

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет

вычислительной техники

Кафедра

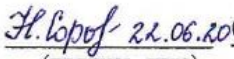
МОиПЭВМ

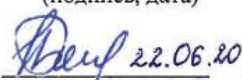
Направление подготовки	09.04.04 «Программная инженерия»
Магистерская программа	Программное обеспечение вычислительной техники и автоматизированных систем

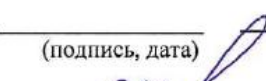
МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему

**Модель и алгоритм поддержки адаптивного управления
электрообогревом технологического оборудования атомной
электростанции**

Студент	 22.06.20	Сорокина Надежда Викторовна
	(подпись, дата)	(ФИО полностью)

Руководитель	 22.06.20	Балашова И. Ю.
	(подпись, дата)	(фамилия, инициалы)

Нормоконтролёр	 22.06.20	Такташкин Д. В.
	(подпись, дата)	(фамилия, инициалы)

Рецензент	 22.06.20	Шеланков О. Е.
	(подпись, дата)	(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой	_____	Макарычев П.П.
	(подпись)	(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК	_____	Попова Н.А.
	(подпись)	(фамилия, инициалы)

Пенза, 2020

Содержание

Введение.....	4
1 Анализ моделей и алгоритмов управления технологическими процессами атомной электростанции	9
1.1 Анализ предметной области	9
1.2 Анализ математических моделей технологических процессов атомной электростанции	11
1.3 Исследование и анализ алгоритмов управления технологическими процессами.....	13
1.3.1 ПРОТАР 120	13
1.3.2 Комплекс ПАССАТ	14
1.3.3 Средства автоматизации для системы контроля и управления электрообогревом натрия	15
1.3.4 Анализ алгоритмов управления технологическими процессами.....	17
1.4 Анализ алгоритмов автоматического регулирования	18
1.5 Методы настройки коэффициентов	23
1.6 Анализ требований к модели и алгоритму поддержки адаптивного управления	26
1.6.1 Функциональные требования	27
1.6.2 Нефункциональные требования	30
1.7 Выбор технологии разработки.....	31
1.8 Выбор инструментальных средств разработки.....	32
Выводы	33
2 Математическая модель и алгоритм поддержки адаптивного управления электрообогревом	35
2.1 Математическая модель управления электрообогревом технологического оборудования атомной электростанции.....	35
2.2 Разработка алгоритма поддержки адаптивного управления электрообогревом.....	40
Выводы	47

3	Реализация модели и алгоритма поддержки адаптивного управления и оценка их эффективности.....	49
3.1	Техническая реализация	49
3.2	Архитектура базовых классов предметной области.....	50
3.3	Программная реализация	52
3.4	Тестирование модели и алгоритма поддержки адаптивного управления электрообогревом	57
3.4.1	Функциональное тестирование.....	58
3.4.2	Проведение экспериментов и анализ результатов.....	60
	Выводы	63
	Заключение	64
	Список использованных источников	66
	Приложение А. Глоссарий	69
	Приложение Б. Листинг программных средств	71
	Приложение В. Результаты тестирования	89
	Приложение Г. Акт о внедрении результатов.....	97

Введение

Актуальность темы исследования. Атомные электростанции (АЭС) предназначены для производства электроэнергии и включают в себя комплекс систем, устройств, оборудования и сооружений. Основными преимуществами атомных электростанций являются огромная энергоемкость, возможность повторного использования расщепляющего материала (урана), снижение «парникового эффекта» по сравнению с тепловыми электростанциями.

Атомная электростанция представляет собой сложный технический комплекс, и функционирующие внутри ее подсистемы требуют автоматизации процессов управления на базе компьютерной техники. Без автоматизированных систем управления (АСУ) и систем автоматического управления (САУ) на базе микроконтроллеров и микропроцессоров люди физически не смогут справиться с задачами управления, заключающимися в обработке большого количества информации и формирования на их основе управляющих воздействий на технологический объект в масштабе реального времени. Центральным звеном АСУ и САУ является программное обеспечение (ПО), которое построено на базе моделей и алгоритмов управления, имеет практическую ценность и реализует необходимую функциональность.

При использовании жидкого натрия в качестве теплоносителя важное место в цикле занимает система поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции.

Алгоритмы управления технологическими процессами в готовых решениях, таких как ПРОТАР 120, ПАССАТ, средства автоматизации для контроля и управления электрообогревом натрия (СА СКУ ЭО), построены на похожих принципах и имеют общие недостатки. Среди основных недостатков можно выделить следующие: алгоритмы могут работать только в ограниченных режимах работы без возможности их расширения, процесс перенастройки этих алгоритмов для новой задачи очень трудоемкий и в

полной мере не осуществимый. Также следует отметить, что программное обеспечение, реализующее данные алгоритмы, не поддерживает принципы объектно-ориентированного программирования, что ограничивает возможности модификации, повторного использования и расширяемости без изменения самой структуры программы.

При модернизации оборудования возникает необходимость в оценке возможных изменений и соответствующей корректировке математического обеспечения. Описанные в открытых источниках математические модели динамики технологических процессов (Митропольский А. Н., Цукерман И. Х., Петренко А. А., Демченко В. А.) не учитывают все существующие режимы управления и не позволяют составить целостное представление об организации управления электрообогревом технологического оборудования атомной электростанции.

Таким образом, задача разработки модели и алгоритма поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции является актуальной.

Объектом исследования является система адаптивного управления электрообогревом технологического оборудования.

Предметом исследования являются математические модели и алгоритмы управления технологическими процессами.

Целью магистерской диссертации является разработка модели и алгоритма, которые обеспечивают поддержку адаптивного управления электрообогревом технологического оборудования атомной электростанции.

Задачи исследования. Основные задачи исследования:

- провести анализ известных математических моделей и алгоритмов управления технологическими процессами атомной электростанции;
- разработать математическую модель управления электрообогревом технологического оборудования атомной электростанции для всех существующих режимов управления;

- разработать базовый алгоритм поддержки адаптивного управления, который может быть применен для различных технологических процессов в зависимости от изменения параметров, коэффициентов и физических величин;

- модифицировать базовый алгоритм для системы управления электрообогревом технологического оборудования атомной электростанции;

- разработать программное обеспечение поддержки адаптивного управления электрообогревом.

Методы исследования. В процессе исследования и разработки были использованы методы анализа и синтеза, обобщения, сравнения, математического моделирования, объектно-ориентированного моделирования.

Научная новизна работы заключается в том, что:

- разработана математическая модель управления электрообогревом технологического оборудования атомной электростанции, которая, в отличие от известных моделей, учитывает все существующие режимы управления электрообогревом;

- разработан базовый алгоритм поддержки адаптивного управления, который, в отличие от известных алгоритмов, может быть использован для разных технологических процессов в зависимости от изменения параметров, коэффициентов и физических величин;

- разработан алгоритм управления электрообогревом технологического оборудования атомной электростанции, который, в отличие от известных, является адаптивным и обеспечивает автоматизированное управление электрическим нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима управления;

- разработано программное обеспечение поддержки адаптивного управления электрообогревом в соответствии с принципами объектно-

ориентированного программирования, что дает возможность модификации, расширяемости и повторного использования.

Практическая значимость исследования. Практическая ценность заключается в разработке на основе предложенной математической модели и алгоритма управления программного обеспечения поддержки адаптивного управления электрообогревом. Разработанное программное обеспечение включено в состав опытного образца программно-технического комплекса для создания и модернизации систем контроля и управления технологическим оборудованием ПТК «УРАЛ-Атом» и успешно используется при проведении его комплексной наладки, испытаний и эксплуатации, что подтверждает акт о внедрении результатов.

Основные положения, выносимые на защиту. На защиту выносятся следующие положения:

- математическая модель управления электрообогревом технологического оборудования атомной электростанции, которая, в отличие от известных моделей, учитывает все существующие режимы управления электрообогревом;
- базовый алгоритм поддержки адаптивного управления, который, в отличие от известных алгоритмов, может быть использован для разных технологических процессов в зависимости от изменения параметров, коэффициентов и физических величин;
- алгоритм управления электрообогревом технологического оборудования атомной электростанции, который, в отличие от известных, является адаптивным и обеспечивает автоматизированное управление электрическим нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима управления;
- программное обеспечение поддержки адаптивного управления электрообогревом, которое, в отличие от известных, поддерживает принципы

объектно-ориентированного программирования, что дает возможность модификации, расширяемости и повторного использования.

Апробация работы. Основные положения, выносимые на защиту, и результаты диссертационной работы докладывались и обсуждались на конференциях:

- XV Международная научно-техническая конференция «Новые информационные технологии и системы» (г. Пенза, 2018 год);
- VI Ежегодная всероссийская межвузовская научно-техническая конференция «Информационные технологии в науке и образовании. Проблемы и перспективы» (г. Пенза, 2019 год);
- XVI Международная научно-техническая конференция «Новые информационные технологии и системы» (г. Пенза, 2019 год);
- VII Ежегодная всероссийская межвузовская научно-техническая конференция «Информационные технологии в науке и образовании. Проблемы и перспективы» (г. Пенза, 2020 год).

Публикации. Основные результаты диссертационного исследования опубликованы в 4 печатных работах, зарегистрированных в РИНЦ.

Структура и объем работы. Магистерская диссертация состоит из введения, 3 тематических глав, заключения, списка использованных источников и приложений. Общий объем работы 98 страниц, включая 29 рисунков, 3 таблицы. Список литературы изложен на 3 страницах и включает 28 источников.

1 Анализ моделей и алгоритмов управления технологическими процессами атомной электростанции

1.1 Анализ предметной области

Атомная электростанция (АЭС) представляет собой комплекс систем, устройств, оборудования и сооружений, предназначенный для генерации электроэнергии. Наличие ядерного реактора служит существенным отличием АЭС от других видов электростанций.

На атомной электростанции происходит три взаимных преобразования форм энергии. Ядерная энергия переходит в тепловую, тепловая энергия – в механическую, а механическая энергия преобразуется в электрическую. На первом этапе важную роль играет реактор, в который загружается ядерное топливо и где протекает управляемая цепная реакция. В качестве ядерного топлива практически всегда используется уран, который делится медленными нейтронами. В результате этого деления происходит выделение очень большого количества тепла. Активная зона – это основной элемент реактора, она расположена в бетонной шахте, и в ней протекает управляемая цепная реакция деления. На втором этапе для отвода тепла из активной зоны используется теплоноситель. В качестве теплоносителя может использоваться вода или жидкий натрий. В нашем случае важно использование последнего. Тепловая энергия, полученная от теплоносителя, применяется для получения пара в парогенераторе. На третьем этапе механическая энергия пара поступает к турбогенератору, где она превращается в электрическую энергию и дальше по проводам поступает к потребителям.

На российских атомных электростанциях в зависимости от типа реактора могут быть одноконтурные, двухконтурные и трехконтурные схемы движения теплоносителя. Одноконтурные схемы используются на атомных электростанциях с реакторами большой мощности канальными (РБМК). Реактор функционирует в блоке с двумя генераторами и двумя конденсационными турбинами. В качестве парогенератора выступает сам

кипящий реактор, что и дает возможность применения одноконтурной схемы. Данная схема довольно проста, однако радиоактивность распространяется на все компоненты блока, что усложняет биологическую защиту. Двухконтурные схемы используются на атомных электростанциях с водо-водяными энергетическими реакторами (ВВЭР). Вода под давлением подается в активную зону реактора. Затем она нагревается. В парогенераторе для образования насыщенного пара используется энергия теплоносителя – воды. Второй контур не радиоактивен. Трехконтурная схема используется на атомных электростанциях с реакторами на быстрых нейтронах (БН) с натриевым теплоносителем [1]. В первом контуре в качестве теплоносителя применяется радиоактивный натрий, во втором – нерадиоактивный натрий, в третьем – вода. Наличие трех контуров необходимо, что исключить контакт радиоактивного натрия с водой. В реакторе на быстрых нейтронах тепловыделение в десять – пятнадцать раз больше, чем в реакторах РБМК и ВВЭР, поэтому использование в качестве теплоносителя воды не целесообразно для передачи такого большого объема энергии. Согласно опытным данным температура натрия на входе в реактор порядка 370°C , а на выходе из реактора около 550°C .

Применение натрия в качестве теплоносителя вызывает необходимость включения в состав комплекса системы поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции. К задачам этой системы относится контроль и регулирование температуры тепловых зон, требуемых скоростей разогрева и точности поддержания температуры, а также диагностирования технологических ситуаций. Электрообогрев оборудования первого и второго контура применяется, чтобы разогревать оборудование и трубопроводы установки перед заполнением рабочей средой расплавом натрия, поддерживать их в разогретом состоянии при эксплуатации, разогревать перед дренированием рабочей среды, разогревать после замораживания рабочей среды. Он также используется для обеспечения режима регенерации и снижения потерь тепла

с оборудования и трубопроводов. Процесс электрообогрева осуществляется посредством нагревателей, объединяемых в тепловые зоны. Питание тепловых зон производится от секций электрощитов через электромагнитные контакторы. Контроль температуры тепловых зон производится с помощью термоэлектрических преобразователей с унифицированными выходными сигналами 4 – 20 мА.

Система управления электрообогревом входит в программно-технический комплекс (ПТК) «УРАЛ-Атом», разрабатываемый АО «НПП «Рубин». ПТК «УРАЛ-Атом» разрабатывается для создания и модернизации систем контроля и управления технологическим оборудованием атомных электростанций в соответствии с «Программой инновационного развития и технологической модернизации Госкорпорации «Росатом» на период до 2030 года (в гражданской части)» [2].

Глоссарий предметной области разработки представлен в приложении А.

1.2 Анализ математических моделей технологических процессов атомной электростанции

В открытых источниках очень мало представлено математических моделей технологических процессов атомной электростанции. Небольшая часть представленных моделей не содержит полного описания процесса управления.

В работе [3] описана математическая модель динамики процесса электрообогрева трубопровода. Математическое описание динамики процесса электрообогрева трубопровода выполнено на основе анализа последовательного прохождения теплового потока от жилы нагревателя к обогреваемой поверхности, а также через теплоизоляцию в окружающую среду. Математическая модель представлена в виде системы дифференциальных уравнений. Решение этих уравнений дает реакцию объекта на включение тепловой зоны. Следует отметить, что математическая модель не учитывает режимы управления электрообогревом, что

немаловажно в процессе управления, поскольку могут возникнуть аварийные ситуации, препятствующие автоматическому регулированию.

В работе [4] представлены математические модели динамики технологических участков, аппаратов и устройств. Математическое описание этих процессов представлено больше с физической точки зрения, чем с точки зрения управляющего процесса и режимов управления.

Обобщив все вышесказанное, возникает необходимость создания математической модели электрообогрева технологического оборудования атомной электростанции, которая учитывает все существующие режимы управления и позволяет вовремя предотвратить аварийные ситуации. Также следует отметить, что при модернизации оборудования возникает необходимость в оценке возможных изменений и обновлении математического обеспечения.

Использование блочного принципа при разработке математической модели сложного процесса электрообогрева технологического оборудования позволит выделить основные режимы управления электрообогревом и тем самым облегчить процесс моделирования.

Имитационное моделирование – это класс математического моделирования, который используется для исследования поведения объекта в различных ситуациях в режиме реального времени. Основными преимуществами создания имитационной модели являются: управление временем (замедление или ускорение) и возможность экспериментального тестирования, которое опасно или невозможно на реальном оборудовании [5].

Разработанная математическая модель послужит основой для имитационного моделирования процесса электрообогрева, необходимого для проведения тестирования разработанного алгоритма адаптивного управления.

1.3 Исследование и анализ алгоритмов управления технологическими процессами

Среди современных методов управления выделяют: упреждающее управление и адаптивное управление.

Упреждающее управление основано на том, чтобы проводить компенсацию значения регулируемой переменной согласно теоретическим предсказаниям на основе измерения других параметров, которые влияют на переменную в рамках математической модели процесса.

Адаптивное управление основано на том, что управляющие параметры или алгоритмы управления приспособляются к возмущающим воздействиям с целью оптимального управления системой [6].

1.3.1 ПРОТАР 120

ПРОТАР 120 – регулирующий программируемый микропроцессорный прибор с автоматизированной настройкой параметров. ПРОТАР 120 используется в автоматизированных системах управления и в небольших подсистемах регулирования в различных отраслях промышленности для стабилизации технологических параметров, программного, каскадного, многосвязного регулирования с реализацией сложных алгоритмов обработки информации. Имеется возможность создания алгоритма функционирования пользователем, при этом специальных знаний от него не требуется.

ПРОТАР 120 реализует следующие функции:

- переключение режимов управления с автоматического на ручное и обратно;
- ручное управление с помощью пульта оператора;
- цифровая индикация входных и выходных аналоговых сигналов, параметров настройки и переменных, входящих в структуру прибора, кода отказа;
- формирование алгоритма диагностики отказа;

- формирование внутреннего дискретного сигнала, показывающего установленный режим управления.

Основными достоинствами ПРОТАР 120 являются:

- алгоритм автоматизированной настройки динамических параметров ПИД-регулятора с возбудителем автоколебаний;
- возможность дополнительной реализации функций, необходимых пользователям;
- возможность перенастройки структуры функционирования прибора.

1.3.2 Комплекс ПАССАТ

Комплекс ПАССАТ – это программно-аппаратные средства систем автоматизации технологических процессов, применяемые для построения и модернизации автоматизированных систем управления технологическим процессом (АСУ ТП) на энергоблоках атомных станций и других промышленных объектах. Комплекс ПАССАТ используется на Нововоронежской, Ленинградской и Белоярской АЭС.

Комплекс ПАССАТ реализует:

- прием аналоговых и дискретных сигналов;
- формирование и выдачу на исполнительный механизм команд управления;
- резервирование выполняемых функций;
- восстановление функциональных модулей без отключения питания контроллера;
- диагностику аппаратных средств;
- сохранность обрабатываемой технологической информации и программных средств при исчезновении питания, авариях и отказах, которая обеспечивается энергонезависимой памятью;
- обмен данными с внешними системами по сети Ethernet;

- обмен информацией между контроллерами по интерфейсу RS-232 и RS-485.

Комплекс ПАССАТ может работать в следующих режимах:

- автоматический режим (управление в соответствии с алгоритмом);
- резервный автоматический режим (контролирует работу основного блока);
- ручной режим (управление с помощью кнопок на лицевой панели блока);
- режим безопасного состояния (каналы вывода в выключенном состоянии, управление с помощью кнопок невозможно);
- тестовый режим (проверка блока с помощью стенда настройки и диагностики) [7].

1.3.3 Средства автоматизации для системы контроля и управления электрообогревом натрия

Средства автоматизации для системы контроля и управления электрообогревом натрия четвертого энергоблока Белоярской АЭС с реактором на быстрых нейтронах типа БН-800 (СА СКУ ЭО) предназначены для контроля и регулирования температуры тепловых зон, а также для диагностирования технологических ситуаций, оборудования и аппаратуры. Средства автоматизации обеспечивают функционирование заданного алгоритма разогрева и поддерживают точность температуры тепловых зон четвертого энергоблока Белоярской АЭС.

Режимы работы электрообогрева:

- начальный разогрев системы до 250°C после монтажа или ремонта перед заполнением оборудования и коммуникаций натрием;
- поддержание температуры обогреваемого оборудования на уровне 250°C;

- нагрев отдельных технологических участков при застывании натрия от 20°C до 250°C;
- разогрев систем, заполненных продуктами взаимодействия натрия с водой, до температуры 450°C с целью расплавления продуктов взаимодействия и удаления их через штатные дренажные системы;
- нагрев и поддержание температуры отдельных тепловых зон до 550°C.

Для предотвращения разрывов трубопроводов с «замороженным» теплоносителем их разогрев ведется в направлении от свободных поверхностей натрия.

СА СКУ ЭО выполняют следующие функции:

- сбор и контроль достоверности данных о состоянии технологической среды и исполнительных механизмов;
- прием и обработку данных от датчиков температуры;
- авторегулирование по пропорциональному закону;
- выдача команд управления подогревом;
- выдача сигналов технологической сигнализации;
- дистанционное управление секциями подогрева натрия;
- функционально-групповое управление, выполняемое в интерактивном режиме с оператором-технологом, осуществляющим контроль и управление технологическим процессом подогрева натрия с использованием автоматизированного рабочего места системы верхнего блочного уровня;
- локальное отображение данных в режиме реального времени;
- архивирование данных;
- диагностика секций подогрева натрия на основании полученных данных;
- самодиагностика программно-технических средств.

Средства автоматизации для системы контроля и управления электрообогревом натрия относятся к системам длительного непрерывного использования.

1.3.4 Анализ алгоритмов управления технологическими процессами

Алгоритмы управления технологическими процессами, лежащие в основе готовых решений: ПРОТАР 120, ПАССАТ, СА СКУ ЭО, – построены на похожих принципах и имеют общие недостатки.

Первый существенный недостаток заключается в том, что эти алгоритмы работают только в заданных режимах управления, и нет возможности их расширения ввиду того, что часть функций выполняется на уровне аппаратных средств.

