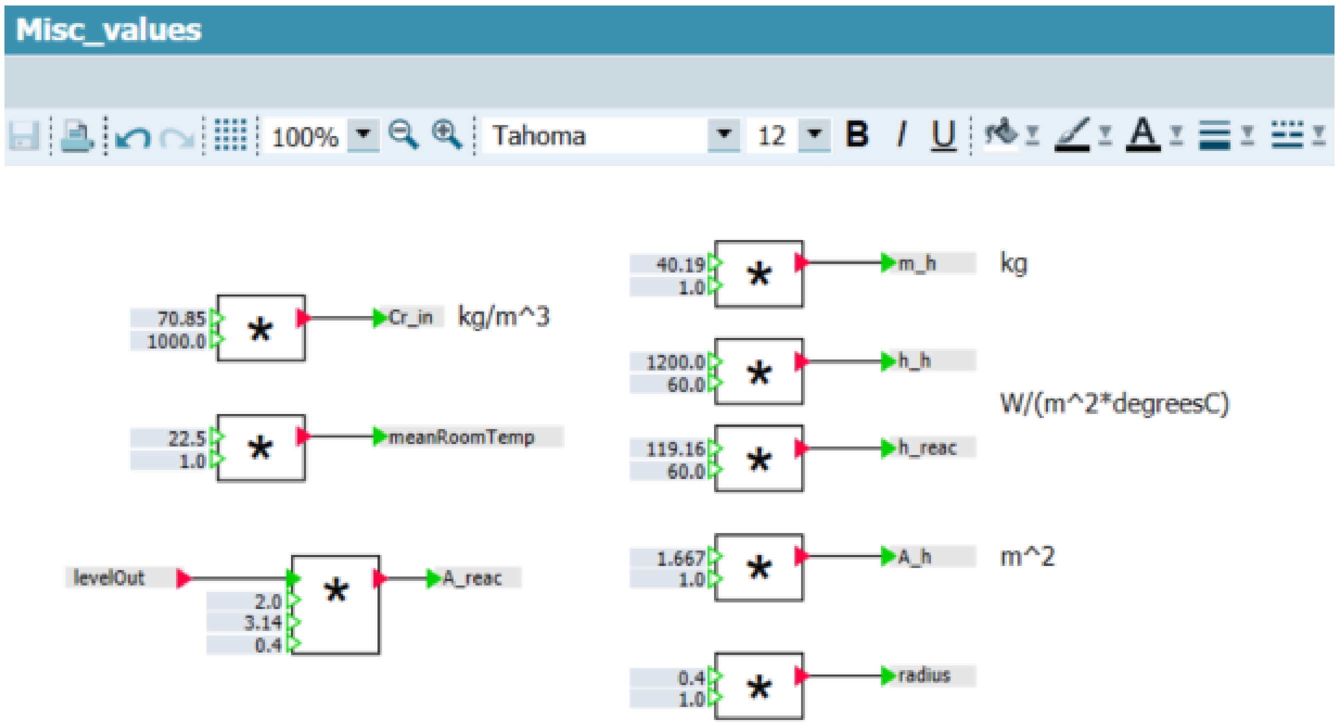


Figure E.0.25: miscellaneous variables chart

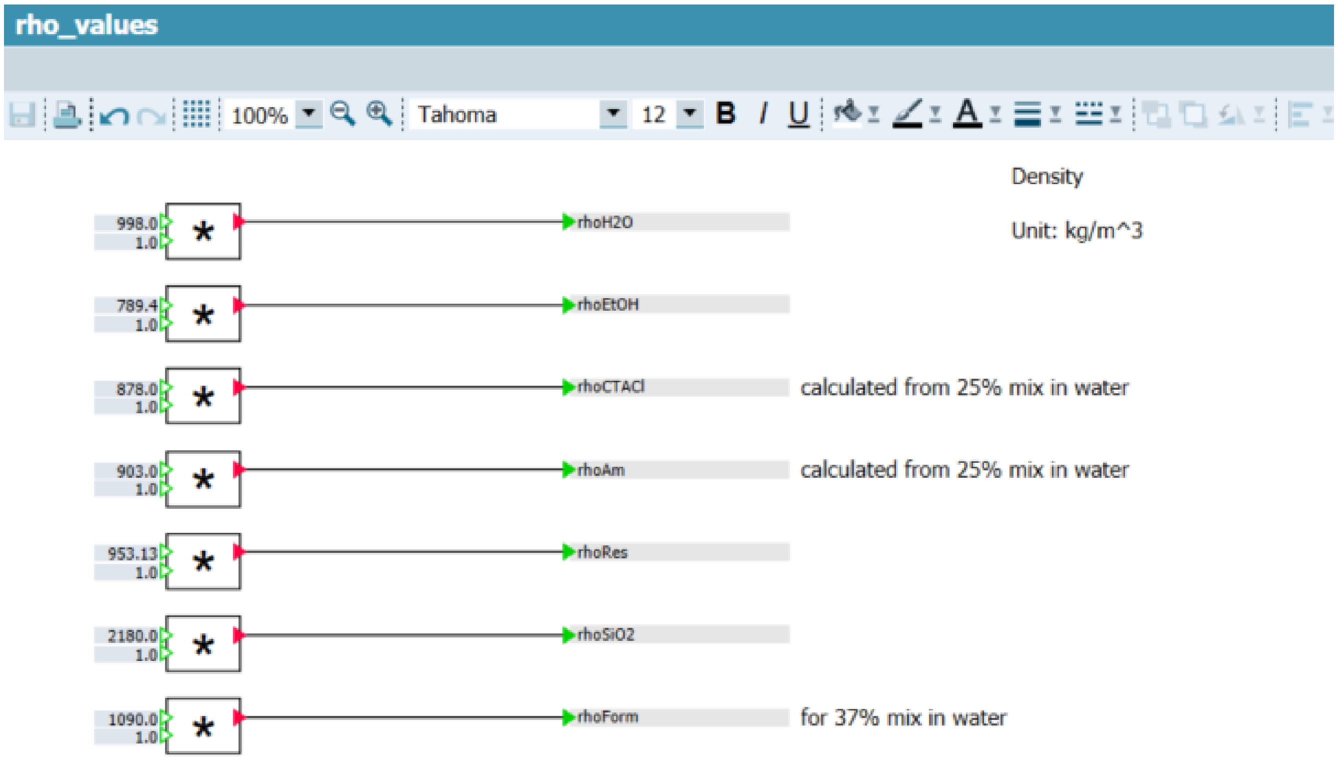


Figure E.0.26: Density variables chart

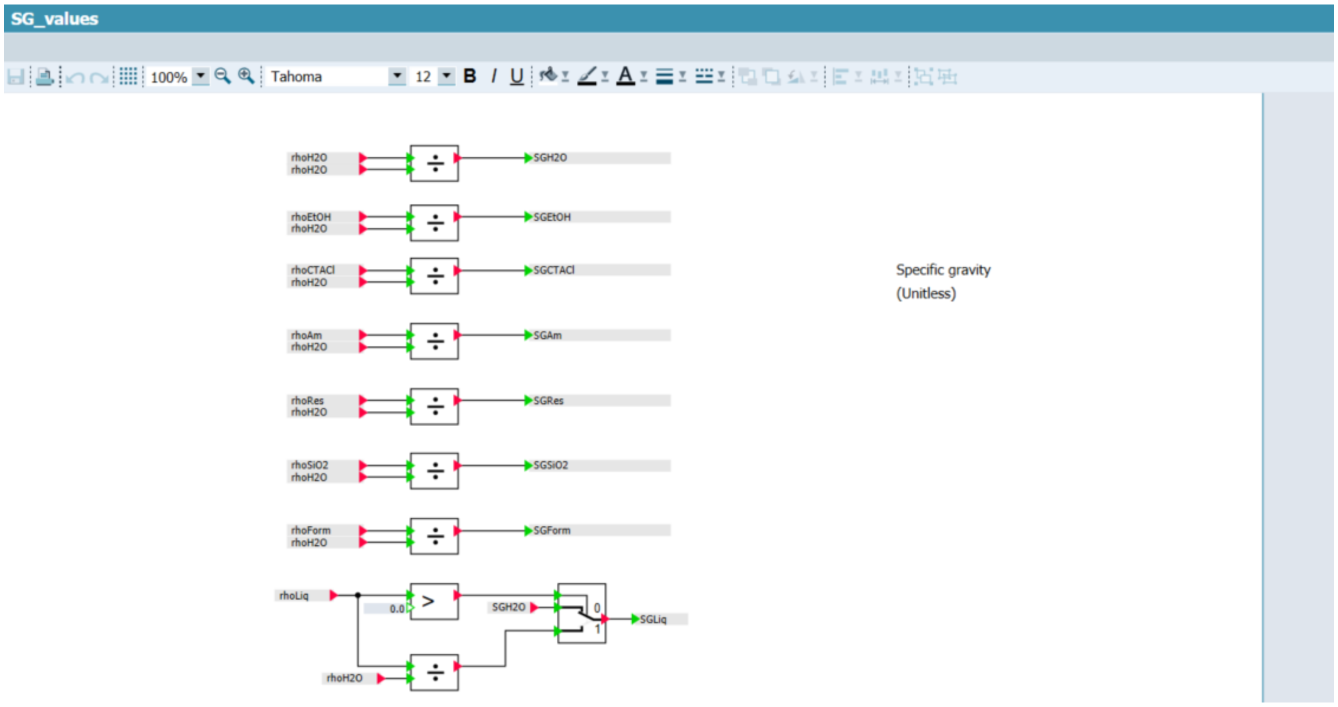


Figure E.0.27: SG variables chart

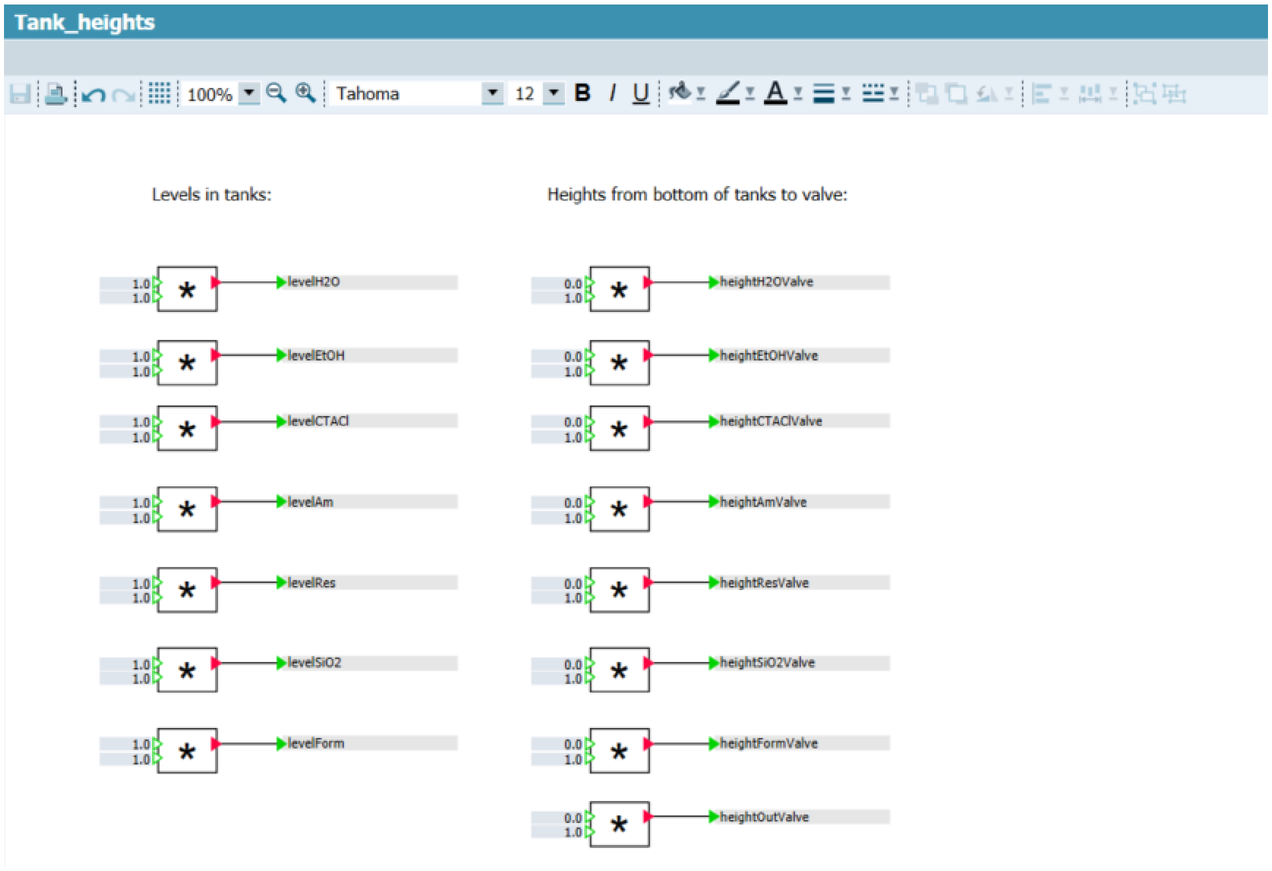


Figure E.0.28: Tank height variables chart

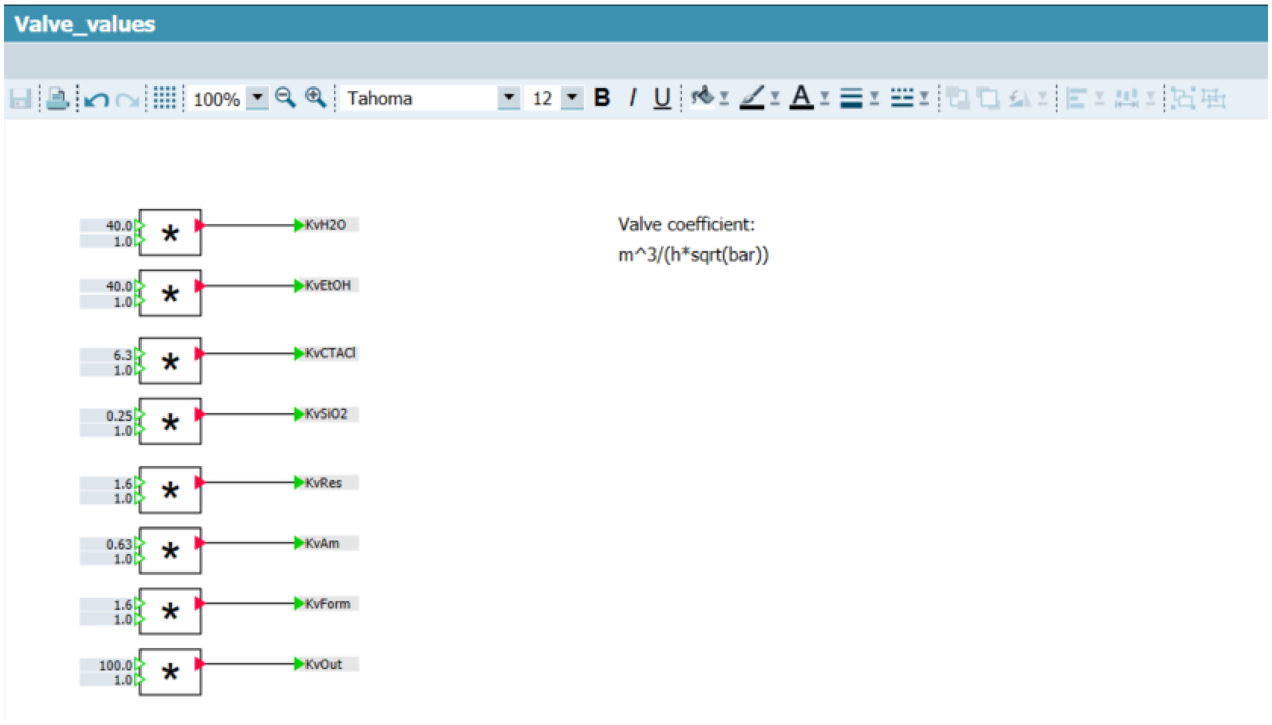
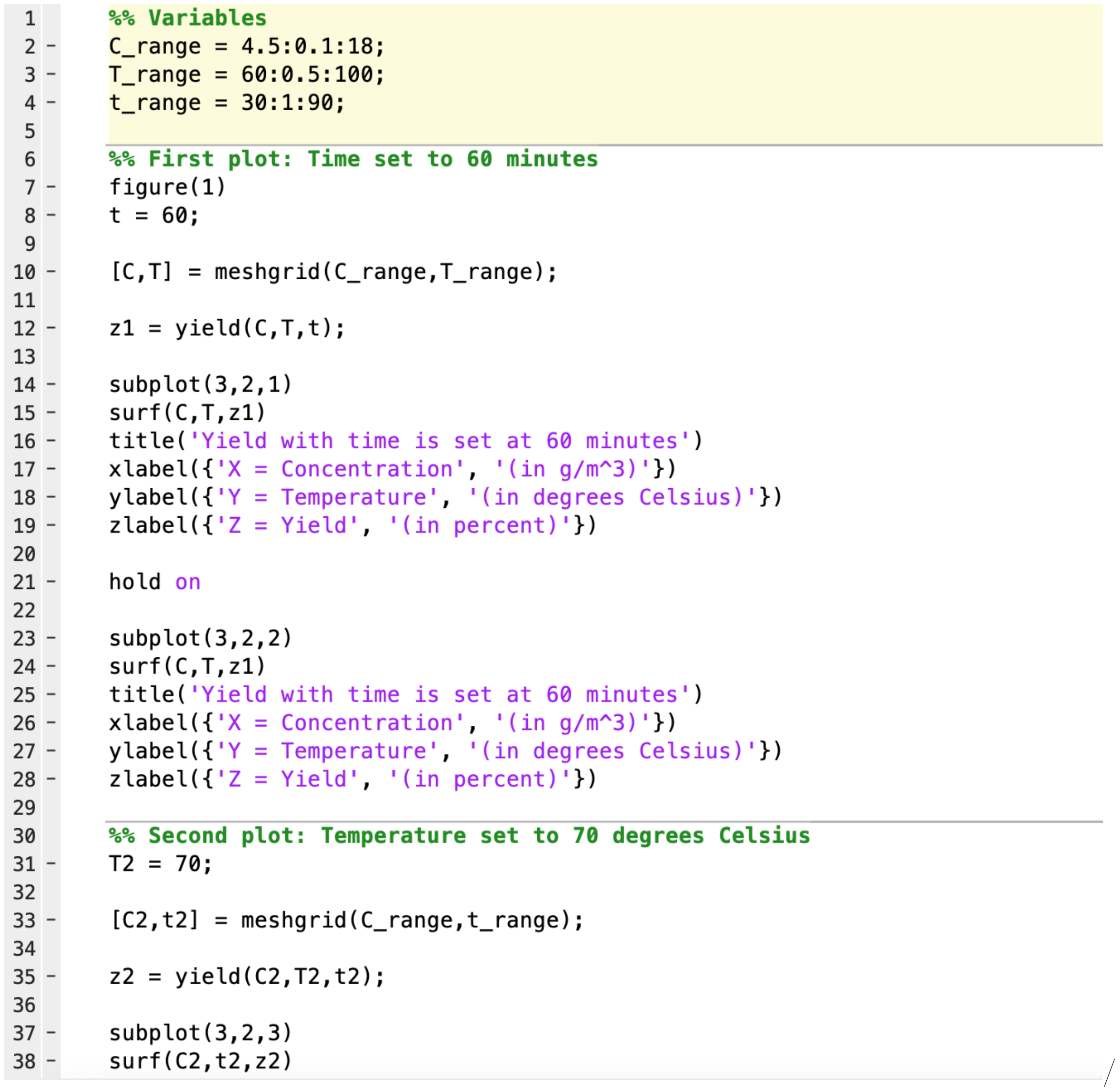


Figure E.0.29: Valve values chart

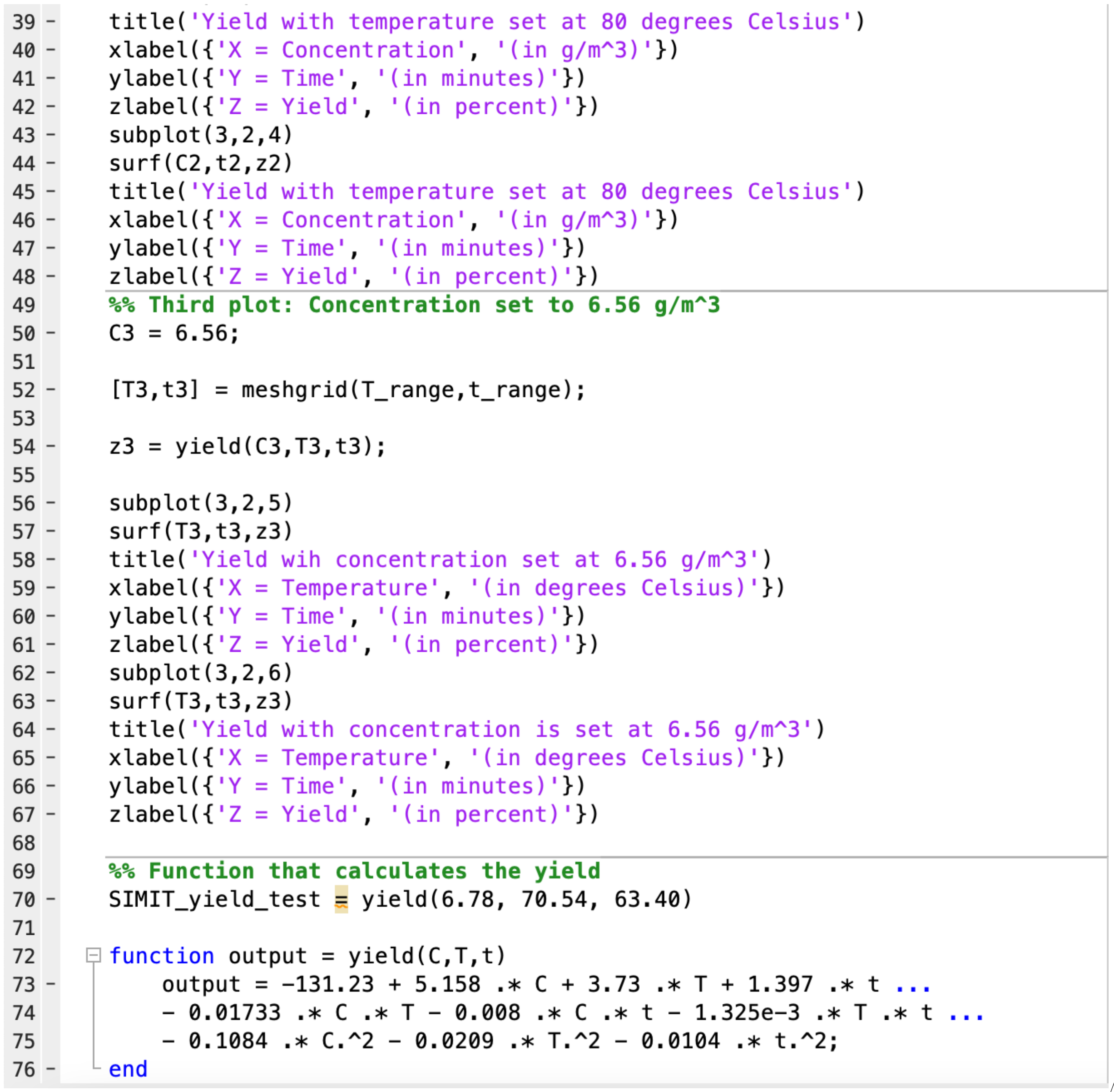
Appendix F

Matlab Scripts

This appendix shown the Matlab code used to convert the state-space model to transfer function and the Matlab script made to create the yield model. Firstly, script for the plot is showed, then the conversion script.