Второй недостаток заключается в том, что процесс перенастройки этих алгоритмов для новой задачи очень трудоемкий и в полной мере не осуществимый, так как каждый алгоритм ориентирован для выполнения своей конкретной задачи. В условиях быстроразвивающихся современных технологий необходимо разработать базовый алгоритм поддержки адаптивного управления, который станет основой для применения в технологических системах.

Важно отметить, что программное обеспечение, реализованное на основе этих алгоритмов управления, не поддерживает принципы объектно-ориентированного программирования (ООП). Из вышесказанного следует, что ограничены возможности модификации, повторного использования и расширяемости без изменения самой структуры программы.

На основе проведенного анализа алгоритмов управления технологическими процессами принято решение о необходимости создания базового алгоритма поддержки адаптивного управления, который может быть применен для разных технологических процессов. Под адаптивным управлением будем понимать автоматическое изменение режима работы алгоритма в зависимости от изменения параметров объекта управления или

действий на объект со стороны эксплуатирующего персонала. Адаптивное управление является эффективным, поскольку позволяет алгоритму приспосабливаться к изменяющимся условиям. Затем на основе базового алгоритма предполагается разработать алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции. При реализации алгоритма будут использоваться свободно распространяемые инструментальные средства и поддерживаться методология объектно-ориентированного программирования, что позволит модифицировать, масштабировать библиотеку классов и изменять параметры алгоритма с незначительным изменением исходного кода.

1.4 Анализ алгоритмов автоматического регулирования

В настоящее время получило большое распространение использование в системах контроля и управления технологическим оборудованием различных программно-аппаратных контроллеров. Одни контроллеры работают на основе алгоритма ПИД-регулирования, другие – принципа самонастройки, третьи работают на основе нечеткой логики, а четвертые – на основе нейронных сетей. Сравним алгоритмы для выбора наиболее подходящего для системы управления электрообогревом технологического оборудования АЭС.

Пропорционально-интегрально-дифференциальный регулятор (ПИД-регулятор) используется в системах автоматического регулирования для формирования управляющего сигнала с целью получения необходимой точности и обеспечения качества переходного процесса. Уравнение ПИД-регулятора

$$u(t) = P + I + D = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt},$$

где u – выходной сигнал регулятора (управляющее воздействие);

t – время;

P – пропорциональная составляющая;

I – интегральная составляющая;

D – дифференциальная составляющая;

K_p – коэффициент пропорциональной составляющей;

e – рассогласование (разность датчика и задатчика);

K_i – коэффициент интегральной составляющей;

τ – время;

K_d – коэффициент дифференциальной составляющей.

Пропорциональная составляющая отвечает за рассогласование между задатчиком и регулируемой величиной (датчиком) в настоящий момент времени. Выходной сигнал тем больше, чем больше рассогласование. Если датчик равен задатчику, то выходной сигнал равен нулю. Однако использование только пропорциональной составляющей не целесообразно, поскольку тогда значение регулируемой величины никогда не стабилизируется на конкретном значении. Интегральная составляющая позволяет уменьшить до нуля установившуюся ошибку в системе, которая не позволяет использовать только пропорциональную компоненту. При неправильном подборе коэффициента интегральной составляющей, система может перейти к автоколебаниям. Дифференциальная составляющая предназначена для противодействия отклонениям от целевого значения [8, 9]. На рисунке 1 представлена система с ПИД-регулятором.

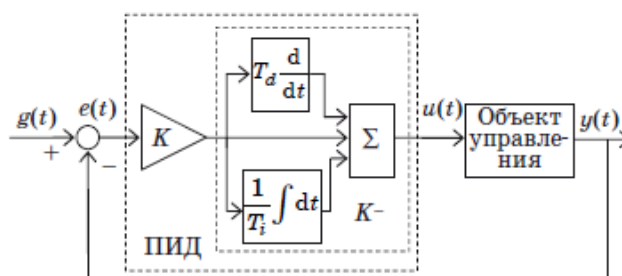


Рисунок 1 – Система с ПИД-регулятором

Несомненными достоинствами ПИД-регуляторов являются высокая надежность, простота реализации и хорошее качество управление при правильной настройке коэффициентов. К недостаткам следует отнести необходимость перенастройки алгоритма посредством изменения коэффициентов, уставок при изменении параметров объекта управления.

Персоналом атомной электростанции процедура перенастройки ПИД-регулятора осуществляется вручную с помощью стендов и является достаточно трудоемкой. Предсказать заранее, какой коэффициент будет обеспечивать корректную работу алгоритма, очень трудно, и в большинстве случаев возможно только экспериментальным путем. Определенные трудности создает и круглосуточный режим работы оборудования [9, 10].

Самонастраивающийся контроллер – усовершенствованный вариант ПИД-контроллера, в котором присутствует функция автоматической адаптации его коэффициентов параметрам объекта управления. На рисунке 2 представлена структура самонастраивающегося контроллера.

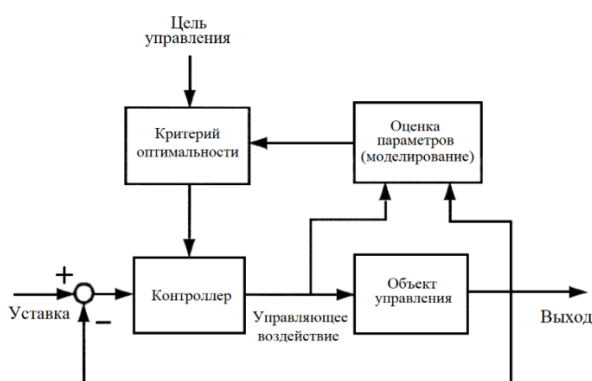


Рисунок 2 – Структура самонастраивающегося контроллера

В отличие от ПИД-контроллера в структуре самонастраивающегося контроллера присутствуют два дополнительных блока: критерий оптимальности и оценка параметров (смотри рисунок 2). Следовательно, коэффициенты пропорциональной, интегральной и дифференциальной составляющей в контроллере устанавливаются автоматически на основе данных о текущем состоянии объекта управления [11].

Нечетко-логический контроллер предназначен для регулирования выходов процесса в окрестности заданной уставки. Контроллер на основе нечеткой логики состоит из набора условных лингвистических операторов или правил, которые еще называют нечеткие ассоциативные матричные правила или НАМ-правила. Каждое правило задает конкретную ситуацию управления. Условные лингвистические операторы могут быть получены на

основе данных о технологическом процессе, которым требуется управлять. Система управления на основе нечеткой логики представлена на рисунке 3.

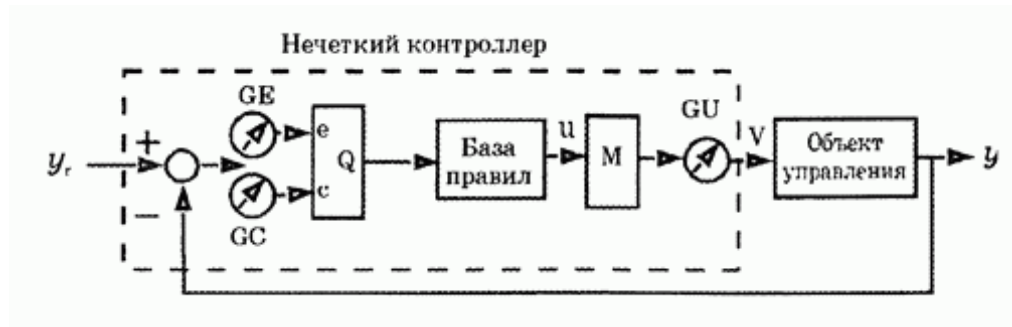


Рисунок 3 – Система управления на основе нечеткой логики

В каждом нечетком ассоциативном матричном правиле могут быть использованы три переменные: ошибка (E), скорость изменения ошибки (C) и изменение управляющего воздействия (U). Все эти переменные задаются фиксированными универсальными множествами, определяющими диапазоны измерений величин, возможные в данной системе управления. Нечетко-логический контроллер имеет несколько параметров, которые необходимо определить заранее. Такими параметрами являются коэффициент масштабирования GE для ошибки процесса, коэффициент масштабирования для изменения ошибки GC , коэффициент масштабирования для выхода контроллера GU . Величины ошибки и изменения ошибки представляют собой элементы соответствующих универсальных множеств, которые можно представить во времени функциями $e(t)$ и $c(t)$, где t – момент дискретизации. Среди преимуществ использования нечетко-логических контроллеров можно выделить простоту реализации, успешное применение для нелинейных, изменяющихся во времени процессах с неизвестной динамикой. Кроме того, для контроллеров на основе нечеткой логики не требуется математическая модель объекта управления. Среди недостатков следует отметить то, что идея нечеткого управления, реализующая в определенной степени методы интеллектуального управления, в неявном виде предполагает дополнительную реализацию возможностей адаптации. Например, если

возникнет ситуация, для которой отсутствует связанное с ней правило нечеткого вывода [12].

Контроллеры на основе нейронных сетей предполагают использование полностью определенных нейронных сетей для выработки управляющих сигналов. При правильном выборе нейронной модели и достаточным объемом информации нейронную сеть можно обучить практически любым функциям. В общей схеме нейронного управления при отсутствии возмущающих воздействий для получения результатов используется контроллер прямой связи (КПС) и контроллер обратной связи (КОС). На рисунке 4 представлена общая схема нейронного управления.

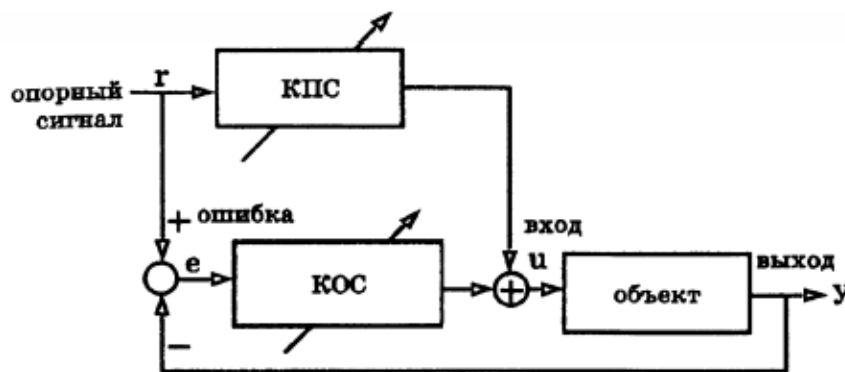


Рисунок 4 – Общая схема нейронного управления при отсутствии возмущающих воздействий

Контроллер обратной связи применяется для стабилизации замкнутой системы. Контроллер прямой связи используется для быстрой передачи выработанного входного сигнала на объект управления, что дает высокую скорость реакции. Среди преимуществ использования контроллеров на основе нейронных сетей можно выделить возможность обучения любым функциям, пригодность для управления в условиях неопределенности, способность параллельной обработки информации. Такие контроллеры являются одними из лучших для решения сложных нелинейных задач управления. Среди недостатков следует отметить то, что данные алгоритмы трудоемки для понимания и реализации [13].

Обобщив все плюсы и минусы использования каждого алгоритма, для наглядного представления составим сводную таблицу (таблица 1).

Таблица 1 – Сравнительная характеристика алгоритмов управления

Критерий сравнения	ПИД-регулятор	Управление на основе нечеткой логики	Управление на основе нейронных сетей
Простота понимания и реализации	+	+	-
Надежность	+	+	+
Высокое качество управления	+	+	+
Самоадаптация алгоритма при изменении параметров объекта управления	-	Присутствует возможность реализации	+
Управление в условия неопределенности	-	-	+

Анализируя каждый алгоритм автоматического регулирования и его возможности по сравнению с другими алгоритмами (смотри таблицу 1), можно сделать вывод, что ПИД-регулятор и алгоритм управления на основе нечеткой логики отвечают всем требованиям, необходимым для системы управления электрообогревом технологического оборудования АЭС: простота понимания и реализации, как для самих разработчиков алгоритма, так и для операторов АЭС, надежность, хорошее качество управления. Алгоритм управления на основе нейронных сетей предоставляет дополнительные возможности, но сложен для понимания и реализации [14]. Окончательный выбор сделан в пользу ПИД-регулятора, поскольку мы имеем дело с технологическим процессом в режиме реального времени и предсказать заранее все ситуации в виде правил вряд ли удастся, а ошибка может дорого стоить.

1.5 Методы настройки коэффициентов

Согласно опытным данным, полученным из работающих систем управления, для регулирования температуры в системе управления электрообогревом технологического оборудования АЭС наилучший

результат достигается при использовании ПИ-регулятора, поскольку датчики температуры оснащены нормирующими преобразователями, которые позволяют получать значения без помех. При использовании ПИ-регулятора дифференциальная составляющая отсутствует. Главной проблемой при применении в системах ПИ-регулятора является настройка коэффициентов пропорциональной и интегральной составляющей, поскольку она зависит от множества факторов и характеристик системы [15]. В настоящий момент известно несколько способов настройки коэффициентов ПИ-регулятора. Для адаптивной системы управления электрообогревом технологического оборудования могут быть применены метод Циглера-Никольса, метод CHR (Chein, Hrones и Reswick) и метод ручной настройки, основанной на правилах.

Методы Циглера-Никольса являются приближенными методами настройки. Различают настройку по процессу двухпозиционного регулирования по релейному закону и настройку по методу максимального коэффициента усиления. Первый метод заключается в том, что систему необходимо перевести в режим двухпозиционного регулирования по релейному закону и определить амплитуду a и период колебаний t_c .

На основании этих данных для ПИ-регулятора вычислить коэффициенты по формулам:

$$K_p = \frac{0.4}{a},$$

$$K_i = 0.8t_c.$$

Второй метод настройки коэффициентов используется, если допустим колебательный процесс, при котором значение регулируемой величины значительно выходит за установленные пределы. Сначала определяется предельный коэффициент K_{max} усиления, при котором система переходит в колебательный режим при значении коэффициента интегрирования $K_i = 0$. Увеличиваем K_{max} , начиная с $K_{max} = 0$, до тех пор, пока система переходит в колебательный режим. Управление осуществляется с помощью

П-регулятора. Определяем период колебаний t_c при K_{max} . На основании полученных данных вычисляем коэффициенты ПИ-регулятора по формулам:

$$K_p = 0.45K_{max},$$

$$K_i = 0.8t_c.$$

К преимуществам методов Циглера-Никольса относится простота понимания и использования. К недостаткам следует отнести то, что не учитываются требования к запасу устойчивости и полученные коэффициенты следует корректировать.

Метод CHR (Chein, Hrones и Reswick) основан на использовании в качестве критерия качества настройки максимальную скорость нарастания при отсутствии перерегулирования или при наличии не более чем 20% перерегулирования. Данный метод позволяет получить больший запас устойчивости, чем в методе Циглера-Никольса. Метод CHR дает две разные системы параметров регулятора. Одна из них получена при наблюдении отклика на изменение уставки, другая – при наблюдении отклика на внешние возмущения. Если для регулятора важно качество регулирования при изменении уставки, то следует выбирать первую систему. Если в задачи регулятора входит ослабление внешних воздействий, то следует выбирать вторую систему. Метод CHR использует аппроксимацию реального объекта моделью первого порядка с задержкой. Вычислить коэффициента ПИ-регулятора по методу CHR по отклику на изменение уставки без перерегулирования можно с использованием следующих формул:

$$K_p = \frac{0.35}{a},$$

$$K_i = \frac{1.2L}{K_p},$$

где a – амплитуда тестового импульса;

L – длительность задержки на входное воздействие.

Вычислить коэффициента ПИ-регулятора по методу CHR по отклику на изменение уставки с 20% перерегулированием можно с использованием следующих формул:

$$K_p = \frac{0.6}{a},$$

$$K_i = \frac{L}{K_p}.$$

Основными преимуществами данного метода являются простота понимания процесса настройки и меньшее время, чем у метода Циглера-Никольса. Недостатки такие же, как и у метода Циглера-Никольса: неполная информация о запасе устойчивости системы, которая определяет надежность работы регулятора, и приближенная настройка [16].

Вычисление коэффициентов по формулам не может дать оптимальной настройки ПИ-регулятора, поскольку полученные результаты основываются на сильно упрощенных моделях объекта. После расчета коэффициентов регулятора необходимо сделать его подстройку. Существуют правила, полученные из опыта, теоретического анализа и численных экспериментов. При увеличении пропорционального коэффициента увеличивается быстродействие и снижается запас устойчивости. При уменьшении интегральной составляющей ошибка регулирования с течением времени уменьшается быстрее. При уменьшении постоянной интегрирования уменьшается запас устойчивости. Использование вышеперечисленных правил возможно только после предварительной настройки ПИ-регулятора по формулам [17].

1.6 Анализ требований к модели и алгоритму поддержки адаптивного управления

Ключевым этапом процесса разработки модели и алгоритма адаптивного управления электрообогревом технологического оборудования является анализ требований. Он включает в себя сбор требований, их систематизацию, выявление взаимосвязей и документирование.

Принято выделять функциональные и нефункциональные требования. Функциональные требования дают ответ на вопрос, что должно быть сделано. Они определяют задачи, которые система должна выполнить,

устанавливают связь входа и выхода в поведении системы. Нефункциональные требования идентифицируют свойства, которыми система должна обладать, или ограничения, которые она должна поддерживать, не относящиеся к поведению системы [18].

1.6.1 Функциональные требования

На основе анализа предметной области и постановки задачи на разработку были выделены основные функциональные требования к модели и алгоритму:

- прием и обработка входных сигналов;
- реализация функции технологической защиты (ТЗ);
- блокировки;
- контроль работы дистанционного управления (ДУ);
- контроль работы функционально-группового управления (ФГУ);
- автоматическое регулирование (АР);
- взаимодействие с верхним уровнем.

С учетом вышеперечисленных функциональных требований была построена диаграмма вариантов использования для оперативного персонала АЭС, представленная на рисунке 5.

Прием и обработка входных сигналов включают в себя:

- масштабирование;
- фильтрацию;
- контроль достоверности;
- задание уставок и коэффициентов.

Масштабирование предполагает преобразование входного токового сигнала из миллиампер в градусы Цельсия.

Фильтрация состоит в экспоненциальном сглаживании значения температуры.

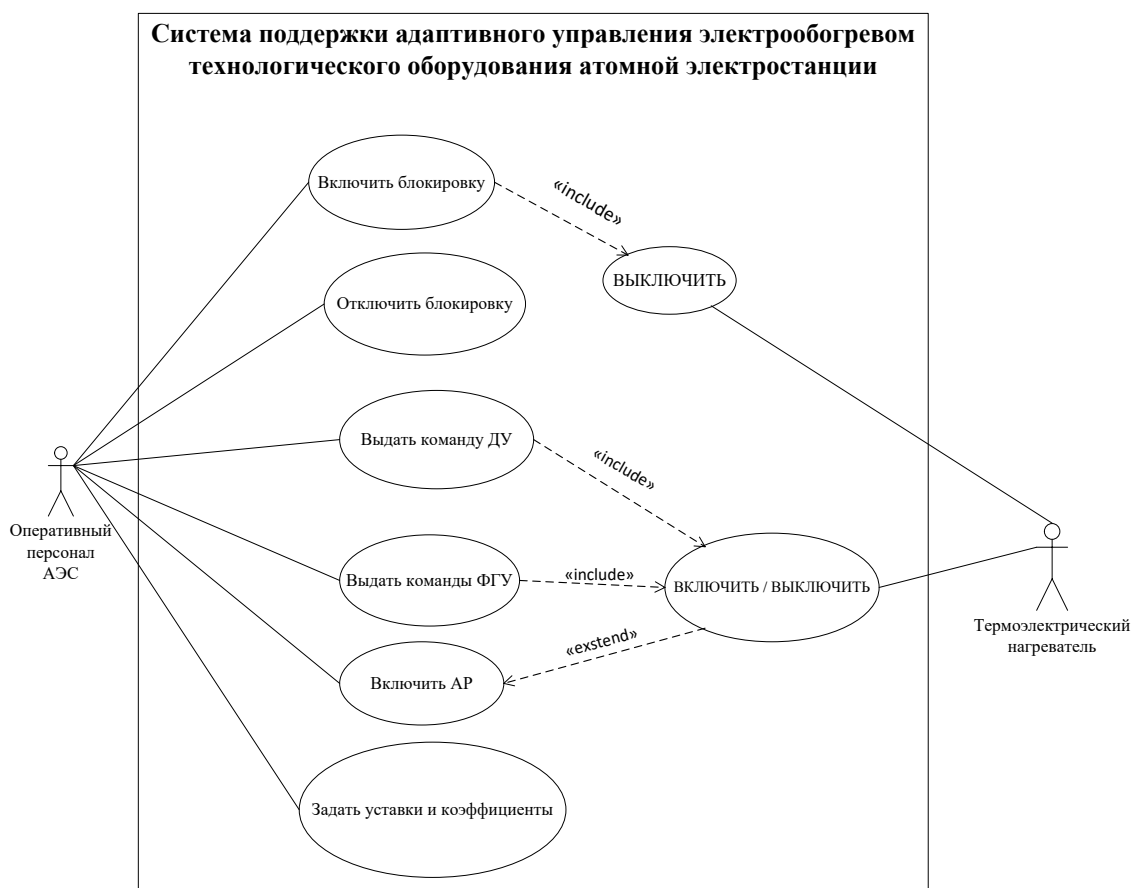


Рисунок 5 – Диаграмма вариантов использования системы адаптивного управления

При контроле достоверности проверяются следующие параметры:

- диапазон;
- скорость изменения.