```
%% Variables
C_range = 4.5:0.1:18;
T_range = 60:0.5:100;
t_range = 30:1:90;

%% First plot: Time set to 60 minutes
figure(1)
t = 60;

[C,T] = meshgrid(C_range,T_range);

z1 = yield(C,T,t);

subplot(3,2,1)
surf(C,T,z1)
title('Yield with time is set at 60 minutes')
xlabel({'X = Concentration', '(in g/m^3)'})
ylabel({'Y = Temperature', '(in degrees Celsius)'})
zlabel({'Z = Yield', '(in percent)'})

hold on

subplot(3,2,2)
surf(C,T,z1)
title('Yield with time is set at 60 minutes')
xlabel({'X = Concentration', '(in g/m^3)'})
ylabel({'Y = Temperature', '(in degrees Celsius)'})
zlabel({'Z = Yield', '(in percent)'})

%% Second plot: Temperature set to 70 degrees Celsius
T2 = 70;

[C2,t2] = meshgrid(C_range,t_range);

z2 = yield(C2,T2,t2);

subplot(3,2,3)
surf(C2,t2,z2)
```



```matlab
title('Yield with temperature set at 80 degrees Celsius')
xlabel({'X = Concentration', '(in g/m^3)'})
ylabel({'Y = Time', '(in minutes)'})
zlabel({'Z = Yield', '(in percent)'})
subplot(3,2,4)
surf(C2,t2,z2)
title('Yield with temperature set at 80 degrees Celsius')
xlabel({'X = Concentration', '(in g/m^3)'})
ylabel({'Y = Time', '(in minutes)'})
zlabel({'Z = Yield', '(in percent)'})
%% Third plot: Concentration set to 6.56 g/m^3
C3 = 6.56;

[T3,t3] = meshgrid(T_range,t_range);

z3 = yield(C3,T3,t3);

subplot(3,2,5)
surf(T3,t3,z3)
title('Yield wih concentration set at 6.56 g/m^3')
xlabel({'X = Temperature', '(in degrees Celsius)'})
ylabel({'Y = Time', '(in minutes)'})
zlabel({'Z = Yield', '(in percent)'})
subplot(3,2,6)
surf(T3,t3,z3)
title('Yield with concentration is set at 6.56 g/m^3')
xlabel({'X = Temperature', '(in degrees Celsius)'})
ylabel({'Y = Time', '(in minutes)'})
zlabel({'Z = Yield', '(in percent)'})

%% Function that calculates the yield
SIMIT_yield_test = yield(6.78, 70.54, 63.40)

function output = yield(C,T,t)
    output = -131.23 + 5.158 .* C + 3.73 .* T + 1.397 .* t ...