Задание уставок и коэффициентов предполагает установку:

- минимального значения сигнала;
- максимального значения сигнала;
- скорости изменения сигнала за определенный период времени;
- коэффициента сглаживания для значения температуры.

Функция технологической защиты должна срабатывать при наличии хотя бы одного из нижеперечисленных дискретных сигналов от исполнительного механизма (нагревателя):

- обрыв цепи нагревателя;
- повреждение изоляции.

При наличии этих сигналов должна быть выдана на исполнительный механизм команда «выключить» и отправлено донесение на верхний уровень, что включен режим технологической защиты.

Режим блокировки оборудования срабатывает при наличии следующих причин:

- блок в тестовом положении;
- выключатель автоматический отключен;
- полученное значение температуры недостоверно;
- самопроизвольное изменение состояния «включен», «выключен»;
- получена команда оперативного персонала на включение блокировки.

Процедура блокировки предполагает выдачу команды «выключить» нагревателю и отправку донесения на верхний уровень, что работает режим блокировки.

Режим блокировки может быть отключен по команде оперативного персонала верхнего уровня.

Режим дистанционного управления включает в себя следующие действия:

- прием команды «включить» или «выключить» от оперативного персонала верхнего уровня;
- проверка условий реализации команды;
- формирование и передача команды на исполнительный механизм;
- контроль выполнения команды;
- передача донесения о выполнении команды на верхний уровень.

Функционально-групповое управление (ФГУ) осуществляется в интерактивном режиме при участии оперативного персонала и включает в себя следующие действия:

- выдача по шагам последовательности команд управления до команды последнего шага;

- контроль результатов выполнения последовательности команд на каждом шаге по состоянию управляемого оборудования;
- переключение оперативным персоналом ветвей исполнения пошаговой программы или группы условий;
- выдача донесений о ходе выполнения и завершения программы.

Режим автоматического регулирования (АР) обеспечивает поддержание температуры нагреваемого оборудования в заданных пределах (датчик температуры) с помощью выдачи команд «включить», «выключить» на исполнительный механизм.

Режим АР предполагает прекращение воздействия на исполнительный механизм при отказах, ведущих к нарушению нормальной работы, с выдачей сообщения на верхний уровень о необходимости перехода в режим дистанционного управления.

Взаимодействие с верхним уровнем осуществляется:

- при приеме алгоритмом команд ДУ и ФГУ;
- при приеме алгоритмом уставок и коэффициентов;
- при передаче алгоритмом значений температуры (датчика и задатчика);
- при передаче алгоритмом донесений о выполнении команд;
- при передаче алгоритмом диагностической информации.

1.6.2 Нефункциональные требования

Программное обеспечение должно быть создано в соответствии с принципами объектно-ориентированного программирования на основе свободно распространяемых инструментальных средств с открытыми исходными кодами и функционировать под управлением операционной системы жесткого реального времени.

Алгоритм адаптивного управления должен обеспечивать выполнение заданных функций в полном объеме с установленным уровнем качества и надежности по ГОСТ 25804.6-83 [19].

Оперативный и обслуживающий персонал АЭС должен иметь доступ только к тем данным, которые необходимы им для выполнения своих производственных обязанностей.

1.7 Выбор технологии разработки

Технология разработки программного обеспечения – комплекс процессов и методов по созданию экономичного программного продукта с заданными характеристиками качества [20].

Существуют несколько наиболее известных технологий разработки программного обеспечения: Datarun, Microsoft Solution Framework (MSF), Oracle Method, Rational Unified Process (RUP), SADT (IDEFx).

В качестве технологии разработки программного обеспечения была выбрана методология, в которой инструментально поддерживаются все этапы жизненного цикла разработки программного обеспечения, – RUP (Rational Unified Process). Методология RUP применяет проверенные практикой методы анализа, проектирования, разработки программного обеспечения, а также методы управления проектами. Данная технология разработки гарантирует прозрачность в процессе управления и дает возможность создавать программное обеспечение в соответствии с актуальными требованиями заказчика. Основой технологии разработки RUP является «пошаговый подход», который определяет этапы жизненного цикла, контрольные точки, правила работ для каждого этапа, тем самым упорядочивая процессы проектирования и разработки программного обеспечения. При данном подходе объем работ по каждому процессу изменяется в течение всего жизненного цикла. Каждый цикл итерации проекта начинается с планирования того, что должно быть выполнено. Заканчивается цикл оценкой того, что было сделано и какие цели были достигнуты [21].

RUP предполагает использование визуальных средств для анализа и проектирования и гарантирует, что все участники проекта используют общий язык моделирования, имеют согласованное видение того, как создавать

программный продукт. В качестве языка моделирования используется UML (Unified Modeling Language), являющийся международным стандартом. Модель предметной области представляется в виде диаграммы классов, модель вариантов использования – при помощи диаграммы вариантов использования (прецедентов), взаимодействие компонентов системы между собой описывается диаграммой последовательности [22].

1.8 Выбор инструментальных средств разработки

Инструментальные средства разработки – это совокупность программ и программных комплексов, предназначенных для разработки, отладки и внедрения программных продуктов.

В качестве инструментального средства разработки будем использовать среду Code::Blocks. Code::Blocks – свободная кроссплатформенная среда разработки программного обеспечения. Она имеет открытую архитектуру и может масштабироваться за счет подключаемых модулей. К основным плюсам данной среды относятся: встроенный отладчик, браузер классов, поддержка параллельной сборки, кроссплатформенность, полная поддержка точек останова, интегрированная интеллектуальная подсветка синтаксиса, автодополнение кода. Главным достоинством все же является то, что это открытое программное обеспечение с возможностью дополнения и настройки его разработчиками для своих потребностей [23].

Отладку разработанного программного обеспечения будем проводить с использованием SCADA-системы. SCADA (Supervisory Control And Data Acquisition) – программный пакет, применяемый для разработки и обеспечения работы в режиме реального времени систем сбора, обработки, отображения, архивирования информации об объекте мониторинга или управления [24]. SCADA-системы позволяют решать следующие задачи: обмен данными с промышленными контроллерами и платами в реальном времени, сбор и обработка данных по соответствующим протоколам, визуализация технологического процесса, логическое управление, аварийная

сигнализация, поддержка и ведение баз данных реального времени, генерация отчетов о ходе выполнения технологического процесса.

Ключевым фактором в автоматизированной системе управления технологическим процессом является работа в режиме реального времени. В технологическом процессе негарантированное время реакции на событие недопустимо.

Выводы

Проведенное исследование математических моделей технологических процессов показывает необходимость создания математической модели электрообогрева технологического оборудования атомной электростанции, которая учитывает все существующие режимы управления и позволяет вовремя предотвратить аварийные ситуации. Использование блочного принципа при создании модели позволит выделить основные режимы управления электрообогревом и тем самым облегчить процесс моделирования.

Для управления электрообогревом технологического оборудования наиболее эффективным является метод адаптивного управления, под которым будем понимать автоматическое изменение режима работы алгоритма в зависимости от изменения параметров объекта управления или действий на объект со стороны эксплуатирующего персонала.

Проведенное исследование алгоритмов управления показывает необходимость разработки базового алгоритма поддержки адаптивного управления с возможностью его применения для различных технологических систем в зависимости от изменения параметров, коэффициентов и физических величин. Базовый алгоритм предполагается модифицировать для поддержки адаптивного управления электрообогревом технологического оборудования. Исследованы алгоритмы на основе ПИД-регулирования, алгоритмы на основе самонастройки, алгоритмы на основе нечеткой логики и нейронных сетей. Выбор был сделан в пользу алгоритма ПИД-регулирования, главными преимуществами которого являются:

простота понимания и реализации, как для самих разработчиков алгоритма, так и для операторов АЭС; надежность; хорошее качество управления.

Использование методологии RUP в качестве технологии разработки и инструментальных средств разработки – среды Code::Blocks и SCADA-системы позволит гарантировать, что все участники проекта используют общий язык моделирования UML, свободные инструментальные средства для разработки и отладки программных продуктов.

2 Математическая модель и алгоритм поддержки адаптивного управления электрообогревом

2.1 Математическая модель управления электрообогревом технологического оборудования атомной электростанции

Математическая модель управления электрообогревом технологического оборудования атомной электростанции описывается совокупностью уравнений, которые определяют управляющее воздействие на технологический объект – нагреватель [25]. Следовательно, в качестве входных переменных будут использоваться дискретные сигналы о состоянии нагревателя (D). Регулирование температуры осуществляется на основе значений датчика и задатчика, которые также относятся к входным технологическим переменным (T). Среди входных переменных следует выделить команды управления, выданные оператором верхнего уровня (V). Немаловажное значение имеют коэффициенты, которые заданы в алгоритме и могут быть изменены оператором верхнего уровня (K). В качестве выходного сигнала математической модели выступает управляющее воздействие на исполнительный механизм (U): команда «включить» или команда «выключить». Обобщенная схема математической модели управления электрообогревом, разработанной на основе блочного принципа, представлена на рисунке 6.



Рисунок 6 – Обобщенная схема математической модели

Рассмотрим каждое входное и выходное множество сигналов более подробно. Множество входных дискретных сигналов о состоянии нагревателя D включает в себя:

- обрыв цепи нагревателя d_1 ;
- повреждение изоляции d_2 ;
- блок в тестовом положении d_3 ;
- выключатель автоматический отключен d_4 .

Множество D может быть записано в виде

$$D = \{d_1, d_2, d_3, d_4\}.$$

Множество входных технологических переменных T включает в себя:

- температуру датчика tc ;
- температуру задатчика ts .

Множество T может быть записано в виде

$$T = \{tc, ts\}.$$

Множество команд управления V , выданных оператором верхнего уровня включает в себя:

- команду дистанционного управления «включить» on ;
- команду дистанционного управления «выключить» off ;
- команду функционально-группового управления «включить» fn ;
- команду функционально-группового управления «выключить» fo ;
- команду управления блокировкой vu .

Множество V может быть представлено в виде

$$V = \{on, off, fn, fo, vu\}.$$

Множество коэффициентов K включает в себя коэффициент пропорциональной составляющей, коэффициент интегральной составляющей, коэффициент сглаживания и другие. Множество K может быть представлено в виде

$$K = \{k_1, k_2, k_3, k_4, k_5, \dots, k_n\}.$$

Множество выходных сигналов управления на исполнительный механизм U включает в себя команду «включить» нагреватель (1) и команду «выключить» нагреватель (0). Множество U может быть представлено в виде

$$U = \{0, 1\}.$$

Используя блочный принцип, математическую модель управления электрообогревом технологического оборудования атомной электростанции можно представить в виде совокупности уравнений

$$\begin{cases} U = f(TP, L, DU, FGU, PI) \\ TP = f(d_1, d_2) \\ L = f(d_3, d_4, tc, ts, vu) \\ DU = f(on, off) \\ FGU = f(fn, fo) \\ PI = f(tc, ts, K), \end{cases}$$

где TP – функция, отвечающая за технологическую защиту;

L – функция, отвечающая за блокировку;

DU – функция, отвечающая за дистанционное управление;

FGU – функция, отвечающая за функционально-групповое управление;

PI – функция автоматического регулирования температуры.

Рассмотрим каждую функцию по отдельности. Функция технологической защиты

$$TP = \begin{cases} 1, & \text{если } d_1 = 1 \\ 1, & \text{если } d_2 = 1 \\ 0, & \text{если } d_1 = d_2 = 0. \end{cases}$$

Функция TP проверяет, приходят ли от нагревателя дискретные сигналы обрыв цепи нагревателя (d_1) и повреждение изоляции (d_2). Если эти сигналы установлены в 1, то включается режим технологической защиты. Основное назначение технологической защиты – защита от аварий действующего технологического оборудования. Режим технологической защиты имеет наивысший приоритет, если имеются условия для его выполнения, никакие больше условия не проверяются.

Функция блокировки

$$L = \begin{cases} 1, & \text{если } d_3 = 1 \\ 1, & \text{если } d_4 = 1 \\ 1, & \text{если } tc = 32767 \\ 1, & \text{если } ts = 32767 \\ 1, & \text{если } vu = 2 \\ 0, & \text{если } vu = 1 \\ 0, & \text{если } d_3 = d_4 = 0. \end{cases}$$

Вторым по приоритету является режим блокировки L , который срабатывает, если:

- блок в тестовом положении $d_3 = 1$;
- выключатель автоматический отключен $d_4 = 1$;
- недостоверное значение датчика температуры ($tc = 32767$);
- недостоверное значение задатчика температуры ($ts = 32767$);
- от оператора верхнего уровня пришла команда на включение блокировки ($vu = 2$).

Режим блокировки отключается, если есть команда на отключение блокировки с верхнего уровня ($vu = 1$). Если имеются условия для работы режима блокировки, никакие больше условия не проверяются.

Функция дистанционного управления

$$DU = \begin{cases} 0, & \text{если } off = 1, on = 0 \\ 1, & \text{если } on = 1, off = 1. \end{cases}$$

Третий по приоритету – режим дистанционного управления DU , который срабатывает, если оператор с верхнего уровня выдал команду «выключить» нагреватель ($off = 1$) или «включить» нагреватель ($on = 1$).

Функция функционально-группового управления

$$FGU = \begin{cases} 0, & \text{если } fo = 1, fn = 0 \\ 1, & \text{если } fn = 1, fo = 0. \end{cases}$$

Режим функционально-группового управления FGU срабатывает, если с верхнего уровня оператором выданы команды «выключить» ($fo = 1$) или «включить» ($fn = 1$).

Функция автоматического регулирования температуры

$$PI = \begin{cases} 0, & \text{если } f(tc, ts, K) < 0 \\ 1, & \text{если } f(tc, ts, K) > 0. \end{cases}$$

Режим автоматического регулирования PI работает, если условия для работы других режимов не выполняются. Функция $f(tc, ts, K)$ отвечает за ПИ-регулирование температуры

$$f(tc, ts, K) = k_1 |tc - ts|(t) + k_2 \int_0^{\tau} |tc - ts|(t) dt,$$

где k_1 – коэффициент пропорциональной составляющей;

k_2 – коэффициент интегральной составляющей;

t – время.

В формуле U может принимать значения

$$U = \begin{cases} 0, & \text{если } TP = 1 \\ 0, & \text{если } L = 1 \\ 0, & \text{если } DU = 0 \\ 1, & \text{если } DU = 1 \\ 0, & \text{если } FGU = 0 \\ 1, & \text{если } FGU = 1 \\ 0, & \text{если } PI < 0 \\ 1, & \text{если } PI > 0. \end{cases}$$

Условия следует проверять в порядке, указанном в формуле.

Если $U = 0$, это соответствует выдаче команды «выключить» нагреватель, если $U = 1$ – «включить» нагреватель.

Математическая модель управления электрообогревом технологического оборудования атомной электростанции, разработанная на основе блочного принципа, в отличие от известных моделей, учитывает все существующие режимы управления электрообогревом. Она служит основой как для разработки алгоритма адаптивного управления, так и для имитационного моделирования процесса электрообогрева, необходимого для проведения тестирования разработанного алгоритма управления.

2.2 Разработка алгоритма поддержки адаптивного управления электрообогревом

На основе математической модели создадим базовый алгоритм поддержки адаптивного управления, который может быть использован в таких технологических системах, как система управления регулирующей арматурой, запорной арматурой, насосом, электрообогревом, электродвигателем и клапанами питательного узла парогенератора. Алгоритм можно представить в виде последовательность команд. Сначала выполняется инициализация (задание значений необходимых параметров и коэффициентов). Затем определяется время работы цикла, в нашем случае – 100 мс. Далее необходимо запросить текущее значение технологических переменных и состояния исполнительного механизма. Потом определить приоритет режимов работы исполнительного механизма. В нашем случае иерархия выглядит следующим образом: технологическая защита, блокировка, дистанционное управление, функционально-групповое управление, автоматическое регулирование. В соответствии с приоритетом, если условия срабатывания режима выполняются, то он работает, иначе – проверяются условия срабатывания для других режимов. Режим работает, пока есть соответствующие условия. На рисунке 7 представлена диаграмма деятельности базового алгоритма поддержки адаптивного управления.

На основе базового алгоритма создадим алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции. В качестве технологических переменных в нашем алгоритме используется датчик и задатчик температуры. От входного канала мы получаем значение сигнала в миллиамперах, которое необходимо масштабировать и фильтровать. В качестве исполнительного механизма выступает нагреватель и его состояния: включен, выключен, обрыв цепи нагревателя, повреждение изоляции, блок в тестовом положении и выключатель автоматический отключен.

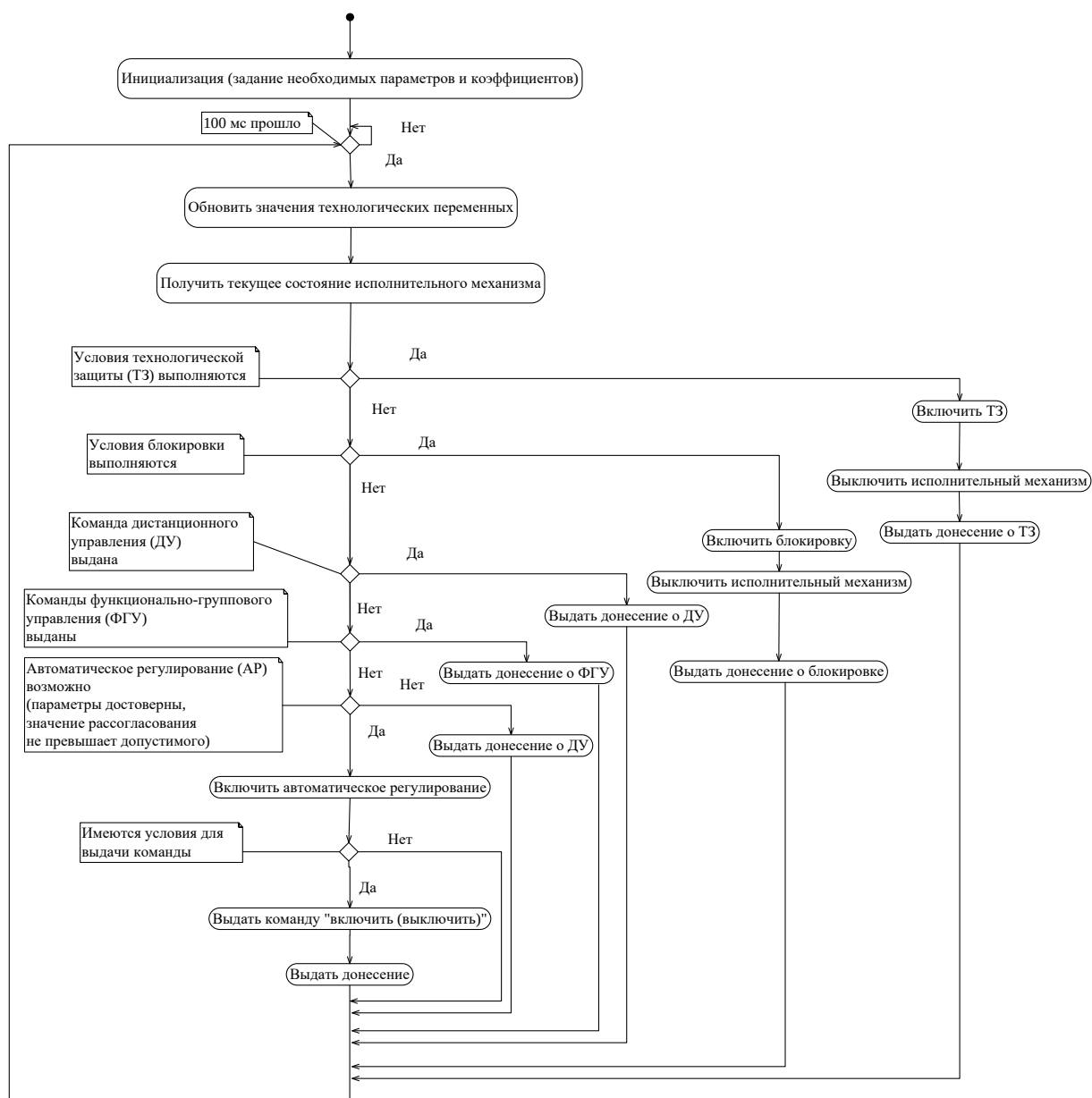


Рисунок 7 – Базовый алгоритм поддержки адаптивного управления

Условия включения режима технологической защиты: обрыв цепи нагревателя, повреждение изоляции. При срабатывании условий включается режим ТЗ, выключается нагреватель, передается донесение о включение режима на верхний уровень и выполнение текущей итерации цикла завершается. Условия включения режима блокировки: блок в тестовом

положении, выключатель автоматический отключен, получена команда на включение блокировки с верхнего уровня, недостоверное значение технологических переменных. При срабатывании условий включается режим блокировки, выключается нагреватель, передается донесение о включение режима на верхний уровень и выполнение текущей итерации цикла завершается. Блокировку можно выключить, если с верхнего уровня поступит соответствующая команда. Режим дистанционного управления активизируется, если с верхнего уровня получена команда управления (включить или выключить). Команда выполняется, отправляется донесение на верхний уровень и выполнение текущей итерации цикла завершается. Режим функционально-группового управления работает, если с верхнего уровня получена последовательность команд управления (включить, выключить). Команды выполняются, отправляется донесение на верхний уровень и выполнение текущей итерации цикла завершается. Режим авторегулирования функционирует, если предыдущие режимы не активны. Если в соответствии с алгоритмом ПИ-регулятора необходимо выдать команду, то команда (включить, выключить) выдается, отправляется донесение на верхний уровень и выполнение текущей итерации цикла завершается. Если разница между датчиком и задатчиком слишком большая, то алгоритм переходит в режим дистанционного управления, отправляется донесение на верхний уровень и выполнение текущей итерации цикла завершается [26]. Каждый режим работает, пока есть соответствующие условия. Цикл работает бесконечно с периодом 100 мс. На рисунке 8 представлен в виде диаграммы деятельности алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции.