    - 0.01733 .* C .* T - 0.008 .* C .* t - 1.325e-3 .* T .* t ...
    - 0.1084 .* C.^2 - 0.0209 .* T.^2 - 0.0104 .* t.^2;
end
```

```
% This Matlab script has the task to calculate the WPs of a set phase
 and
% use these WPs to transfer the state-space representation to transfer
% functions.


%Inputs: (Are changed during different phases of the process)
x_am = 1;
x_form = 0;
x_out = 0;

m_res = 0; %rho_res * q_res * time (min)
m_am = 0;
m_form = 0;


%Constants:
rho_EtOH = 789.4;
rho_CTACl = 878;
rho_SiO2 = 2180;
rho_res = 953.13;
rho_am = 903;
rho_form = 1090;
rho_H2O = 998;

m_prep = ((rho_H2O * 0.32) + (rho_EtOH * 0.25) + (30) + (2));
m_tot = (m_res + m_am + m_form + m_prep);


g = 9.81;

h_res = 1;
h_am = 1;
h_form = 1;

rho_prep = ((rho_H2O^2 * 0.32) + (rho_EtOH^2 * 0.25) + (rho_CTACl *
 30) + (rho_SiO2 * 2))/m_prep;
rho_tot = (rho_res * m_res + rho_am * m_am + rho_form * m_form +
 rho_prep * m_prep)/m_tot;


Cp_EtOH = 2510;
Cp_CTACl = 2000; %?
Cp_SiO2 = 705;
Cp_res = 3428;
Cp_am = 4744;
Cp_form = 3328;
Cp_H2O = 4181.3;
Cp_prep = ((Cp_H2O * rho_H2O * 0.32) + (Cp_EtOH * rho_EtOH * 0.25) +
 (Cp_CTACl * 30) + (Cp_SiO2 * 2))/m_tot;
```

```
SG_res = rho_res/rho_H2O;
SG_am = rho_am/rho_H2O;
SG_form = rho_form/rho_H2O;
SG_H2O = 1;
SG_out = rho_tot/rho_H2O;

Tin_res = 22.5;
Tin_am = 22.5;
Tin_form = 22.5;
T_Ambient = 22.5;

h_reac = 7149.6;

Kv_res = 1.6;
Kv_am = 0.63;
Kv_form = 1.6;
Kv_out = 100;

h_h = 72000;
A_h = 1.667;
m_h = 40.19;
Cp_h = 460;

a = 25;
Ea = 30000;
R = 75.5;

Cres_in = 70850;

r = 0.4;


%Simplification of equations.

V_tot = (m_res/rho_res) + (m_am/rho_am) + (m_form/rho_form)+ (m_prep/
rho_prep);

Cp_tot = (Cp_res * m_res + Cp_am * m_am + Cp_form * m_form + Cp_prep *
 m_prep)/m_tot;

A_reac = (5 * m_tot)/rho_tot;

h_out = m_tot/(rho_tot*pi*0.4^2);

q_am = ((Kv_am*x_am)/60) * sqrt((rho_am * g * h_am)/(100000 * SG_am));
q_form = ((Kv_form*x_form)/60) * sqrt((rho_form * g * h_form)/(100000
 * SG_form));

%Calculations of WPs based on desired output and constant values:

%Desired reactor temperature and concentration.
C_res = 6.56;
T_reac = 70;
```

```
%Flow and valve position based of desired temp and conc
q_res = C_res * ((q_am + q_form) + (V_tot * a * exp((-Ea)/(R*(T_reac
+273.15)))))/(Cres_in-C_res);
x_res = (q_res*60/Kv_res) * sqrt((100000 * SG_res)/(rho_res*g*h_res));

%Heating element temp and power on the element at desired temp.
T_h = ((h_reac * A_reac *(T_reac - T_Ambient))-(q_res*Cp_res * rho_res
 * (Tin_res-T_reac) + q_am * Cp_am * rho_am * ...
    (Tin_am-T_reac) + q_form * Cp_form * rho_form * (Tin_form-
T_reac)))/(h_h*A_h)+ T_reac;
P = (h_h * A_h *(T_h - T_reac))/60;

%State space coefficiants

%Reactor Temp
A1_1 = -(1/(m_tot*Cp_tot)) * ((h_h*A_h) + ((Cp_res * rho_res *
 q_res) + (Cp_am * rho_am * q_am) + (Cp_form * rho_form * q_form)) +
 (h_reac*A_reac));
A1_2 = 0;
A1_3 = (h_h*A_h/(m_tot*Cp_tot));
A1_4 = 1/((rho_tot)^2*(Cp_tot)^2*(m_tot)^4)*(m_res^2*(-
Cp_res*rho_res^2*h_h*A_h*(T_h-T_reac)-
 Cp_res*rho_res^2*((rho_res*Cp_res*(Tin_res-T_reac)*q_res) ...
        + (rho_am*Cp_am*(Tin_am-T_reac)*q_am) +
 (rho_form*Cp_form*(Tin_form-T_reac)*q_form)) ...
        + 5 * h_reac * (T_reac - T_Ambient)*(2*Cp_res*rho_res*(m_am
+m_form+m_prep)-Cp_res*(m_am*rho_am+m_form*rho_form+m_prep*rho_prep)-
rho_res*(m_am*Cp_am+m_form*Cp_form+m_prep*Cp_prep))) ...
        - m_res*(2*(m_am*rho_am+m_form*rho_form
+m_prep*rho_prep)*(Cp_res*rho_res*h_h*A_h*(T_h - T_reac) + Cp_res *
 rho_res *(rho_res*Cp_res*(Tin_res-T_reac)*q_res+rho_am*Cp_am*(Tin_am-
T_reac)*q_am+rho_form*Cp_form*(Tin_form-T_reac)*q_form))...
        + 5 * h_reac * (T_reac - T_Ambient) * (m_am*Cp_am
+m_form*Cp_form+m_prep*Cp_prep))...
        - (m_am*rho_am+m_form*rho_form+m_prep*rho_prep) *
 (Cp_res*h_h*A_h*(T_h-T_reac)*(m_am*rho_am+m_form*rho_form
+m_prep*rho_prep) + Cp_res*(Cp_res*rho_res*(Tin_res-T_reac)*q_res
+Cp_am*rho_am*(Tin_am-T_reac)*q_am+Cp_form*rho_form*(Tin_form-
T_reac)*q_form)...