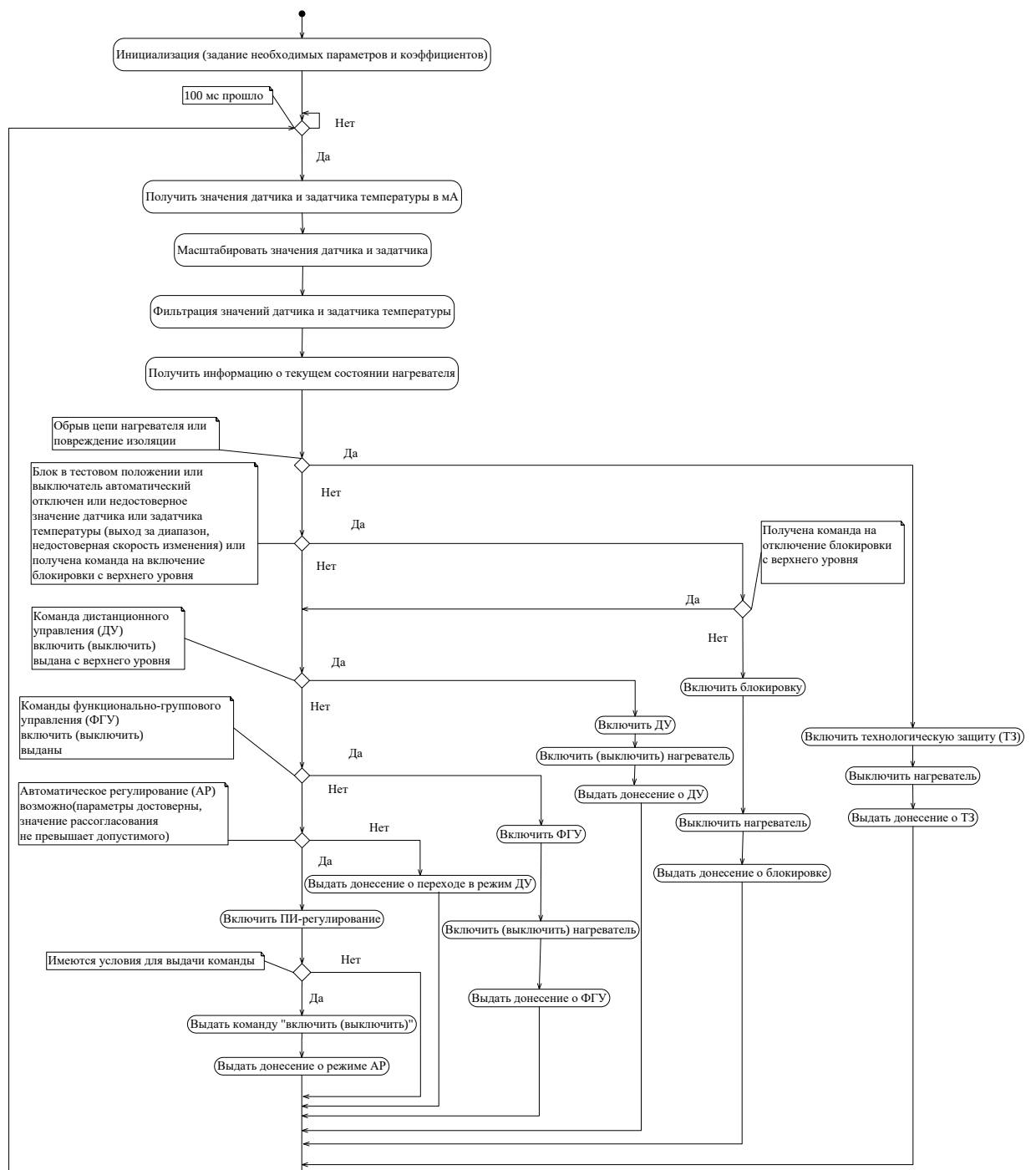


Рисунок 8 – Алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции

На рисунке 9 представлена диаграмма последовательности для варианта использования «Включить блокировку», этот вариант справедлив для ситуации, когда нет условий для работы режима технологической защиты. Режим блокировки может быть включен как по команде оператора, так и автоматически при выполнении определенных условий.

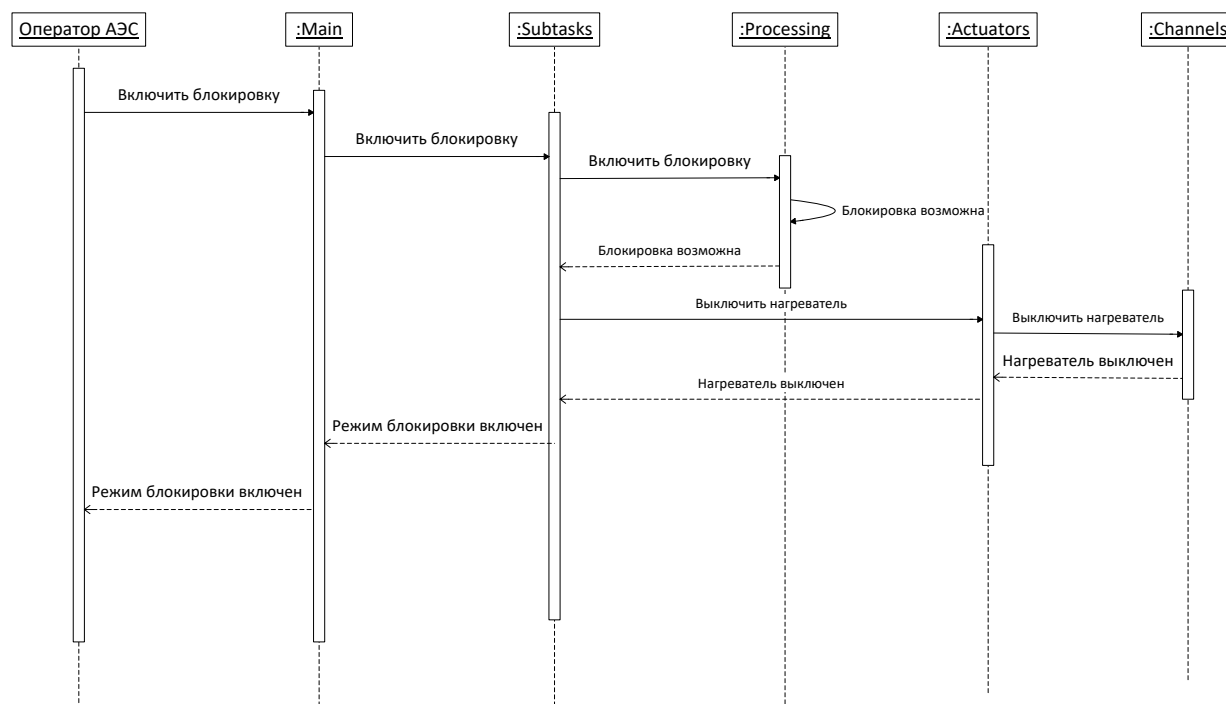


Рисунок 9 – Диаграмма последовательности для варианта использования «Включить блокировку» оператором АЭС

На рисунке 10 представлена диаграмма последовательности для варианта использования «Выдать команду ДУ «включить». Дистанционное управление возможно, если нет условий для работы режима технологической защиты, и не работает блокировка. На рисунке 11 представлена диаграмма последовательности для варианта использования «Включить АР».

Автоматическое регулирование может быть включено как автоматически в алгоритме, так и принудительно оператором АЭС.

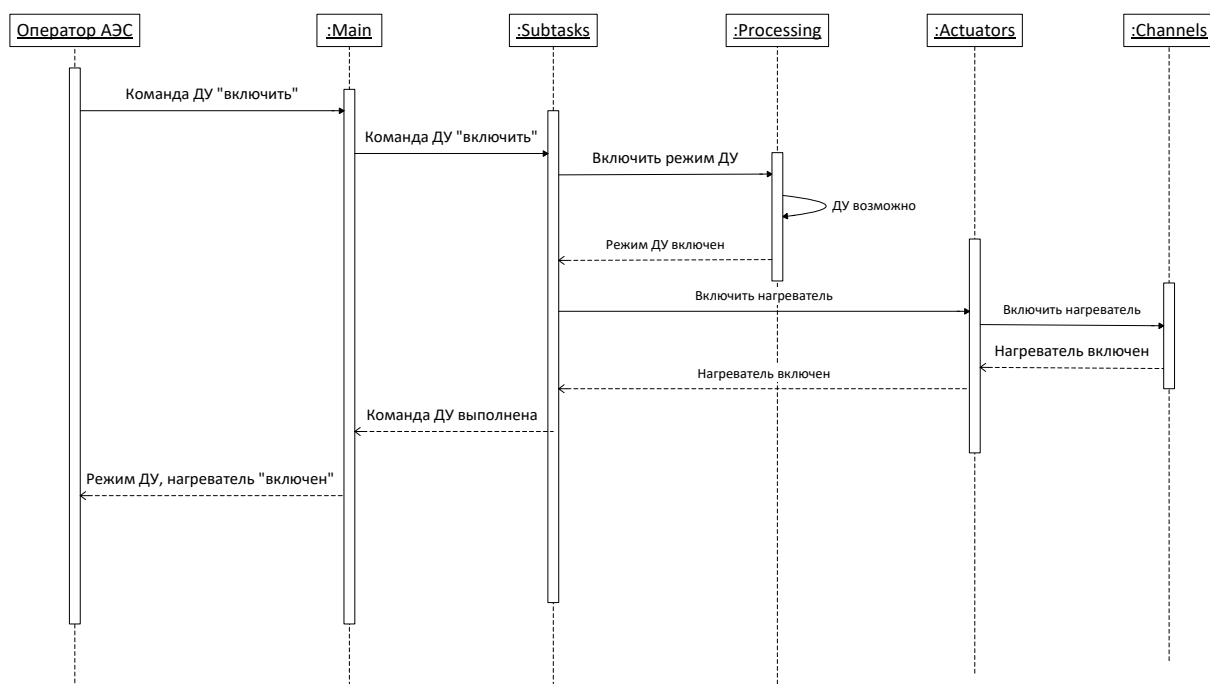


Рисунок 10 – Диаграмма последовательности для варианта использования
«Выдать команду ДУ «включить» оператором АЭС

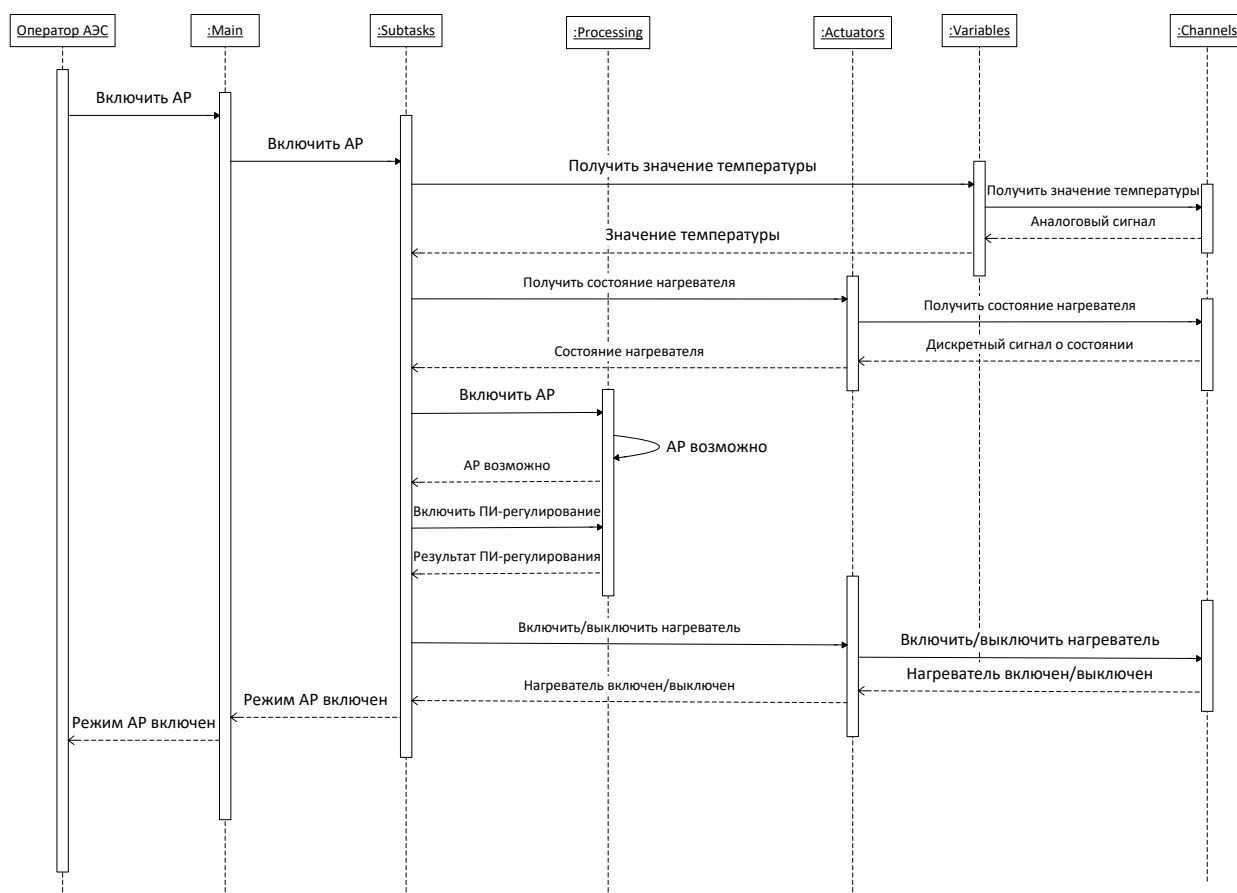


Рисунок 11 – Диаграмма последовательности для варианта использования
«Включить AP» оператором АЭС

Каждый входной сигнал, представляющий температуру, подвергается трем процедурам: масштабированию, фильтрации и контролю достоверности.

Масштабирование предполагает перевод температуры из миллиампер в градусы Цельсия. Пересчет осуществляется по формуле:

$$T = T_L + \frac{(T_H - T_L)(I - I_L)}{I_H - I_L},$$

где T – текущая температура;

T_L – нижняя граница диапазона температуры;

T_H – верхняя граница диапазона температуры;

I_H – верхняя граница диапазона тока;

I_L – нижняя граница диапазона тока;

I – измеренный ток.

Фильтрация предполагает выделение желательного электрического сигнала на фоне помех. Процедура фильтрации осуществляется по формуле:

$$T = T_M K_S + T_P (1 - K_S),$$

где T – температура (сглаженное значение);

T_M – температура (измеренное значение);

K_S – коэффициент сглаживания (диапазон значений от 0 до 1);

T_P – температура (предыдущее значение).

Критериями контроля достоверности являются диапазон значений параметра и допустимая скорость его изменения.

Алгоритм автоматического регулирования функционирует на основе ПИ-регулирования на базе двух значений: датчика и задатчика. Датчик – это измеряемое значение температуры. Задатчик – устанавливаемое значение (уставка) температуры.

Запишем формулу ПИ-регулирования в виде

$$u(t) = P + I = K_p e(t) + K_i \int_0^t e(t) dt,$$

где $u(t)$ – управляющее воздействие;

P – пропорциональная составляющая;

I – интегральная составляющая;

$e(t)$ – рассогласование (разница между датчиком и задатчиком);

K_p – пропорциональный коэффициент;

K_i – интегральный коэффициент.

Данная формула сложна для расчетов в программе, поэтому в программной реализации необходимо перейти к дискретной форме:

$$u(t) = P(t) + I(t),$$

$$P(t) = K_p e(t),$$

$$I(t) = I(t-1) + K_i e(t).$$

Рассчитывая пропорциональную и интегральную составляющую, а затем, складывая эти значения, мы получаем управляющее воздействие ПИ-регулятора.

Выводы

Математическая модель управления электрообогревом технологического оборудования атомной электростанции, разработанная на основе блочного принципа, учитывает все существующие режимы управления электрообогревом. Она служит основой как для разработки алгоритма адаптивного управления, так и для имитационного моделирования процесса электрообогрева, необходимого для проведения тестирования разработанного алгоритма управления.

Разработанный базовый алгоритм поддержки адаптивного управления может быть использован для разных технологических процессов в зависимости от изменения параметров, коэффициентов и физических величин.

Алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции предназначен для автоматизированного управления электрическим нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима работы.

3 Реализация модели и алгоритма поддержки адаптивного управления и оценка их эффективности

3.1 Техническая реализация

Схема подключения модуля автоматизации индивидуального управления к технологическому модулю индивидуального управления представлена на рисунке 12.



Рисунок 12 – Схема подключения

Модуль автоматизации индивидуального управления предназначен для автоматизированного управления трубчатым электрическим нагревателем, погруженным в резервуар, на основе данных о температуре технологической среды, транспортируемой по трубопроводу из резервуара. Модуль индивидуального управления имеет шесть каналов дискретного ввода для приема информации о текущем состоянии электрического нагревателя: состояние «включен», состояние «выключен», обрыв цепи нагревателя, блок в тестовом положении, выключатель автоматический отключен, повреждение

изоляции. Также в модуле присутствуют два канала дискретного вывода: команда «включить» и команда «выключить», – для выдачи управляющих команд и два канала аналогового ввода 4 – 20 мА для приема сигналов от датчика и задатчика температуры.

3.2 Архитектура базовых классов предметной области

В результате анализа предметной области были выделены базовые классы: технологическая переменная, исполнительный механизм, канал, артефакт обработки и управления, подзадача.

Класс технологическая переменная (ТП) описывает параметры состояния технологического объекта управления. Наследником данного класса, в нашем случае, является температура. Каждая ТП соотносится с датчиком, который измеряет ее величину в некоторой контрольной точке.

Класс исполнительный механизм (ИМ) представляет собой техническое устройство, которое способно изменить текущее состояние технологического объекта управления. Наследником этого класса является нагреватель. Проверить его работу можно по температуре, которая приходит с датчика.

Класс канал относится к аппаратным средствам автоматизации. Он представляет собой часть модуль автоматизации и отвечает за прием или выдачу сигнала технологическому объекту управления. Каналы подразделяются на входные и выходные.

Класс артефакт обработки и управления (АОУ) представляет часть задачи, которая реализует элементарный алгоритм [27].

Класс подзадача является центральным управляющим классом, в нем запускается адаптивный алгоритм путем вызова экземпляров других классов.

Диаграмма взаимодействия базовых классов предметной области представлена на рисунке 13.

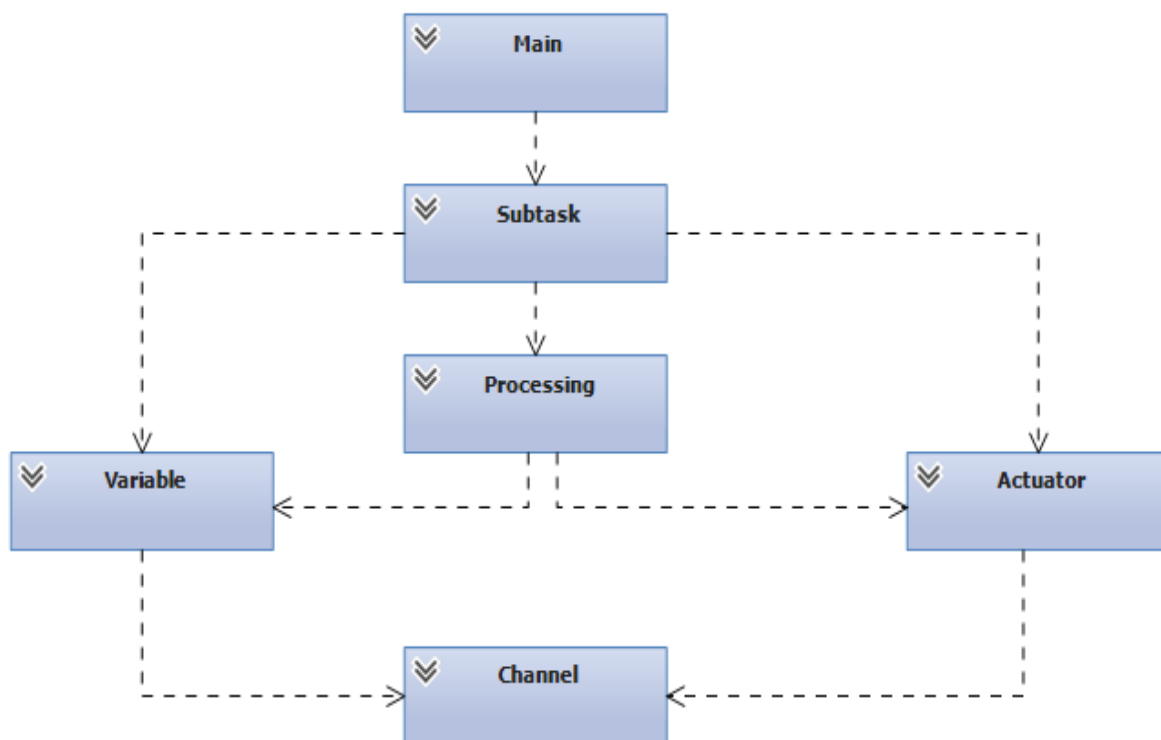


Рисунок 13 – Диаграмма взаимодействия базовых классов предметной области

На диаграмме класс технологическая переменная – Variable, исполнительный механизм – Actuator, канал – Channel, артефакт обработки и управления – Processing, подзадача – Subtask. Main – главный класс.