        * (m_am*rho_am+m_form*rho_form+m_prep*rho_prep) + 10 *
 h_reac * (T_reac-T_Ambient) * (m_am+m_form+m_prep) * (m_am*Cp_am
+m_form*Cp_form+m_prep*Cp_prep)));


A1_5 = 1/((rho_tot)^2*(Cp_tot)^2*(m_tot)^4)*(m_am^2*(-
Cp_am*rho_am^2*h_h*A_h*(T_h-T_reac)-
 Cp_am*rho_am^2*((rho_res*Cp_res*(Tin_res-T_reac)*q_res) ...
        + (rho_am*Cp_am*(Tin_am-T_reac)*q_am) +
 (rho_form*Cp_form*(Tin_form-T_reac)*q_form)) ...
        + 5 * h_reac * (T_reac - T_Ambient)*(2*Cp_am*rho_am*(m_res
+m_form+m_prep)-Cp_am*(m_res*rho_res+m_form*rho_form+m_prep*rho_prep)-
rho_am*(m_res*Cp_res+m_form*Cp_form+m_prep*Cp_prep))) ...
```

```
       - m_am*(2*(m_res*rho_res+m_form*rho_form
+m_prep*rho_prep)*(Cp_am*rho_am*h_h*A_h*(T_h - T_reac) + Cp_am *
 rho_am *(rho_res*Cp_res*(Tin_res-T_reac)*q_res+rho_am*Cp_am*(Tin_am-
T_reac)*q_am+rho_form*Cp_form*(Tin_form-T_reac)*q_form))...
       + 5 * h_reac * (T_reac - T_Ambient) * (m_res*Cp_res
+m_form*Cp_form+m_prep*Cp_prep))...
       - (m_res*rho_res+m_form*rho_form+m_prep*rho_prep) *
 (Cp_am*h_h*A_h*(T_h-T_reac)*(m_res*rho_res+m_form*rho_form
+m_prep*rho_prep) + Cp_am*(Cp_res*rho_res*(Tin_res-T_reac)*q_res
+Cp_am*rho_am*(Tin_am-T_reac)*q_am+Cp_form*rho_form*(Tin_form-
T_reac)*q_form)...
       * (m_res*rho_res+m_form*rho_form+m_prep*rho_prep) + 10 *
 h_reac * (T_reac-T_Ambient) * (m_res+m_form+m_prep) * (m_res*Cp_res
+m_form*Cp_form+m_prep*Cp_prep)));

A1_6 = 1/((rho_tot)^2*(Cp_tot)^2*(m_tot)^4)*(m_form^2*(-
Cp_form*rho_form^2*h_h*A_h*(T_h-T_reac)-
 Cp_form*rho_form^2*((rho_res*Cp_res*(Tin_res-T_reac)*q_res) ...
       + (rho_am*Cp_am*(Tin_am-T_reac)*q_am) +
 (rho_form*Cp_form*(Tin_form-T_reac)*q_form)) ...
       + 5 * h_reac * (T_reac - T_Ambient)*(2*Cp_form*rho_form*(m_res
+m_am+m_prep)-Cp_form*(m_res*rho_res+m_am*rho_am+m_prep*rho_prep)-
rho_form*(m_res*Cp_res+m_am*Cp_am+m_prep*Cp_prep))) ...
       - m_form*(2*(m_res*rho_res+m_am*rho_am
+m_prep*rho_prep)*(Cp_form*rho_form*h_h*A_h*(T_h - T_reac) +
 Cp_form * rho_form *(rho_res*Cp_res*(Tin_res-T_reac)*q_res
+rho_am*Cp_am*(Tin_am-T_reac)*q_am+rho_form*Cp_form*(Tin_form-
T_reac)*q_form))...
       + 5 * h_reac * (T_reac - T_Ambient) * (m_res*Cp_res+m_am*Cp_am
+m_prep*Cp_prep))...
       - (m_res*rho_res+m_am*rho_am+m_prep*rho_prep) *
 (Cp_form*h_h*A_h*(T_h-T_reac)*(m_res*rho_res+m_am*rho_am
+m_prep*rho_prep) + Cp_form*(Cp_res*rho_res*(Tin_res-T_reac)*q_res
+Cp_am*rho_am*(Tin_am-T_reac)*q_am+Cp_form*rho_form*(Tin_form-
T_reac)*q_form)...
       * (m_res*rho_res+m_am*rho_am+m_prep*rho_prep) + 10 * h_reac *
 (T_reac-T_Ambient) * (m_res+m_am+m_prep) * (m_res*Cp_res+m_am*Cp_am
+m_prep*Cp_prep)));

A1_7 = 1/((rho_tot)^2*(Cp_tot)^2*(m_tot)^4)*(m_prep^2*(-
Cp_prep*rho_prep^2*h_h*A_h*(T_h-T_reac)-
 Cp_prep*rho_prep^2*((rho_res*Cp_res*(Tin_res-T_reac)*q_res) ...
       + (rho_am*Cp_am*(Tin_am-T_reac)*q_am) +
 (rho_form*Cp_form*(Tin_form-T_reac)*q_form)) ...
       + 5 * h_reac * (T_reac - T_Ambient)*(2*Cp_prep*rho_prep*(m_res
+m_am+m_form)-Cp_prep*(m_res*rho_res+m_am*rho_am+m_form*rho_form)-
rho_prep*(m_res*Cp_res+m_am*Cp_am+m_form*Cp_form))) ...
       - m_prep*(2*(m_res*rho_res+m_am*rho_am
+m_form*rho_form)*(Cp_prep*rho_prep*h_h*A_h*(T_h - T_reac) +
 Cp_prep * rho_prep *(rho_res*Cp_res*(Tin_res-T_reac)*q_res
+rho_am*Cp_am*(Tin_am-T_reac)*q_am+rho_form*Cp_form*(Tin_form-
T_reac)*q_form))...
       + 5 * h_reac * (T_reac - T_Ambient) * (m_res*Cp_res+m_am*Cp_am
+m_form*Cp_form))...
```

```
       - (m_res*rho_res+m_am*rho_am+m_form*rho_form) *
 (Cp_prep*h_h*A_h*(T_h-T_reac)*(m_res*rho_res+m_am*rho_am
+m_form*rho_form) + Cp_prep*(Cp_res*rho_res*(Tin_res-T_reac)*q_res
+Cp_am*rho_am*(Tin_am-T_reac)*q_am+Cp_form*rho_form*(Tin_form-
T_reac)*q_form)...