В главной функции класса Main вызывается объект класса подзадача – Subtask, который обращается за текущими значениями к объектам класса Variable. Затем вызываются в соответствии с приоритетом объекты класса Processing. Эти объекты работают с экземплярами классов Actuator и Variable, которые получают значения через экземпляры класса Channel. На основе анализа полученных данных объект Subtask воздействует на соответствующий экземпляр класса Actuator.

3.3 Программная реализация

Программная реализация модели и алгоритма адаптивного управления электрообогревом технологического оборудования выполнялась на языке программирования C++ в среде CodeBlocks. В результате были созданы следующие классы:

- Main – главный класс программы;
- Actuator – исполнительный механизм;
- Heater – наследник класса Actuator, нагреватель;
- Channel – канал;
- Status_channel – наследник класса Channel, канал статуса;
- Output_channel – наследник класса Channel, канал вывода;
- Parameter_channel – наследник класса Channel, канал параметра;
- Input_channel – наследник класса Channel, канал ввода;
- Control – вспомогательный класс для организации процесса управления;
- MA_control_NA – наследник класса Control, класс для запуска алгоритма адаптивного управления электрообогревом;
- VU_control_NA – наследник класса Control, класс для передачи информации на верхний уровень;
- Memory – класс для организации памяти;
- Modbus – класс для организации взаимодействия по протоколу Modbus;
- Processing – артефакты обработки и управления;
- Tech_protection – наследник класса Processing, класс технологическая защита;
- Lock – наследник класса Processing, класс блокировка;
- Remote_control – наследник класса Processing, класс дистанционного управления;

- Fgroup_control – наследник класса Processing, класс функционально-группового управления;
- Auto_control – наследник класса Processing, класс авторегулирования;
- PI_control – наследник класса Auto_control, класс ПИ-регулирования;
- Subtask – подзадача;
- Control_heater – наследник класса Subtask, подзадача поддержки адаптивного управления электрообогревом;
- Variable – технологическая переменная;
- Temperature – наследник класса Variable, температура.

Диаграмма классов системы поддержки адаптивного управления электрообогревом представлена на рисунке 14.

Рассмотрим более подробно классы PI_control и Control_heater.

Класс PI_control имеет атрибуты:

- k_p – коэффициент пропорциональной составляющей;
- k_i – коэффициент интегральной составляющей;
- current_value – предыдущее значение датчика;
- set_value – предыдущее значение задатчика;
- e_curr – предыдущее рассогласование;
- p_comp – пропорциональная составляющая;
- i_comp – интегральная составляющая;
- dtt – допустимая разница между датчиком и задатчиком, при котором работает авторегулирование;
- gs_na – гистерезис для температуры.

Класс PI_control содержит методы:

- tuning – метод задания коэффициентов ПИ-регулятора;
- control_PI – метод ПИ-регулирования.

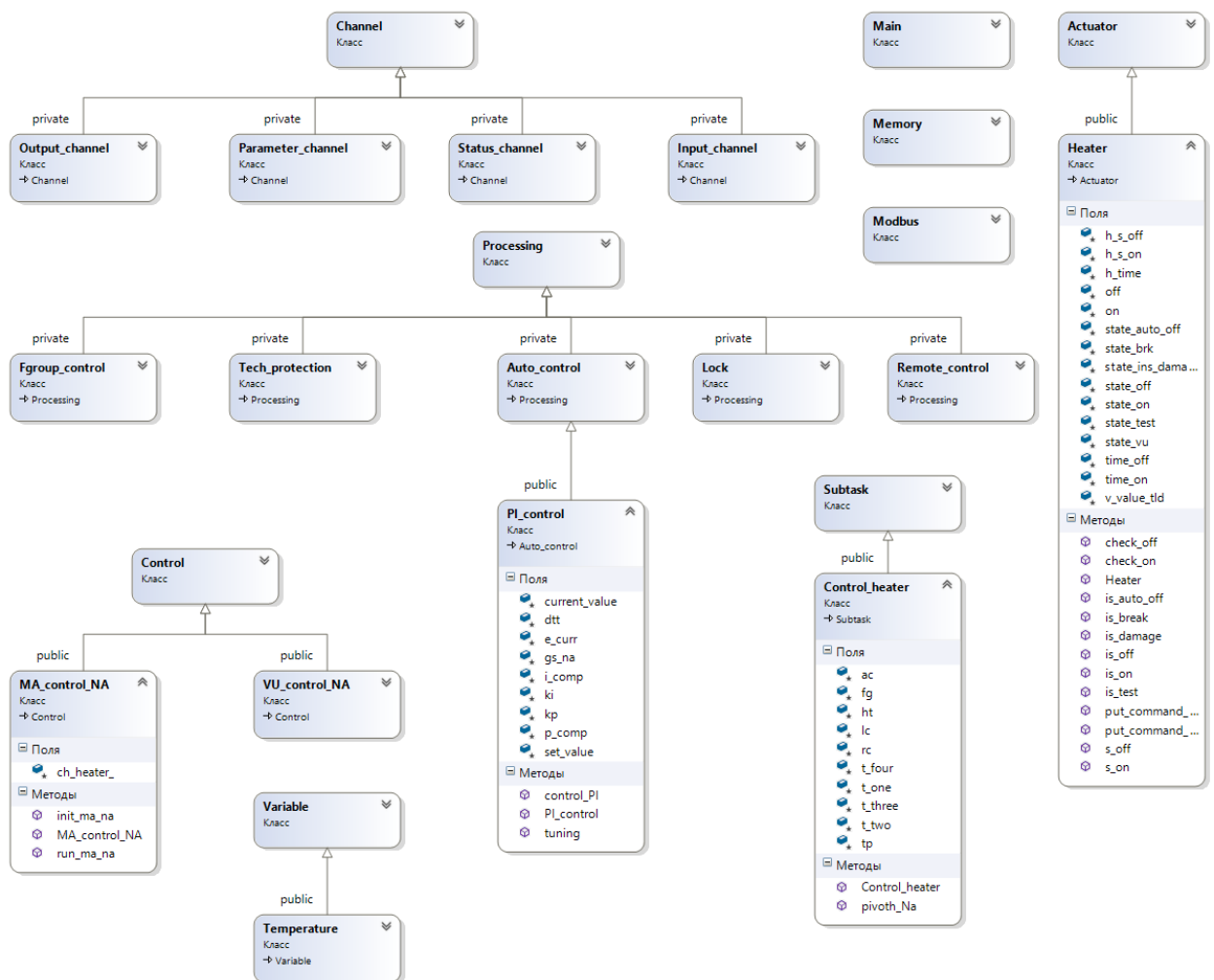


Рисунок 14 – Диаграмма классов системы поддержки адаптивного управления электрообогревом

Класс Control_heater имеет атрибуты:

- ac – объект класса ПИ-регулирования;
- fg – объект класса функционально-группового управления;
- ht – объект класса нагреватель;
- lc – объект класса блокировка;
- rc – объект класса дистанционное управление;
- t_one – объект класса температура (резерв датчик);
- t_two – объект класса температура (резерв задатчик);
- t_three – объект класса температура (датчик);
- t_four – объект класса температура (задатчик);
- tp – объект класса технологическая защита.

Класс `Control_heater` имеет метод:

- `pivoth_Na` – алгоритм поддержки адаптивного управления электрообогревом технологического оборудования.

Диаграмма компонентов системы поддержки адаптивного управления электрообогревом представлена на рисунке 15.

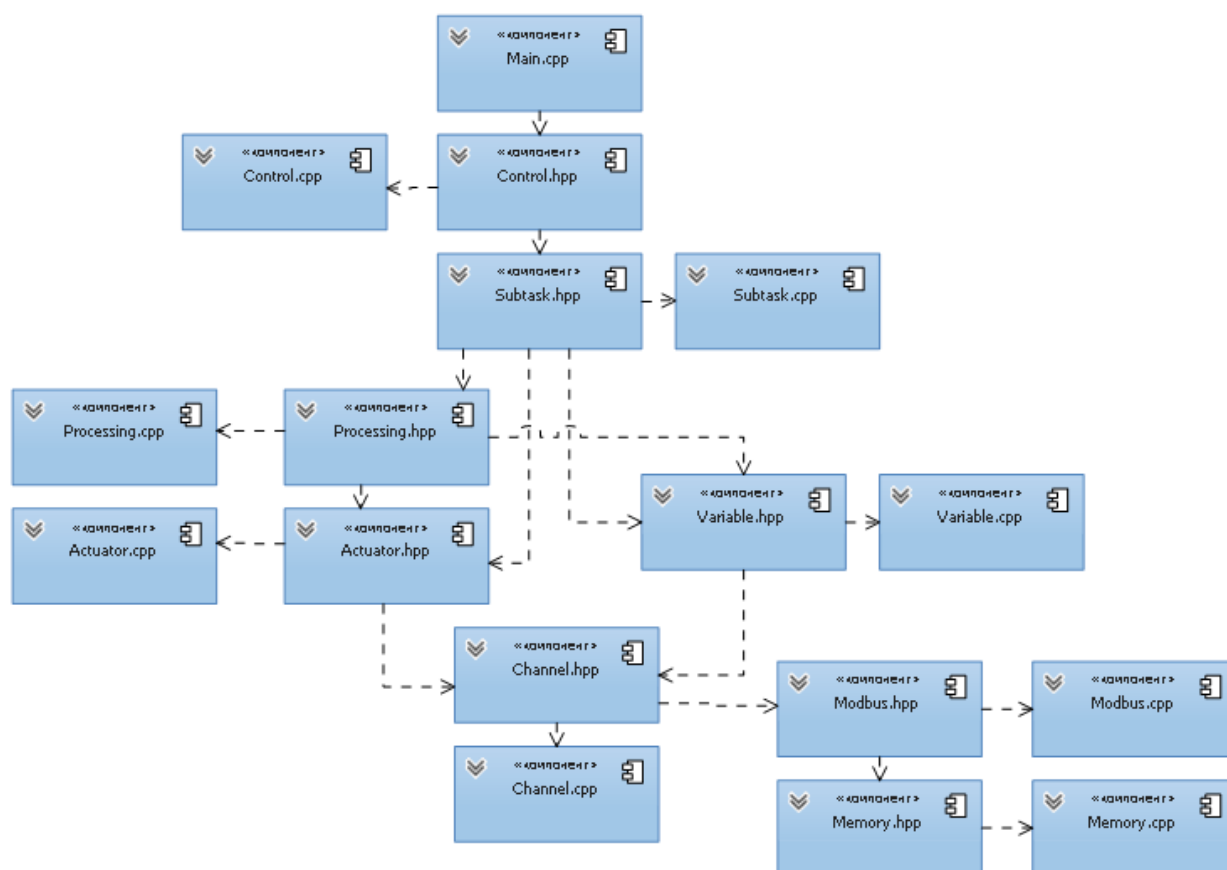


Рисунок 15 – Диаграмма компонентов системы поддержки адаптивного управления электрообогревом

На рисунке представлены следующие компоненты:

- `Main.cpp` – исходный код, отвечающий за запуск алгоритма управления;
- `Control.hpp` – заголовочный файл, отвечающий за инициализацию задачи управления и передачу информации на верхний уровень;

- Control.cpp – исходный код заголовочного файла Control.hpp, реализующий инициализацию задачи управления и передачу информации на верхний уровень;
- Subtask.hpp – заголовочный файл, отвечающий за задачу поддержки адаптивного управления электрообогревом технологического оборудования;
- Subtask.cpp – исходный код заголовочного файла Subtask.hpp, реализующий задачу поддержки адаптивного управления электрообогревом технологического оборудования;
- Processing.hpp – заголовочный файл, отвечающий за алгоритмы артефактов обработки и управления;
- Processing.cpp – исходный код заголовочного файла Processing.hpp, реализующий алгоритмы артефактов обработки и управления;
- Variable.hpp – заголовочный файл, отвечающий за технологические переменные;
- Variable.cpp – исходный код заголовочного файла Variable.hpp, отвечающего за технологические переменные;
- Actuator.hpp – заголовочный файл, отвечающий за исполнительный механизм;
- Actuator.cpp – исходный код заголовочного файла Actuator.hpp, отвечающего за исполнительный механизм;
- Channel.hpp – заголовочный файл, отвечающий за каналы;
- Channel.cpp – исходный код заголовочного файла Channel.hpp, отвечающего за каналы;
- Modbus.hpp – заголовочный файл, отвечающий за протокол взаимодействия;
- Modbus.cpp – исходный код заголовочного файла Modbus.hpp, реализующий протокол взаимодействия;
- Memory.hpp – заголовочный файл, отвечающий за организацию памяти;

- Memory.cpp – исходный код заголовочного файла Memory.hpp, отвечающего за организацию памяти.

Листинг программной реализации модели и алгоритма адаптивного управления электрообогревом технологического оборудования представлен в приложении Б.

3.4 Тестирование модели и алгоритма поддержки адаптивного управления электрообогревом

Тестирование – процесс испытания программного продукта с целью проверки соответствия между реальным и ожидаемым поведением программы на определенном наборе тестов. Под тестированием чаще всего понимают процедуру обнаружения ошибок. Каждый тест должен включать две обязательные компоненты:

- описание входных данных;
- описание точного и корректного результата, который соответствует набору входных данных [28].

По знанию внутреннего строения системы:

- тестирование черного ящика;
- тестирование белого ящика.

Тестирование черного ящика предполагает проверку программы по входу-выходу, при этом тестировщик ничего не знает про внутреннюю структуру и логику программы.

Тестирование белого ящика предполагает знание внутренней структуры тестируемой программы. Составление тестовых комбинаций ведется из расчета прохождения программы по всем возможным маршрутам ее пути передачи управления.

По объекту тестирования различают:

- функциональное тестирование;
- конфигурационное тестирование;
- юзабилити-тестирование;

- тестирование производительности;
- тестирование безопасности.

3.4.1 Функциональное тестирование

Функциональное тестирование – это вид тестирования, основная задача которого состоит в проверке реализации функциональных требований. Иначе говоря, проверяется способность программного продукта решать в определенных условиях поставленные перед ним задачи. Тестирование будем проводить с использованием программного пакета SCADA-системы под операционной системой Debian-10.0. План тестирования модели и алгоритма поддержки адаптивного управления электрообогревом технологического оборудования АЭС представлен в таблице 2.

Таблица 2 – План функционального тестирования в SCADA-системе

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
1	Включение блокировки оператором	Система работает в режиме технологической защиты. С верхнего уровня оператор включает блокировку.	Система продолжает работать в режиме технологической защиты, поскольку приоритет этого режима выше.	Соответствует ожидаемым результатам (рисунок В.1)
2	Включение блокировки оператором	Система работает в режиме автоматического регулирования. С верхнего уровня оператор включает блокировку.	Система автоматически переходит в режим блокировки. На верхний уровень приходит донесение о переходе в режим по команде оператора. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.2)

Продолжение таблицы 2

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
3	Отключение блокировки оператором	Система работает в режиме блокировки. Оператор с верхнего уровня отключает блокировку.	Система автоматически переходит в режим автоматического регулирования. На верхний уровень приходит донесение об отключении блокировки оператором.	Соответствует ожидаемым результатам (рисунок В.3)
4	Выдана команда дистанционного управления «включить»	Система работает в режиме технологической защиты. С верхнего уровня оператор выдает команду ДУ «включить».	Система продолжает работать в режиме технологической защиты, поскольку приоритет этого режима выше, чем у дистанционного управления.	Соответствует ожидаемым результатам (рисунок В.4)
5	Выдана команда дистанционного управления «включить»	Система работает в режиме автоматического регулирования. С верхнего уровня оператор выдает команду ДУ «включить».	Система автоматически переходит в режим дистанционного управления, поскольку приоритет этого режима выше. На верхний уровень приходит донесение о выполнении команды «включить». Выдана команда «включить нагреватель».	Соответствует ожидаемым результатам (рисунок В.5)
6	Установка коэффициента ПИ-регулятора	Система работает в режиме автоматического регулирования. На верхнем уровне оператор устанавливает коэффициент пропорциональной составляющей, равный 1.	Система автоматически меняет в ПИ-регулировании коэффициент и продолжает регулировать с коэффициентом, равным 1.	Соответствует ожидаемым результатам (рисунок В.6)

Результаты тестирования представлены в приложении В. Функциональное тестирование свидетельствует о корректности работы разработанного программного обеспечения на основе созданной модели и алгоритма поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции.

3.4.2 Проведение экспериментов и анализ результатов

Эксперименты будем проводить на модели имитационного процесса электрообогрева в SCADA-системе под операционной системой Debian-10.0. План проведения экспериментов представлен в таблице 3.

Таблица 3 – План проведения экспериментов в SCADA-системе

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
1	Обрыв цепи нагревателя	Система работает в режиме автоматического регулирования. На нижнем уровне имитируем дискретный сигнал «обрыв цепи нагревателя».	Система автоматически переходит в режим технологической защиты. На верхний уровень приходит донесение о причине срабатывания технологической защиты – обрыве цепи нагревателя. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.7)
2	Повреждение изоляции	Система работает в режиме автоматического регулирования. На нижнем уровне имитируем сигнал «повреждение изоляции».	Система автоматически переходит в режим технологической защиты. На верхний уровень приходит донесение о причине срабатывания технологической защиты – повреждении изоляции. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.8)

Продолжение таблицы 3

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
3	Блок в тестовом положении	Система работает в режиме автоматического регулирования. На нижнем уровне имитируем дискретный сигнал «блок в тестовом положении».	Система автоматически переходит в режим блокировки. На верхний уровень приходит донесение о причине срабатывания блокировки – блок в тестовом положении. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.9)
4	Выключатель автоматически отключен	Система работает в режиме автоматического регулирования. На нижнем уровне имитируем дискретный сигнал «выключатель автоматический отключен».	Система автоматически переходит в режим блокировки. На верхний уровень приходит донесение о причине срабатывания блокировки – выключатель автоматический отключен. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.10)
5	Недостоверное значение датчика температуры	Система работает в режиме автоматического регулирования. На нижнем уровне устанавливаем скорость изменения датчика температуры выше допустимой.	Система автоматически переходит в режим блокировки. На верхний уровень приходит донесение о срабатывании блокировки – недостоверное значение датчика температуры. Выдана команда выключить нагреватель.	Соответствует ожидаемым результатам (рисунок В.11)

Окончание таблицы 3

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
6	Датчик температуры меньше задатчика температуры	Система работает в режиме автоматического регулирования. Датчик температуры меньше задатчика температуры $269.5 < 350$.	Система автоматически в соответствии с законом ПИ-регулирования выдает команду «включить». Нагреватель переходит в состояние «включен» (0НАС01НН001_003).	Соответствует ожидаемым результатам (рисунок В.12)
7	Датчик температуры больше задатчика температуры	Система работает в режиме автоматического регулирования. Датчик температуры больше задатчика температуры $303.3 > 300.1$.	Система автоматически в соответствии с законом ПИ-регулирования выдает команду «выключить». Нагреватель переходит в состояние «выключен» (0НАС01НН001_004).	Соответствует ожидаемым результатам (рисунок В.13)
8	Разница между значениями датчика и задатчика температуры больше допустимой	Система работает в режиме автоматического регулирования. Допустимая разница между значениями датчика и задатчика – 250. Текущая разница равна $545 - 268.2 = 276.8$.	Система останавливает автоматическое регулирование с выдачей донесения, что необходимо перейти в режим ДУ, так как слишком большая разница между датчиком и задатчиком, режим автоматического регулирования невозможен.	Соответствует ожидаемым результатам (рисунок В.14)

По результатам тестирования можно судить о эффективности работы модели и алгоритма в экспериментальных условиях. Алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции реализует автоматизированное управление нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима работы.

Программное обеспечение, созданное на основе математической модели и алгоритма поддержки адаптивного управления включено в состав опытного образца программно-технического комплекса для создания и модернизации систем контроля и управления технологическим оборудованием ПТК «УРАЛ-Атом» и активно используется при проведении его комплексной наладки, испытаний и эксплуатации (приложение Г).

Выводы

В ходе разработки была представлена архитектура технических средств и базовых классов предметной области. На основе предложенной математической модели и алгоритма поддержки адаптивного управления электрообогревом было разработано программное обеспечение. Построена диаграмма классов и компонентов разработанного программного обеспечения. Проведено функциональное и экспериментальное тестирование под операционной системой Debian-10.0 в SCADA-системе. По результатам тестирования можно судить о корректности и эффективности работы модели и алгоритма в заданных условиях. Алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции реализует автоматизированное управление электрическим нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима работы.

Заключение

В ходе магистерской диссертации были решены следующие задачи:

- проанализированы известные математические модели и алгоритмы управления технологическими процессами атомной электростанции;
- разработана математическая модель управления электрообогревом технологического оборудования атомной электростанции, которая, в отличие от известных моделей, учитывает все существующие режимы управления электрообогревом;
- разработан базовый алгоритм поддержки адаптивного управления, который, в отличие от известных алгоритмов, может быть использован для разных технологических процессов в зависимости от изменения параметров, коэффициентов и физических величин;
- разработан алгоритм управления электрообогревом технологического оборудования атомной электростанции, который, в отличие от известных, является адаптивным и обеспечивает автоматизированное управление электрическим нагревателем на основе данных о температуре технологической среды с выдачей управляющих воздействий и автоматическим изменением режима управления;
- разработано программное обеспечение поддержки адаптивного управления электрообогревом, которое, в отличие от известных, поддерживает принципы объектно-ориентированного программирования, что дает возможность модификации, расширяемости и повторного использования.

В ходе анализа предметной области и математических моделей технологических процессов было принято решение о создании математической модели электрообогрева технологического оборудования атомной электростанции с использованием блочного принципа, которая учитывает все существующие режимы управления и позволяет вовремя предотвратить аварийные ситуации.

В ходе исследования был выбран метод адаптивного управления, под которым будем понимать автоматическое изменение режима работы алгоритма в зависимости от изменения параметров объекта управления или действий на объект со стороны эксплуатирующего персонала.

Анализируя алгоритмы управления технологическими процессами: ПРОТАР 120, ПАССАТ, СА СКУ ЭО, – принято решение о разработке базового алгоритма поддержки адаптивного управления с возможностью его применения для различных технологических систем в зависимости от изменения параметров, коэффициентов и физических величин.

Исследованы алгоритмы на основе ПИД-регулирования, алгоритмы на основе самонастройки, алгоритмы на основе нечеткой логики и нейронных сетей. Обобщив все достоинства и недостатки каждого алгоритма, выбор был сделан в пользу алгоритма ПИД-регулирования. Его главными преимуществами являются простота понимания и реализации, как для самих разработчиков алгоритма, так и для операторов АЭС, надежность, хорошее качество управления. Для настройки коэффициентов ПИД-регулятора были проанализированы методы: Циглера-Никольса, CHR (Chein, Hrones и Reswick) и метод ручной настройки, основанной на правилах.

Как результат разработки, представлена техническая реализация, построена диаграмма классов и компонентов разработанного программного обеспечения. Проведено функциональное и экспериментальное тестирование под операционной системой Debian-10.0 в SCADA-системе. По результатам тестирования можно судить о корректности и эффективности работы модели и алгоритма в заданных условиях.

Список использованных источников

1. Как работает АЭС // РОСЭНЕРГОАТОМ [Электронный ресурс]. URL: https://www.rosenergoatom.ru/stations_projects/atomnye-elektrostantsii-rossii/kak-rabotaet-aes/ (дата обращения: 31.03.2020)
2. Программа инновационного развития и технологической модернизации Госкорпорации «Росатом» на период до 2030 года (в гражданской части) [Электронный ресурс]. URL: <https://rosatom.ru/upload/iblock/5e1/5e130b6e7fba0fb511f400defad83aca.pdf> (дата обращения: 1.04.2020)
3. Митропольский, А. Н. Автоматизация системы электрообогрева на АЭС с быстрым реактором / А. Н. Митропольский, И. Х. Цукерман, А. А. Петренко // Атомная энергия. – 1975. – С. 83 – 85.
4. Демченко, В. А. Автоматизация и моделирование технологических процессов АЭС и ТЭС / В. А. Демченко. – Одесса: «Астропринт», 2001. – 303 с.
5. Звонарев, С.В. Основы математического моделирования / С.В. Звонарев. – Екатеринбург: «Уральский университет», 2019. – 112 с.
6. Александров, А. Г. Оптимальные и адаптивные системы / А. Г. Александров. – Москва: «Высшая школа», 2003. – 278 с.
7. Мякишев, Д.В. Комплекс «ПАССАТ» - набор базовых средств для построения АСУ высокой надежности / Д. В. Мякишев, Ю. А. Тархов, К.А. Столяров // Автоматизация в промышленности. – 2007. – №8. – С. 25 – 30.
8. Денисенко, В. ПИД-регуляторы: принципы построения и модификации / В. Денисенко // Современные технологии автоматизации. – 2006. – №4. – С. 66 – 74.
9. Сорокина, Н.В. Адаптивный алгоритм управления электрообогревом технологического оборудования АЭС / Н. В. Сорокина, И. Ю. Балашова // Информационные технологии в науке и образовании. Проблемы и перспективы. – 2019. – С. 223 – 225.

10. Бураков, М.В. Синтез нечетких логических регуляторов / М. В. Бураков, А. С. Коновалов // Информационно-управляющие системы. – 2011. – С. 22 – 27.
11. Абрамов, И. А. Применение методов нейроуправления в ПТК «УРАЛ-Атом» – концепция и перспективы / И. А. Абрамов, В. И. Горбаченко, П. Ю. Кочегаров, Д. В. Мякишев // Проблемы информатики в образовании, управлении, экономике и техники. – 2018. – С. 62 – 67.
12. Чернов, В. Г. Нечеткие контроллеры. Основы теории и построения / В. Г. Чернов. – Владимир: «Владимирский государственный университет», 2003. – 148 с.
13. Омату, С. Нейроуправление и его приложения / Сигеру Омату, Марзуки Халид, Рубия Юсоф. – Перевод с англ. Батина Н. – Москва: «ИПРЖР», 2000. – 272 с.
14. Сорокина, Н. В. Сравнительный анализ алгоритмов для системы управления электрообогревом технологического оборудования атомных электростанций / Н. В. Сорокина, И. Ю. Балашова // Новые информационные технологии и системы. – 2019. – С. 156 – 159.
15. Карпов, В.Э. ПИД-управление в нестрогом изложении / В.Э. Карпов. – Москва: «НИИ информационных технологий», 2012. – 27 с.
16. Сорокина, Н.В. Настройка коэффициентов ПИ-регулятора для адаптивной системы управления электрообогревом технологического оборудования атомной электростанции / Н. В. Сорокина, И. Ю. Балашова // Информационные технологии в науке и образовании. Проблемы и перспективы. – 2020. – С. 135 – 138.
17. Денисенко, В. ПИД-регуляторы: вопросы реализации / В. Денисенко // Современные технологии автоматизации. – 2008. – №1. – С. 86 – 99.
18. Вигерс, К. Разработка требований к программному обеспечению / К. Вигерс. – Перевод с англ. Леоновой Ю. – Москва: «Русская Редакция», 2004. – 576 с.
19. ГОСТ 25804.6-83. Аппаратура, приборы, устройства и оборудование

- систем управления технологическими процессами атомных электростанций. – Введ. 1984-01-01. – М. :Изд-во стандартов, 1984. – 38 с.
20. Савенко, И.И. Технология разработки программного обеспечения / И.И. Савенко. – Томск: «Томский политехнический университет», 2014. – 67 с.
21. Крачтен, Ф. Введение в Rational Unified Process / Ф. Крачтен. – Москва: «Вильямс», 2002. – 297 с.
22. Арлоу, Д. UML 2 и Унифицированный процесс / Д. Арлоу, А. Нейштадт. – 2-е издание, перевод с англ. Шатохиной Н. – Санкт-Петербург: «Символ-Плюс», 2007. – 624 с.
23. Садовский, Б. С. Руководство по использованию интегрированной среды разработки Code::Blocks / Б. С. Садовский. – Москва: «НИУ МГСУ», 2008. – 25 с.
24. Кангин, В. Разработка SCADA-систем / В. Кангин, Д. Ямолдинов, М. Кангин. – Вологда: «Инфа-Инженерия», 2019. – 564 с.
25. Мальков, М. В. Моделирование технологических процессов: методы и опыт / М. В. Мальков, А. Г. Олейник, А. М. Федоров // Сборник научных трудов 2010. – 2010. – С. 93 – 101.
26. Сорокина, Н. В. Разработка аппаратно-программного обеспечения системы управления электрообогревом технологического оборудования АЭС / Н. В. Сорокина, И. Ю. Балашова // Новые информационные технологии и системы. – 2018. – С. 159 – 162.
27. Мякишев, Д.В. Проектирование программного обеспечения автоматизированных систем управления технологическим процессом на основе объектных моделей / Д. В. Мякишев // Новые информационные технологии и системы. – 2019. – С. 168 – 171.
28. Степанченко, И. В. Методы тестирования программного обеспечения / И.В. Степанченко. – Волгоград: «ВолгГТУ», 2006. – 74 с.

Приложение А
ГЛОССАРИЙ
(рекомендуемое)

Автоматизированная система управления технологическими процессами (АСУ ТП) – это система управления с участием человека, выполняющая автоматизированный сбор и обработку данных, необходимых для действующего алгоритма управления.

Автоматическое регулирование (АР) – это изменение величины регулируемого параметра по заранее установленному закону.

Адаптивное управление – это управление, при котором параметры или алгоритмы управления приспособляются к возмущающим воздействиям с целью оптимального управления системой.

Атомная электростанция (АЭС) – это комплекс систем, оборудования и сооружений, предназначенных для генерации электроэнергии.

Блокировка – запрет выполнения операций на технологическом оборудовании, приводящим к возникновению аварийных ситуаций.

Дистанционное управление (ДУ) – это режим управления, при котором оператор выдает команду управления, а алгоритм проверяет возможность ее реализации.

Математическая модель – это описание реального объекта или процесса с использованием математических символов.

Технологическая защита (ТЗ) – защита от аварий действующего технологического оборудования.

Управление – это воздействие на объект с целью перевода его в требуемое состояние.

Функционально-групповое управление (ФГУ) – это последовательность команд управления, выданных оператором, для группы функционально связанного оборудования.

Электрообогрев – это поддержание температуры технологического оборудования с помощью электронагревателей.

Приложение Б
ЛИСТИНГ ПРОГРАММНЫХ СРЕДСТВ
(рекомендуемое)

Листинг Б.1 – Файл Main.cpp

```

#include "Channel.hpp"
#include "Variable.hpp"
#include "Actuator.hpp"
#include "Control.hpp"
#include "Processing.hpp"
#include "Subtask.hpp"
#include "Memory.hpp"
#include "Modbus.hpp"

#include <iostream>
using namespace std;

int main()
{
    uint16_t buf[2048] = {0};
    Memory m((uint16_t*)&buf[0], (size_t)sizeof(buf));
    Modbus mb(&m);
    const char dev_name[] = "NA";
    // Коэффициент пропорциональной составляющей
    Parameter_channel k_p(0x32F43A, mb, (char*)"0НАС01НН001.32F43A000", 50,
50);
    // Коэффициент интегральной составляющей
    Parameter_channel k_i(0x32F43C, mb, (char*)"0НАС01НН001.32F43C000", 50,
50);
    // Верхняя граница измерения температуры (датчик)
    Parameter_channel t1_ht(0x320800, mb, (char*)"0НАС01НН001.320800000", 0,
1000);
    // Нижняя граница измерения температуры (датчик)
    Parameter_channel t1_lt(0x320802, mb, (char*)"0НАС01НН001.320802000", 0,
1000);
    // Верхняя предаварийная граница температуры (датчик)
    Parameter_channel t1_htp(0x320804, mb, (char*)"0НАС01НН001.320804000",
0, 1000);
    // Нижняя предаварийная граница температуры (датчик)
    Parameter_channel t1_ltp(0x320806, mb, (char*)"0НАС01НН001.320806000",
0, 1000);
    // Верхняя аварийная граница температуры (датчик)
    Parameter_channel t1_hta(0x320808, mb, (char*)"0НАС01НН001.320808000",
0, 1000);
    // Нижняя аварийная граница температуры (датчик)
    Parameter_channel t1_lta(0x32080A, mb, (char*)"0НАС01НН001.32080A000",
0, 1000);

    // Верхняя граница измерения температуры (задатчик)
    Parameter_channel t2_ht(0x320820, mb, (char*)"0НАС01НН001.320820000", 0,
1000);
    // Нижняя граница измерения температуры (задатчик)
    Parameter_channel t2_lt(0x320822, mb, (char*)"0НАС01НН001.320822000", 0,
1000);
    // Верхняя предаварийная граница температуры (задатчик)
    Parameter_channel t2_htp(0x320824, mb, (char*)"0НАС01НН001.320824000",
0, 1000);
    // Нижняя предаварийная граница температуры (задатчик)
    Parameter_channel t2_ltp(0x320826, mb, (char*)"0НАС01НН001.320826000",
0, 1000);
    // Верхняя аварийная граница температуры (задатчик)
    Parameter_channel t2_hta(0x320828, mb, (char*)"0НАС01НН001.320828000",
0, 1000);
    // Нижняя аварийная граница температуры (задатчик)
    Parameter_channel t2_lta(0x32082A, mb, (char*)"0НАС01НН001.32082A000",
0, 1000);
    // Блокировка с ВУ

```

```

Parameter_channel lvu(0x02F4A6, mb, (char*)"0HAC01HH001.02F4A6000", 0,
5);
// Коэффициент сглаживания
Parameter_channel kvu(0x32F4A8, mb, (char*)"0HAC01HH001.32F4A8000", 0,
2.0f);
// Максимальная скорость изменения температуры
Parameter_channel spvu(0x32F4AA, mb, (char*)"0HAC01HH001.32F4AA000", 0,
50.0f);
// ДУ, ФГУ, включение с ВУ
Parameter_channel dfvu(0x02F4AC, mb, (char*)"0HAC01HH001.02F4AC000", 0,
1000);
// Время в мсек для перехода в новое состояние
Parameter_channel heater_time(0x02F4AD, mb,
(char*)"0HAC01HH001.02F4AD000", 0, 90000);
// Разница между датчиком и задатчиком, при которой переходим в ДУ
Parameter_channel dt(0x32F4E0, mb, (char*)"0HAC01HH001.32F4E0000", 0,
1000);
// Гистерезис для ПИ-регулятора
Parameter_channel g_na(0x32F4E2, mb, (char*)"0HAC01HH001.32F4E2000", 0,
100);
Parameter_channel sti0(0x030400, mb, (char*)"0HAC01HH001.020400000", 0,
65535);
Parameter_channel sti1(0x030402, mb, (char*)"0HAC01HH001.020402000", 0,
65535);
Parameter_channel sti2(0x030430, mb, (char*)"0HAC01HH001.020430000", 0,
65535);
Parameter_channel sti3(0x030431, mb, (char*)"0HAC01HH001.020431000", 0,
65535);
Parameter_channel sti4(0x030432, mb, (char*)"0HAC01HH001.020432000", 0,
65535);
Parameter_channel sti5(0x030433, mb, (char*)"0HAC01HH001.020433000", 0,
65535);
Parameter_channel sti6(0x030434, mb, (char*)"0HAC01HH001.020434000", 0,
65535);
Parameter_channel sti7(0x030435, mb, (char*)"0HAC01HH001.020435000", 0,
65535);
Parameter_channel sti8(0x030440, mb, (char*)"0HAC01HH001.020440000", 0,
65535);
Parameter_channel sti9(0x030441, mb, (char*)"0HAC01HH001.020441000", 0,
65535);
Status_channel st0(0x02F4C0, mb, (char*)"0HAC01HH001.02F4C0000", 0,
65535);
Status_channel st1(0x02F4C2, mb, (char*)"0HAC01HH001.02F4C2000", 0,
65535);
Status_channel st2(0x02F4C8, mb, (char*)"0HAC01HH001.02F4C8000", 0,
65535);
Status_channel st3(0x02F4C9, mb, (char*)"0HAC01HH001.02F4C9000", 0,
65535);
Status_channel st4(0x02F4CA, mb, (char*)"0HAC01HH001.02F4CA000", 0,
65535);
Status_channel st5(0x02F4CB, mb, (char*)"0HAC01HH001.02F4CB000", 0,
65535);
Status_channel st6(0x02F4CC, mb, (char*)"0HAC01HH001.02F4CC000", 0,
65535);
Status_channel st7(0x02F4CD, mb, (char*)"0HAC01HH001.02F4CD000", 0,
65535);
Status_channel st8(0x02F4CE, mb, (char*)"0HAC01HH001.02F4CE000", 0,
65535);
Status_channel st9(0x02F4CF, mb, (char*)"0HAC01HH001.02F4CF000", 0,
65535);

// Состояние "включен"
Input_channel i_on(0x010050, mb, (char*)"0HAC01HH001.010050003", 0, 1,
&st2, &sti2);
// Состояние "выключен"

```

```

    Input_channel i_off(0x010051, mb, (char*)"0HAC01HH001.010051004", 0, 1,
&st3, &sti3);
    // Обрыв цепи нагревателя
    Input_channel brk_heater_chain(0x010052, mb,
(char*)"0HAC01HH001.010052005", 0, 1, &st4, &sti4);
    // Блок в тестовом положении
    Input_channel test(0x010053, mb, (char*)"0HAC01HH001.010053006", 0, 1,
&st5, &sti5);
    // Выключатель автоматический отключен
    Input_channel auto_off(0x010054, mb, (char*)"0HAC01HH001.010054007", 0,
1, &st6, &sti6);
    // Повреждение изоляции
    Input_channel ins_damage(0x010055, mb, (char*)"0HAC01HH001.010055008",
0, 1, &st7, &sti7);
    // Команда "включить"
    Output_channel sw_on(0x000050, mb, (char*)"0HAC01HH001.000050001", 0, 1,
&st8);
    // Команда "выключить"
    Output_channel sw_off(0x000051, mb, (char*)"0HAC01HH001.000051002", 0,
1, &st9);
    // Причины ТЗ и блокировки + режим
    Status_channel tld(0x02F4A4, mb, (char*)"0HAC01HH001.02F4A4000", 0, 20);
//protokol
    // Температура
    Input_channel t1(0x330100, mb, (char*)"0HAC01HH001.330100007", 4000.0f,
20000.0f, 320.0f, &st0, &sti0);
    Input_channel t2(0x330104, mb, (char*)"0HAC01HH001.330104008", 4000.0f,
20000.0f, 320.0f, &st1, &sti1);
    Input_channel t3(0x330100, mb, (char*)"0HAC01HH001.330100009", 4000.0f,
20000.0f, 320.0f, &st0, &sti0);
    Input_channel t4(0x330104, mb, (char*)"0HAC01HH001.330104010", 4000.0f,
20000.0f, 320.0f, &st1, &sti1);
    // Резерв
    Temperature temp1(&t1, (char*)"0HAC01CT001", &t1_ht, &t1_lt, &t1_hpt,
&t1_ltp, &t1_hpa, &t1_lpa, 20000, 4000, 0x7fff, &kvu, &spvu, &st0, &sti0);
    // Резерв
    Temperature temp2(&t2, (char*)"0HAC01CT002", &t2_ht, &t2_lt, &t2_hpt,
&t2_ltp, &t2_hpa, &t2_lpa, 20000, 4000, 0x7fff, &kvu, &spvu, &st1, &sti1);
    // Датчик
    Temperature temp3(&t3, (char*)"0HAC01CT003", &t1_ht, &t1_lt, &t1_hpt,
&t1_ltp, &t1_hpa, &t1_lpa, 20000, 4000, 0x7fff, &kvu, &spvu, &st0, &sti0);
    // Задатчик
    Temperature temp4(&t4, (char*)"0HAC01CT004", &t2_ht, &t2_lt, &t2_hpt,
&t2_ltp, &t2_hpa, &t2_lpa, 20000, 4000, 0x7fff, &kvu, &spvu, &st1, &sti1);
    // Температура, физическая величина
    Output_channel phys_t1(0x330000, mb, (char*)"0HAC01HH001.330000009",
245, 555);
    Output_channel phys_t2(0x330004, mb, (char*)"0HAC01HH001.330004010",
245, 555);
    Output_channel phys_t3(0x330000, mb, (char*)"0HAC01HH001.330000009",
245, 555);
    Output_channel phys_t4(0x330004, mb, (char*)"0HAC01HH001.330004010",
245, 555);
    Output_channel led_l(0x020200, mb, (char*)"0HAC01HH001.020200000", 0,
62000);
    Output_channel led_r(0x020201, mb, (char*)"0HAC01HH001.020201001", 0,
62000);

    // Нагреватель
    Heater heater(&sw_on, &sw_off, &i_on, &i_off, &brk_heater_chain,
&test, &auto_off, &ins_damage, &tld, &heater_time);
    // Артефакт обработки и управления - технологическая защита
    Tech_protection tp(&heater, 0);
    // Артефакт обработки и управления - блокировка
    Lock lc(&heater, &lvu, &temp3, &temp4);