       * (m_res*rho_res+m_am*rho_am+m_form*rho_form) + 10 * h_reac *
 (T_reac-T_Ambient) * (m_res+m_am+m_form) * (m_res*Cp_res+m_am*Cp_am
+m_form*Cp_form)));

B1_1 = 0;
B1_2 = 0;
B1_3 = (rho_res * Cp_res * Kv_res * (Tin_res - T_reac))/(60 * m_tot *
 Cp_tot) * sqrt((rho_res * g * h_res)/(100000*SG_res));
B1_4 = (rho_am * Cp_am * Kv_am * (Tin_am - T_reac))/(60 * m_tot *
 Cp_tot) * sqrt((rho_am * g * h_am)/(100000*SG_am));
B1_5 = (rho_form * Cp_form * Kv_form * (Tin_form - T_reac))/(60 *
 m_tot * Cp_tot) * sqrt((rho_form * g * h_form)/(100000*SG_form));

Bd1_1 = (rho_res * Cp_res * Kv_res * x_res)/(60 * m_tot * Cp_tot) *
 sqrt((rho_res * g * h_res)/(100000*SG_res));
Bd1_2 = (rho_am * Cp_res * Kv_am * x_am)/(60 * m_tot * Cp_tot) *
 sqrt((rho_am * g * h_am)/(100000*SG_am));
Bd1_3 = (rho_form * Cp_form * Kv_form * x_form)/(60 * m_tot * Cp_tot)
 * sqrt((rho_form * g * h_form)/(100000*SG_form));
Bd1_4 = ((5*h_reac*(m_tot)^2)/((m_tot)^2*Cp_tot*rho_tot));

%Res Concentration
A2_1 = -((a*Ea*C_res)/(R*(T_reac+273.15)^2))* exp((-Ea)/(R*(T_reac
+273.15)));
A2_2 = (((q_res+q_am+q_form)/V_tot) - (a*exp((-Ea/(R*(T_reac
+273.15))))));
A2_3 = 0;
A2_4 = (((C_res * (q_res+q_am+q_form)) - (Cres_in * q_res))/
(rho_res*(V_tot)^2));
A2_5 = (((C_res * (q_res+q_am+q_form)) - (Cres_in * q_res))/
(rho_am*(V_tot)^2));
A2_6 = (((C_res * (q_res+q_am+q_form)) - (Cres_in * q_res))/
(rho_form*(V_tot)^2));
A2_7 = (((C_res * (q_res+q_am+q_form)) - (Cres_in * q_res))/
(rho_prep*(V_tot)^2));

B2_1 = 0;
B2_2 = 0;
B2_3 = (((Cres_in - C_res)*Kv_res)/(V_tot*60))
 *sqrt((rho_res*g*h_res)/(100000*SG_res));
B2_4 = -(((C_res*Kv_am)/(V_tot*60)) *sqrt((rho_am*g*h_am)/
(100000*SG_am)));
B2_5 = -(((C_res*Kv_form)/(V_tot*60)) *sqrt((rho_form*g*h_form)/
(100000*SG_form)));

Bd2_1 = 0;
Bd2_2 = 0;
Bd2_3 = 0;
Bd2_4 = 0;
```

```
%Heating Element

A3_1 = (h_h*A_h)/(m_h*Cp_h);
A3_2 = 0;
A3_3 = -(h_h*A_h)/(m_h*Cp_h);
A3_4 = 0;
A3_5 = 0;
A3_6 = 0;
A3_7 = 0;

B3_1 = 60/(m_h*Cp_h);
B3_2 = 0;
B3_3 = 0;
B3_4 = 0;
B3_5 = 0;

Bd3_1 = 0;
Bd3_2 = 0;
Bd3_3 = 0;
Bd3_4 = 0;

%Mass of resorcinol

A4_1 = 0;
A4_2 = 0;
A4_3 = 0;
A4_4 = - ((Kv_out * x_out * g * rho_H2O * (rho_res * m_res^2 +
 3 * rho_res * (m_am+m_form+m_prep)* m_res + 2 *(rho_am*m_am +
 rho_form*m_form + rho_prep*m_prep)...
    *(m_am+m_form+m_prep)) / (120*r*100000*pi*m_tot^3*rho_tot))) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A4_5 = - ((m_res*Kv_out*x_out*g*rho_H2O*(rho_am*((m_res+m_form
+m_prep)-m_am)-2*(rho_res*m_res + rho_form*m_form + ...
    rho_prep*m_prep)))/(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A4_6 = - ((m_res*Kv_out*x_out*g*rho_H2O*(rho_form*((m_res+m_am
+m_prep)-m_form)-2*(rho_res*m_res + rho_am*m_am + ...
    rho_prep*m_prep)))/(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A4_7 = - ((m_res*Kv_out*x_out*g*rho_H2O*(rho_prep*((m_res+m_am
+m_form)-m_prep)-2*(rho_res*m_res + rho_am*m_am + ...
    rho_form*m_form)))/(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));


B4_1 = 0;
B4_2 = - (m_res*m_tot*rho_tot*Kv_out)/(60*r*m_tot) * sqrt((g*rho_H2O)/
(100000*pi*m_tot*rho_tot));
B4_3 = (rho_res*Kv_res)/(60) * sqrt((rho_res*g*h_res)/
(100000*SG_res));
B4_4 = 0;
B4_5 = 0;
```

```
Bd4_1 = 0;
Bd4_2 = 0;
Bd4_3 = 0;
Bd4_4 = 0;

%Mass of Ammonia
A5_1 = 0;
A5_2 = 0;
A5_3 = 0;
A5_4 = - ((m_am*Kv_out*x_out*g*rho_H2O*(rho_res*((m_am+m_form+m_prep)-
m_res)-2*(rho_am*m_am + rho_form*m_form + rho_prep*m_prep)))...
    /(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A5_5 = - ((Kv_out * x_out * g * rho_H2O * (rho_am * m_am^2 + 3
 * rho_am * (m_res+m_form+m_prep)* m_am + 2 *(rho_res*m_res +
 rho_form*m_form + rho_prep*m_prep)...
    *(m_res+m_form+m_prep)) / (120*r*100000*pi*m_tot^3*rho_tot))) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A5_6 = - ((m_am*Kv_out*x_out*g*rho_H2O*(rho_form*((m_res+m_am+m_prep)-
m_form)-2*(rho_res*m_res + rho_am*m_am + rho_prep*m_prep)))...
    /(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A5_7 = - ((m_am*Kv_out*x_out*g*rho_H2O*(rho_prep*((m_res+m_am+m_form)-
m_prep)-2*(rho_res*m_res + rho_am*m_am + rho_form*m_form)))...