```

```

//Arteфакт обработки и управления - дистанционное управление
Remote_control rc(&heater, &dfvu);
//Arteфакт обработки и управления - функционально-групповое управление
Fgroup_control fg(&heater, &dfvu);
//Arteфакт обработки и управления - ПИ-контроллер
PI_control reg(&k_p, &k_i, &t1_ht, &t1_lt, 100, &spvu, &dt);
//Подзадача управления обменом с верхним уровнем
VU_control_NA vu_na(mb, &i_on, &i_off, &brk_heater_chain, &test,
&auto_off, &ins_damage, &sw_on, &sw_off, &tld, &heater, &phys_t3, &phys_t4,
&temp3, &temp4, &lled_l, &lled_r, &lvu);
//Подзадача управления нагревателем
Control_heater ch_heater(&temp1, &temp2, &temp3, &temp4, &heater, &reg,
&tp, &lc, &rc, &fg);

MA_control_NA ma_na(&ch_heater);
ma_na.init_ma_na();
vu_na.init_vu_na();

ma_na.run_ma_na();
vu_na.run_vu_na();

return 0;
}

```

Листинг Б.2 – Файл Actuator.hpp

```

#ifndef ACTUATOR_HPP
#define ACTUATOR_HPP
#include "Channel.hpp"
#include "Variable.hpp"

class Actuator
{
protected:
    int a;
    Output_channel* ch;

    //Режим работы
    struct workbit
    {
        unsigned w0: 1;
        unsigned w1: 1;
        unsigned w2: 1;
        unsigned w3: 1;
        unsigned w4: 1;
        unsigned w5: 1;
        unsigned w6: 1;
        unsigned w7: 1;
        unsigned w8: 1;
        unsigned w9: 1;
        unsigned w10: 1;
        unsigned w11: 1;
        unsigned w12: 1;
        unsigned w13: 1;
        unsigned w14: 1;
        unsigned w15: 1;
    };
    union work_code
    {
        unsigned char value_mode;
        struct workbit mode;
    };

    //Признаки достоверности значения

```

```

        union work_code work_flags;

public:
    Actuator ()
    {
    }
    Actuator (Output_channel* ptch)
    {
        ch = ptch;
    }
    ~Actuator()
    {
    }
    // Получить состояние
    int get_state();
    // Выдать команду
    void put_command(int c);
    // Технологическая защита
    int set_tech_protection_t1();
    // Технологическая защита
    int set_tech_protection_t2();
    // Блокировка
    int set_locking_l1();
    // Блокировка
    int set_locking_l2();
    // Блокировка
    int set_locking_l3();
    // Блокировка
    int set_locking_l4();
    // Блокировка
    int set_locking_l5();
    // Блокировка
    int set_locking_l6();
    // Блокировка
    int set_locking_l7();
    // Блокировка
    int set_locking_l8();
    // Команда ДУ
    int set_remote_control_rc1();
    // Команда ДУ
    int set_remote_control_rc2();
    // Команда ДУ
    int set_remote_control_rc3();
    // Команда функционально-группового управления
    int set_function_group_control_fg1();
    // Команда функционально-группового управления
    int set_function_group_control_fg2();
    // Авторегулирование
    int set_automatic_control();
    // Получить текущий режим работы
    int get_mode_m();
};
#endif

```

Листинг Б.3 – Файл Channel.hpp

```

#ifndef CHANNEL_HPP
#define CHANNEL_HPP
#define VALUE_IS_NON (0x7FFF)
#include "Modbus.hpp"

using namespace std;

union _ret_float

```

```

{
    float f_val;
    uint16_t u_val[2];
};
struct KKS
{
    char* bn;
    char* sc;
    char* sn;
    char* tuc;
    char* tun;
};
union KKS_repr
{
    char* KKS_whole;
    struct KKS KKS_field;
};
class Channel
{
protected:
    float a;
    volatile uint32_t pio_fun;
    volatile uint32_t pio_reg;
    unsigned long pio_data;
    class Modbus* _modbus;

public:
    Channel(unsigned long pio, Modbus* _m);
    ~Channel()
    {
    }
};

// Статусы
class Status_channel: Channel
{
    int min_status;
    int max_status;
    char* KKS_output_status;
public:
    Status_channel(unsigned long pio, Modbus* modbu, char* iKKS_s, int
min_s, int max_s):Channel(pio, modbu)
    {
        min_status=min_s;
        max_status=max_s;
        KKS_output_status = iKKS_s;
    }

    // Установить минимальное значение сигнала
    int set_min_osign_status(int min_os);
    // Установить максимальное значение сигнала
    int set_max_osign_status(int max_os);
    // Получить код KKS
    char* get_output_KKS_status();
    void damage(int on_off);
    void put_value_status(int a);
};

class Output_channel: Channel
{
    int min_s;
    int max_s;
    char* KKS_output;
    Status_channel* o_st;
    _ret_float fout;

```

```

    public:
        Output_channel(unsigned long pio, Modbus* modbu, char* iKKS, int
min_, int max_):Channel(pio, modbu)
        {
            min_s=min_;
            max_s=max_;
            KKS_output = iKKS;
        }
        Output_channel(unsigned long pio, Modbus* modbu, char* iKKS, int
min_, int max_, Status_channel* ost):Channel(pio, modbu)
        {
            min_s=min_;
            max_s=max_;
            KKS_output = iKKS;
            o_st = ost;
        }
        ~Output_channel()
        {
        }

        void put_value(int c);
        void put_value(float c);
        // Установить минимальное значение сигнала
        int set_min_osign(int min_os);
        // Установить максимальное значение сигнала
        int set_max_osign(int max_os);
        // Получить минимальное значение
        int get_min_osign();
        // Получить максимальное значение сигнала
        int get_max_osign();
        // Получить код KKS
        char* get_output_KKS();
};

class Parameter_channel: Channel
{
    private:
        // Текущее значение принятого сигнала
        int sign_value;
        // Минимальное значение сигнала
        float min_sign;
        // Максимальное значение сигнала
        float max_sign;
        _ret_float fret;
        // Код KKS
        char* KKS;
    public:
        Parameter_channel(unsigned long pio, Modbus* modbu, char* iKKS,
float min_val, float max_val): Channel(pio, modbu)
        {
            sign_value = 0;
            min_sign = min_val;
            max_sign = max_val;
            KKS = iKKS;
        }
        ~Parameter_channel()
        {
        }

        // Получить значение
        float get_value_parameter();
        // Установить минимальное значение сигнала
        int set_min_sign_parameter(int min_s);
        // Установить максимальное значение сигнала
        int set_max_sign_parameter(int max_s);

```

```

        // Получить код KKS
        char* get_KKS_parameter();
        int out_of_range_(bool ft, int n);
};

class Input_channel: Channel
{
private:
    // Текущее значение принятого сигнала
    int sign_value;
    // Минимальное значение сигнала
    int min_sign;
    // Максимальное значение сигнала
    int max_sign;
    _ret_float fret;
    // Код KKS
    char* KKS;
    float delta;
    Status_channel* och_st;
    Parameter_channel* ich_st;
public:
    Input_channel(unsigned long pio, Modbus* modbu, char* iKKS, int
min_val, int max_val):Channel(pio, modbu)
    {
        min_sign=min_val;
        max_sign=max_val;
        KKS = iKKS;
    }
    Input_channel(unsigned long pio, Modbus* modbu, char* iKKS, int
min_val, int max_val, Status_channel* ost, Parameter_channel*
ist):Channel(pio, modbu)
    {
        min_sign=min_val;
        max_sign=max_val;
        KKS = iKKS;
        och_st = ost;
        ich_st = ist;
    }
    Input_channel(unsigned long pio, Modbus* modbu, char* iKKS, int
min_val, int max_val, float idelta, Status_channel* ost, Parameter_channel*
ist):Channel(pio, modbu)
    {
        min_sign=min_val;
        max_sign=max_val;
        KKS = iKKS;
        delta = idelta;
        och_st = ost;
        ich_st = ist;
    }
    ~Input_channel()
    {
    }
    // Получить значение
    float get_value();
    // Установить минимальное значение сигнала
    int set_min_sign(int min_s);
    // Установить максимальное значение сигнала
    int set_max_sign(int max_s);
    // Получить код KKS
    char* get_KKS();
};
#endif

```

Листинг Б.4 – Файл Control.hpp

```

#ifndef CONTROL_HPP
#define CONTROL_HPP
#include "Subtask.hpp"
class Control
{
    public:
        Control()
        {
        };
        void init();
        void run();
};

class MA_control_NA:public Control
{
    protected:
        Control_heater* ch_heater_;
    public:
        MA_control_NA(Control_heater* ch_h)
        {
            ch_heater_ = ch_h;
        };
        void init_ma_na();
        void run_ma_na();
};

class VU_control_NA:public Control
{
    protected:
        Modbus* mo;
        char n;
        char *tz;
        uint32_t now;
        // Состояние "включен"
        Input_channel* channel_on;
        // Состояние "выключен"
        Input_channel* channel_off;
        // Обрыв цепи нагревателя
        Input_channel* channel_brk;
        // Блок в тестовом положении
        Input_channel* channel_test;
        // Выключатель автоматический отключен
        Input_channel* channel_auto_off;
        // Повреждение изоляции
        Input_channel* channel_damage;
        Output_channel* channel_sw_on;
        Output_channel* channel_sw_off;
        Status_channel* channel_tld;
        Output_channel* p_t3;
        Output_channel* p_t4;
        Temperature* t3;
        Temperature* t4;
        Output_channel* l_led;
        Output_channel* r_led;
        Heater* heat;
        int value_tld;
        Parameter_channel* llvu;
    public:
        VU_control_NA(Modbus *mb, Input_channel* ion, Input_channel* ioff,
        Input_channel* ibrk_heater_chain, Input_channel* itest, Input_channel*
        iauto_off, Input_channel* iins_damage, Output_channel* isw_on,
        Output_channel* isw_off, Status_channel* tld, Heater* h, Output_channel*

```

```

ip_t3,    Output_channel*    ip_t4,    Temperature*    it3,    Temperature*    it4,
Output_channel* l, Output_channel* r, Parameter_channel* ilvu)
{
    mo = mb;
    channel_on = ion;
    channel_off = ioff;
    channel_brk = ibrk_heater_chain;
    channel_test = itest;
    channel_auto_off = iauto_off;
    channel_damage = iins_damage;
    channel_sw_on = isw_on;
    channel_sw_off = isw_off;
    channel_tld = tld;
    heat = h;
    p_t3 = ip_t3;
    p_t4 = ip_t4;
    t3 = it3;
    t4 = it4;
    l_led = l;
    r_led = r;
    llvu = ilvu;
};
void init_vu_na();
void run_vu_na();
};

#endif //CONTROL_HPP

```

Листинг Б.5 – Файл Heater.hpp

```

#ifndef HEATER_HPP
#define HEATER_HPP
#include "Actuator.hpp"
#include "Channel.hpp"
#include "Variable.hpp"
class Heater:public Actuator
{
protected:
    // Включить
    Output_channel* on;
    // Выключить
    Output_channel* off;
    // Состояние "включен"
    Input_channel* state_on;
    // Состояние "выключен"
    Input_channel* state_off;
    // Обрыв цепи нагревателя
    Input_channel* state_brk;
    // Блок в тестовом положении
    Input_channel* state_test;
    // Выключатель автоматический отключен
    Input_channel* state_auto_off;
    // Повреждение изоляции
    Input_channel* state_ins_damage;
    // Статус для верхнего уровня
    Status_channel* state_vu;
    // Максимальное время изменения состояния нагревателя
    Parameter_channel* h_time;

    int v_value_tld;
    int h_s_on;
    int h_s_off;
    // Время ожидания команды включить

```

```

        float time_on;
        // Время ожидания команды выключить
        float time_off;
    public:
        Heater      (Output_channel*   pton,      Output_channel*   ptoff,
Input_channel*   ptson,   Input_channel*   ptsoff,   Input_channel*   ptbreak,
Input_channel*   pttest,   Input_channel*   ptaoff,   Input_channel*   ptdam,
Status_channel* pttld, Parameter_channel* h_itime)
        {
            on = pton;
            off = ptoff;
            state_on = ptson;
            state_off = ptsoff;
            state_brk = ptbreak;
            state_test = pttest;
            state_auto_off = ptaoff;
            state_ins_damage = ptdam;
            state_vu = pttld;
            v_value_tld = 0;
            work_flags.mode.w0=0;
            work_flags.mode.w1=0;
            work_flags.mode.w2=0;
            work_flags.mode.w3=0;
            work_flags.mode.w4=0;
            h_s_on = 0;
            h_s_off = 0;
            time_on = -1;
            time_off = -1;
            h_time = h_itime;
        }
        // Включить
        void put_command_on(int c);
        // Выключить
        void put_command_off(int c);
        // Состояние команды включить
        bool s_on();
        // Состояние команды выключить
        bool s_off();
        // Состояние "включен"
        int is_on();
        // Состояние "выключен"
        int is_off();
        // Обрыв цепи нагревателя
        int is_break();
        // Блок в тестовом положении
        int is_test();
        // Выключатель автоматический отключен
        int is_auto_off();
        // Повреждение изоляции
        int is_damage();
        // Проверка работы состояния "включен"
        int check_on();
        // Проверка работы состояния "выключен"
        int check_off();

};
#endif

```

Листинг Б.6 – Файл Processing.hpp

```

#ifndef PROCESSING_HPP
#define PROCESSING_HPP
#include "Channel.hpp"
#include "Variable.hpp"

```

```

#include "Actuator.hpp"
#include "Heater.hpp"
#include <math.h>
#include <unistd.h>

class Tech_protection
{
protected:
    Pressure* prs_value;
    Control_valve* cv_t;
    Heater* hh;
    int in_auto;

public:
    Tech_protection (Heater* ihh, int aut)
    {
        hh = ihh;
        in_auto = aut;
    }
    ~Tech_protection()
    {
    }
    int start_Na(Heater* h);
};

class Lock
{
protected:
    Heater* hl;
    Parameter_channel* l_vu;
    Temperature* t1;
    Temperature* t2;
public:
    Lock (Heater* ihl, Parameter_channel* lock_vu, Temperature* it1,
Temperature* it2)
    {
        hl = ihl;
        l_vu = lock_vu;
        t1 = it1;
        t2 = it2;
    }
    ~Lock()
    {
    }
    int start_Na(Heater* hl);
};

class Remote_control
{
protected:
    int value;
    Heater* hr;
    int flag_rem;
    Parameter_channel* d_vu;

public:
    Remote_control (Heater* ihr, Parameter_channel* du_vu)
    {
        hr = ihr;
        d_vu = du_vu;
    }
    ~Remote_control()

```

```

        {
        }
        int start_Na();
};
class Fgroup_control
{
protected:
    int value;
    Heater* hfgr;
    int flag_fgr;
    Parameter_channel* f_vu;
public:
    Fgroup_control (Heater* ihfgr, Parameter_channel* fg_vu)
    {
        hfgr = ihfgr;
        f_vu = fg_vu;
    }
    ~Fgroup_control()
    {
    }
    int start_Na();
};
class Auto_control
{
public:
    Auto_control ()
    {
    }
    ~Auto_control()
    {
    }
    int start ();
};
// Пропорционально-интегральный регулятор
class PI_control:public Auto_control
{
protected:
    // Коэффициент пропорциональной составляющей
    Parameter_channel* kp;
    // Коэффициент интегральной составляющей
    Parameter_channel* ki;
    // Максимальное значение регулируемого параметра
    Parameter_channel* e_max;
    // Минимальное значение регулируемого параметра
    Parameter_channel* e_min;
    // Допустимое отклонение регулируемого параметра от заданного
    unsigned delta;
    // Предыдущее значение датчика
    int current_value;
    // Предыдущее значение задатчика
    int set_value;
    // Предыдущее рассогласование
    int e_curr;
    // Пропорциональная составляющая
    int p_comp;
    // Интегральная составляющая
    int i_comp;
    Parameter_channel* sp;
    Parameter_channel* dtt;

public:
    PI_control (Parameter_channel* k_p,      Parameter_channel* k_i,
Parameter_channel* ae_max,  Parameter_channel* ae_min,  unsigned adelta,
Parameter_channel* s,  Parameter_channel* iddt)
    {

```

```

        kp = k_p;
        ki = k_i;
        e_max = ae_max;
        e_min = ae_min;
        delta = adelta;
        current_value = 0;
        set_value = 0;
        e_curr = 0;
        p_comp = 0;
        i_comp = 0;
        sp = s;
        dtt = iddt;
    }
    void tuning (Parameter_channel* akp, Parameter_channel* aki)
    {
        kp = akp;
        ki = aki;
    }
    // Автоматическое регулирование
    int control_PI (float curr_val, float set_val);
};
#endif

```

Листинг Б.7 – Файл Subtask.hpp

```

#ifndef SUBTASK_HPP
#define SUBTASK_HPP
#include "Variable.hpp"
#include "Actuator.hpp"
#include "Heater.hpp"
#include "Processing.hpp"

class Subtask
{
protected:
    Variable* var;
    Actuator* act;
    int a;
public:
    Subtask ()
    {
    }
    Subtask (Variable* ptvar, Actuator* ptact )
    {
        var = ptvar;
        act = ptact;
    }
    ~Subtask()
    {
    }
    void pivot ();
};

class Control_heater:public Subtask
{
protected:
    Temperature* t_one;
    Temperature* t_two;
    Temperature* t_three;
    Temperature* t_four;
    Heater* ht;
    Tech_protection* tp;
    Lock* lc;
    Remote_control* rc;
    Fgroup_control* fg;

```

```

        PI_control*    ac;
        Output_channel *vu;
        int a;
        int count;

    public:
        Control_heater (Temperature* ptt1, Temperature* ptt2, Temperature*
ptt3, Temperature* ptt4, Heater* ptact, PI_control* reg, Tech_protection*
pttp, Lock* ptlc, Remote_control* ptrc, Fgroup_control* ptfg)
        {
            t_one = ptt1;
            t_two = ptt2;
            t_three = ptt3;
            t_four = ptt4;
            ht = ptact;
            ac = reg;
            tp = pttp;
            lc = ptlc;
            rc = ptrc;
            fg = ptfg;
            count = 0;
        }
        void pivoth_Na();
};

#endif

```

Листинг Б.8 – Файл Variable.hpp

```

#ifndef VARIABLE_HPP
#define VARIABLE_HPP
#include "Variable.hpp"
#include "Channel.hpp"

#include <math.h>
#include <ctime>

class Variable
{
    protected:
        float a;
        Input_channel* ch;
        // Верхняя граница
        Parameter_channel* h_var;
        // Нижняя граница
        Parameter_channel* l_var;
        // Верхняя граница предаварийная
        Parameter_channel* hp_var;
        // Нижняя граница предаварийная
        Parameter_channel* lp_var;
        // Верхняя граница аварийная
        Parameter_channel* ha_var;
        // Нижняя граница аварийная
        Parameter_channel* la_var;
        // Верхняя граница тока
        int h_am;
        // Нижняя граница тока
        int l_am;
        float z;
        // Старое значение
        float old_var;
        // Коэффициент сглаживания
        Parameter_channel* k;
        // Разница значений

```

```

Parameter_channel* speed;
// Значение
float vv;
long now_time;
long prev_time;
Status_channel* och_st;
Parameter_channel* ich_st;

public:
Variable()
{
    a = 0;
}
Variable(Input_channel* ptch)
{
    a = 1;
    ch = ptch;
}
Variable(Input_channel* ptch, char* kks, Parameter_channel*
h_ivar, Parameter_channel* l_ivar, Parameter_channel* hp_ivar,
Parameter_channel* lp_ivar, Parameter_channel* ha_ivar, Parameter_channel*
la_ivar, int h_iam, int l_iam, float old_ivar, Parameter_channel* ik,
Parameter_channel* sp, Status_channel* ioch_st, Parameter_channel* iich_st)
{
    a = 1;
    ch = ptch;
    h_var = h_ivar;
    l_var = l_ivar;
    hp_var = hp_ivar;
    lp_var = lp_ivar;
    ha_var = ha_ivar;
    la_var = la_ivar;
    h_am = h_iam;
    l_am = l_iam;
    z = 0;
    old_var = old_ivar;
    k = ik;
    speed = sp;
    och_st = ioch_st;
    ich_st = iich_st;
}
~Variable()
{
}
// Получение значения
float get_v();
// Получение значения
float get_value_v();
// Масштабирование
float scaling(float v);
// Фильтрация(сглаживание)
float smoothing(float sm);
// Линеаризация
float linearization(float v);
// Контроль диапазона
int check_range(float t);
// Скорость изменения
int check_speed(float sp);
// Получение значения из памяти
float get_var();
// Установить минимальное значение
int set_min_value(int vmin);
// Установить максимальное значение
int set_max_value(int vmax);
// Установить минимальное значение сигнала (тока)