    /(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));


B5_1 = 0;
B5_2 = - (m_am*m_tot*rho_tot*Kv_out)/(60*r*m_tot) * sqrt((g*rho_H2O)/
(100000*pi*m_tot*rho_tot));
B5_3 = 0;
B5_4 = (rho_am*Kv_am)/(60) * sqrt((rho_am*g*h_am)/(100000*SG_am));
B5_5 = 0;

Bd5_1 = 0;
Bd5_2 = 0;
Bd5_3 = 0;
Bd5_4 = 0;

%Mass of Formaldehyde
A6_1 = 0;
A6_2 = 0;
A6_3 = 0;
A6_4 = - ((m_form*Kv_out*x_out*g*rho_H2O*(rho_res*((m_am
+m_form+m_prep)-m_res)-2*(rho_am*m_am + rho_form*m_form +
 rho_prep*m_prep)))...
    /(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A6_5 = - ((m_form*Kv_out*x_out*g*rho_H2O*(rho_am*((m_res
+m_form+m_prep)-m_am)-2*(rho_res*m_res + rho_form*m_form +
 rho_prep*m_prep)))...
```

```
    /(120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A6_6 = - ((Kv_out * x_out * g * rho_H2O * (rho_form * m_form^2 +
 3 * rho_form * (m_res+m_am+m_prep)* m_form + 2 *(rho_res*m_res +
 rho_am*m_am + rho_prep*m_prep)...
    *(m_res+m_am+m_prep)) / (120*r*100000*pi*m_tot^3*rho_tot))) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));
A6_7 = - ((m_form*Kv_out*x_out*g*rho_H2O*(rho_prep*((m_res
+m_am+m_form)-m_prep)-2*(rho_res*m_res + rho_am*m_am +
 rho_form*m_form)))/...
    (120*r*100000*pi*m_tot^3*rho_tot)) *
 sqrt((100000*pi*m_tot*rho_tot)/(g*rho_H2O));


B6_1 = 0;
B6_2 = - (m_form*m_tot*rho_tot*Kv_out)/(60*r*m_tot) *
 sqrt((g*rho_H2O)/(100000*pi*m_tot*rho_tot));
B6_3 = 0;
B6_4 = 0;
B6_5 = (rho_form*Kv_form)/(60) * sqrt((rho_form*g*h_form)/
(100000*SG_form));

Bd6_1 = 0;
Bd6_2 = 0;
Bd6_3 = 0;
Bd6_4 = 0;

%Mass of preparation
A7_1 = 0;
A7_2 = 0;
A7_3 = 0;
A7_4 = -
 ((m_prep*rho_res*Kv_out*x_out*((rho_res*m_res*m_tot)+(2*(rho_am*m_am
 + rho_form*m_form + rho_prep*m_prep)*(m_am+m_form+m_prep))))...
    /(120*r*m_tot^3*rho_tot)) * sqrt((g*rho_H2O)/
(100000*pi*m_tot*rho_tot));
A7_5 = -
 ((m_prep*rho_am*Kv_out*x_out*((rho_am*m_am*m_tot)+(2*(rho_res*m_res +
 rho_form*m_form + rho_prep*m_prep)*(m_res+m_form+m_prep))))...
    /(120*r*m_tot^3*rho_tot)) * sqrt((g*rho_H2O)/
(100000*pi*m_tot*rho_tot));
A7_6 = -
 ((m_prep*rho_form*Kv_out*x_out*((rho_form*m_form*m_tot)+(2*(rho_res*m_res
 + rho_am*m_am + rho_prep*m_prep)*(m_res+m_am+m_prep))))...
    /(120*r*m_tot^3*rho_tot)) * sqrt((g*rho_H2O)/
(100000*pi*m_tot*rho_tot));
A7_7 = - ((Kv_out * x_out * g * rho_H2O * (rho_prep * m_prep^2 +
 3 * rho_prep * (m_res+m_am+m_form)* m_prep + 2 *(rho_res*m_res +
 rho_am*m_am ...
    + rho_form*m_form) *(m_res+m_am+m_form)) /
 (120*r*100000*pi*m_tot^3*rho_tot))) * sqrt((100000*pi*m_tot*rho_tot)/
(g*rho_H2O));
```

```
B7_1 = 0;
B7_2 = - (m_prep*m_tot*rho_tot*Kv_out)/(60*r*m_tot) *
 sqrt((g*rho_H2O)/(100000*pi*m_tot*rho_tot));
B7_3 = 0;
B7_4 = 0;
B7_5 = 0;

Bd7_1 = 0;
Bd7_2 = 0;
Bd7_3 = 0;
Bd7_4 = 0;


%Defining matrices


%Vector Matrix
A = [A1_1, A1_2, A1_3, A1_4, A1_5, A1_6, A1_7;
     A2_1, A2_2, A2_3, A2_4, A2_5, A2_6, A2_7;
     A3_1, A3_2, A3_3, A3_4, A3_5, A3_6, A3_7;
     A4_1, A4_2, A4_3, A4_4, A4_5, A4_6, A4_7;
     A5_1, A5_2, A5_3, A5_4, A5_5, A5_6, A5_7;
     A6_1, A6_2, A6_3, A6_4, A6_5, A6_6, A6_7;
     A7_1, A7_2, A7_3, A7_4, A7_5, A7_6, A7_7;];

%Input Matrix
B = [B1_1, B1_2, B1_3, B1_4, B1_5;
     B2_1, B2_2, B2_3, B2_4, B2_5;
     B3_1, B3_2, B3_3, B3_4, B3_5;
     B4_1, B4_2, B4_3, B4_4, B4_5;
     B5_1, B5_2, B5_3, B5_4, B5_5;
     B6_1, B6_2, B6_3, B6_4, B6_5;
     B7_1, B7_2, B7_3, B7_4, B7_5;];

%Disturbance Matrix
Bd = [Bd1_1, Bd1_2, Bd1_3, Bd1_4;
      Bd2_1, Bd2_2, Bd2_3, Bd2_4;
      Bd3_1, Bd3_2, Bd3_3, Bd3_4;
      Bd4_1, Bd4_2, Bd4_3, Bd4_4;
      Bd5_1, Bd5_2, Bd5_3, Bd5_4;
      Bd6_1, Bd6_2, Bd6_3, Bd6_4;
      Bd7_1, Bd7_2, Bd7_3, Bd7_4;];


%Output Matrix
C = [1 0 0 0 0 0 0;
     0 1 0 0 0 0 0;];

D = zeros(2,5);

%Representing the matrices as a state-space and then converting to TF
sys = tf(ss(A,B,C,D));
```