```

```

        int set_min_sign(int min_am);
        // Установить максимальное значение сигнала (тока)
        int set_max_sign(int max_am);
        // Установить максимальное значение скорости изменения
        int set_max_speed(int s);
        // Установить коэффициент сглаживания
        int set_smooth_ratio(int sm_r);
        // Получить минимальное значение
        float get_min_value();
        // Получить максимальное значение
        float get_max_value();
};

class Temperature:public Variable
{
public:
    Temperature (Input_channel* ptch):Variable(ptch)
    {
    }

    Temperature (Input_channel* ptch, char* kks, Parameter_channel*
h_itep,      Parameter_channel*      l_itep,      Parameter_channel*      hp_itep,
Parameter_channel* lp_itep, Parameter_channel* ha_itep, Parameter_channel*
la_itep,int h_iam, int l_iam, int old_itep, Parameter_channel* ik,
Parameter_channel* sp, Status_channel* ioch_st,      Parameter_channel*
iich_st):Variable(ptch, kks, h_itep, l_itep, hp_itep, lp_itep, ha_itep,
la_itep, h_iam, l_iam, old_itep, ik, sp, ioch_st, iich_st)
    {
    }

    Temperature (Input_channel* ptch, char* kks, Parameter_channel*
h_itep,      Parameter_channel*      l_itep,      Parameter_channel*      hp_itep,
Parameter_channel* lp_itep, Parameter_channel* ha_itep, Parameter_channel*
la_itep,int h_iam, int l_iam, int old_itep, Parameter_channel* ik,
Parameter_channel* il, Parameter_channel* sp, Status_channel* ioch_st,
Parameter_channel* iich_st):Variable(ptch, kks, h_itep, l_itep, hp_itep,
lp_itep, ha_itep, la_itep, h_iam, l_iam, old_itep, ik, il, sp, ioch_st,
iich_st)
    {
    }

    ~Temperature()
    {
    }

};

#endif

```

Приложение В
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ
(рекомендуемое)

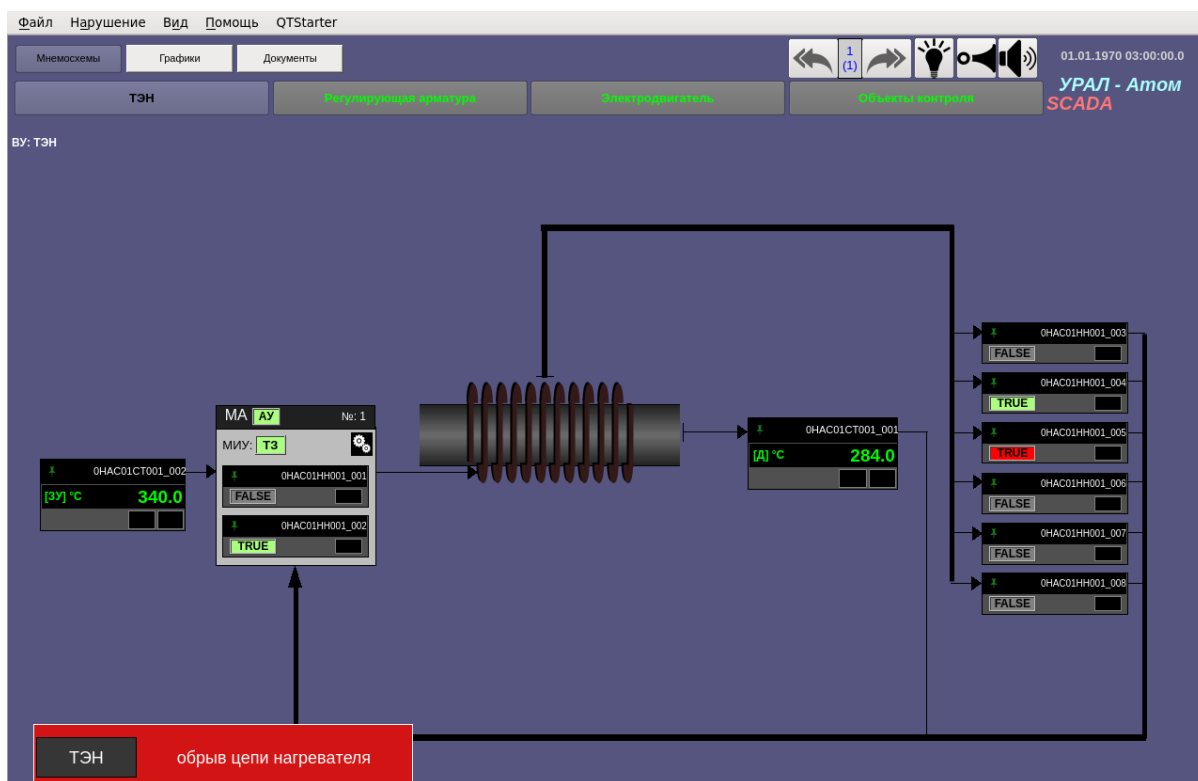


Рисунок В.1 – Включение блокировки оператором, режим технологической защиты

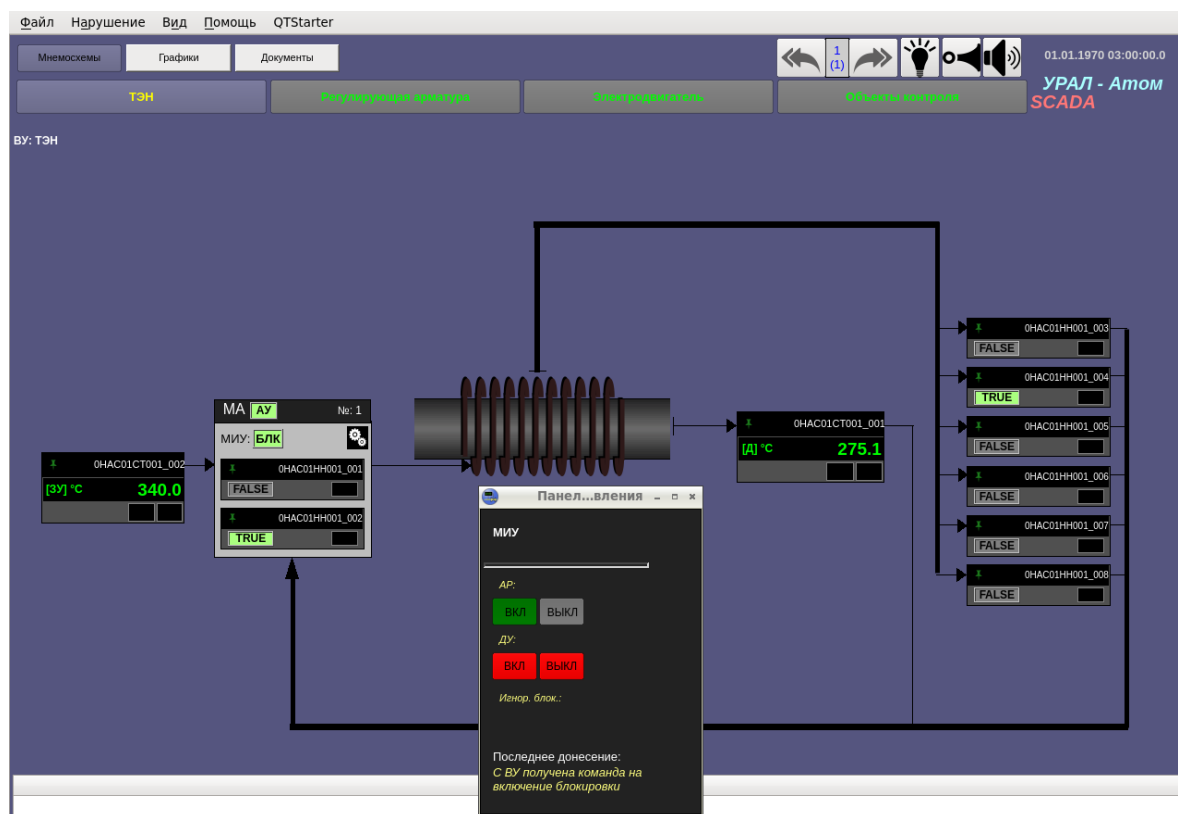


Рисунок В.2 – Включение блокировки оператором, режим блокировки

Рисунок В.4 – Выдана команда ДУ «включить», режим технологической защиты

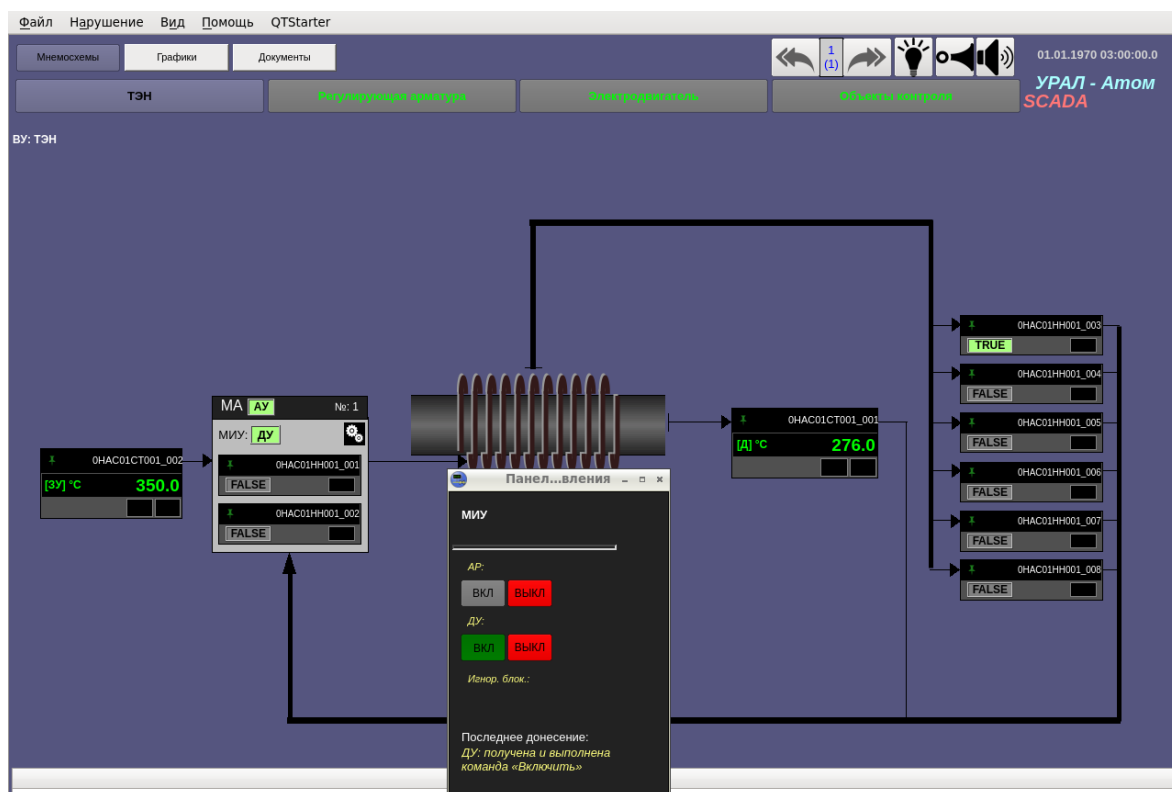


Рисунок В.5 – Выдана команда ДУ «включить», режим дистанционного управления



Рисунок В.6 – Установка коэффициента ПИ-регулятора, режим автоматического регулирования

Файл Нарушение Вид Помощь QTstarter

Мнемосхемы Графики Документы

ТЭН Регулирующая арматура Электродвигатель Объекты контроля

01.01.1970 03:00:00.0 УРАЛ - Атом SCADA

ВУ: ТЭН

MA AY No: 1

МИУ: T3

ОНАС01СТ001_002 350.0

ОНАС01НН001_001 FALSE

ОНАС01НН001_002 TRUE

ОНАС01СТ001_001 281.7

ОНАС01НН001_003 FALSE

ОНАС01НН001_004 TRUE

ОНАС01НН001_005 FALSE

ОНАС01НН001_006 FALSE

ОНАС01НН001_007 FALSE

ОНАС01НН001_008 TRUE

ТЭН повреждение изоляции

Рисунок В.8 – Повреждение изоляции, режим технологической защиты

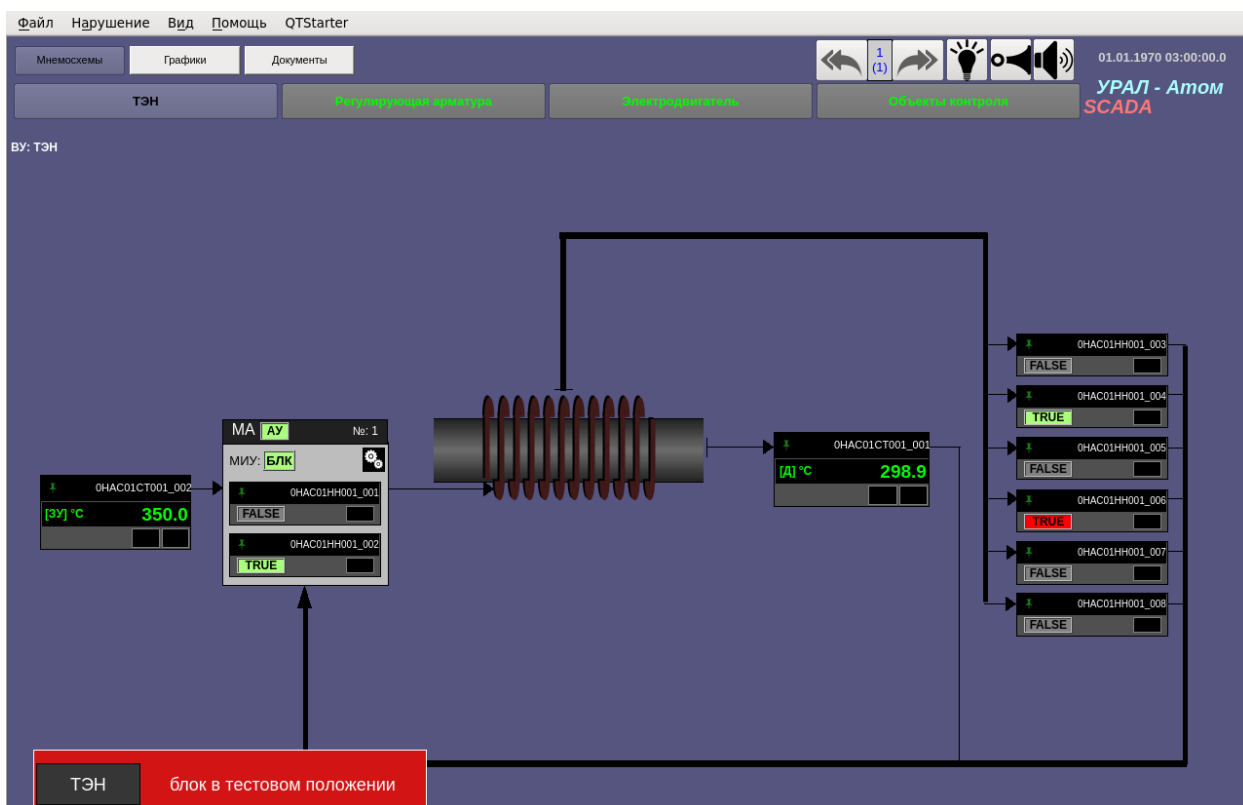


Рисунок В.9 – Блок в тестовом положении, режим блокировки

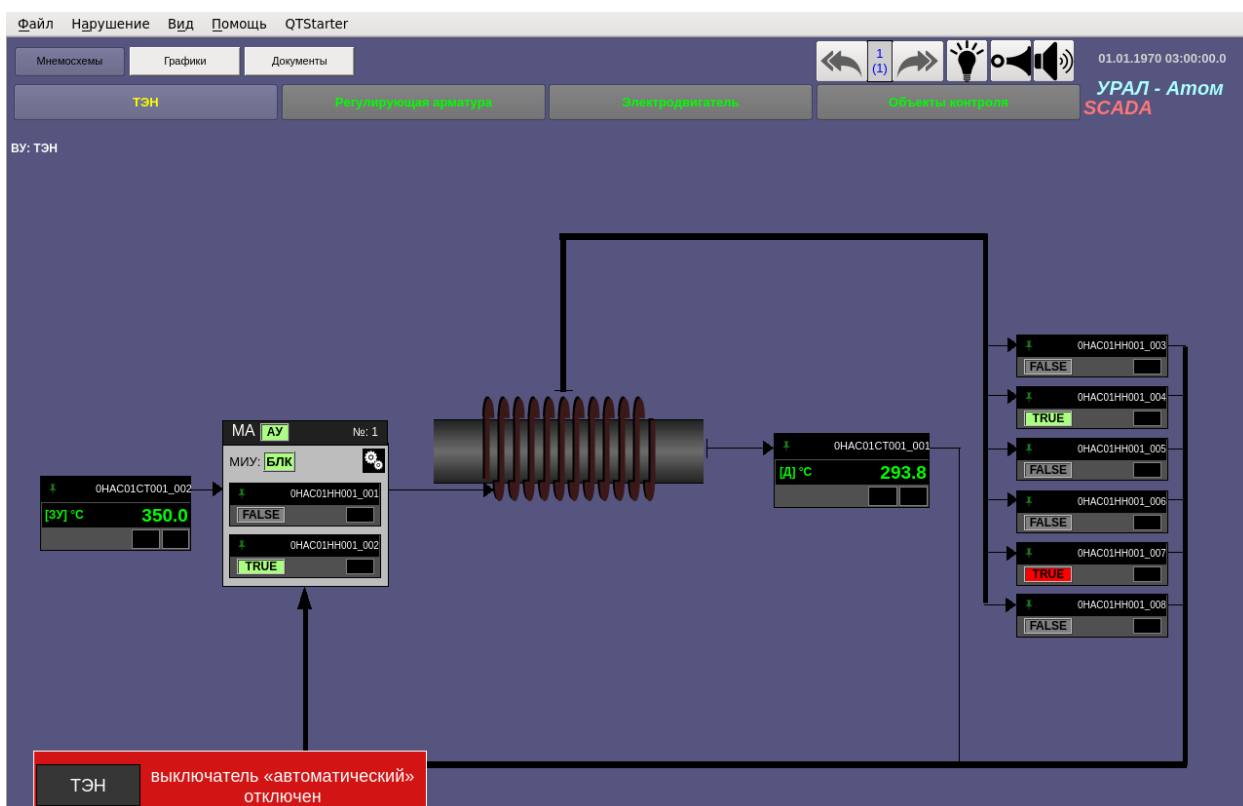


Рисунок В.10 – Выключатель автоматический отключен, режим блокировки

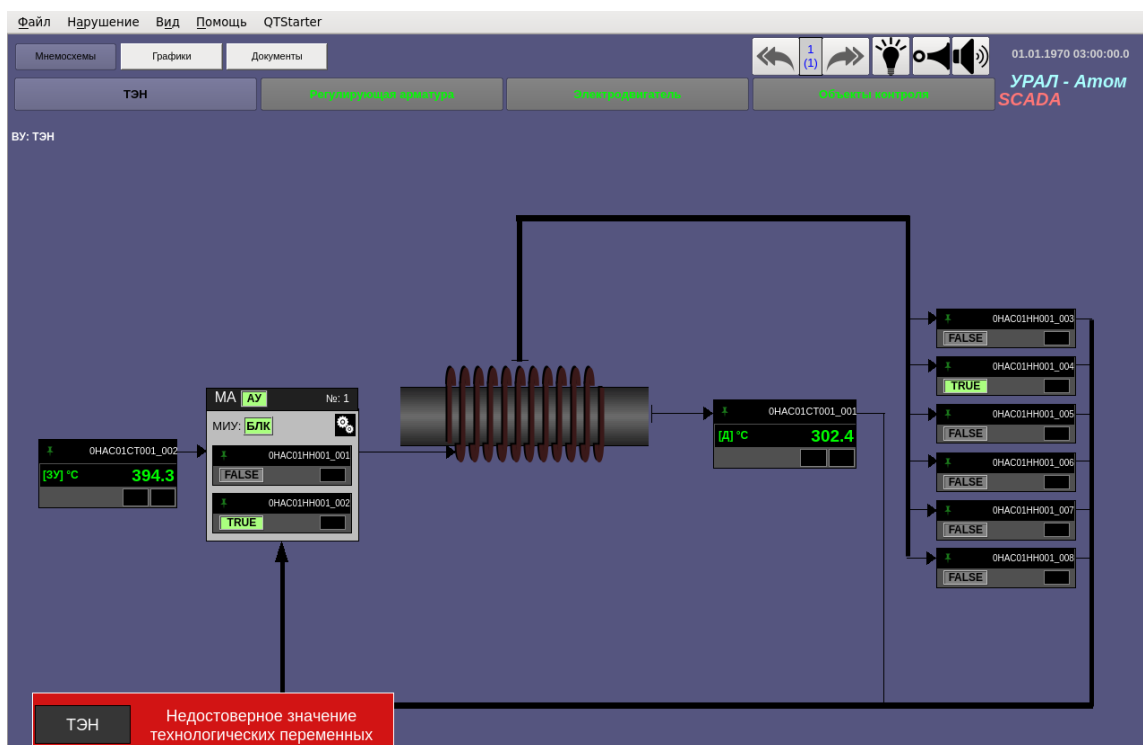
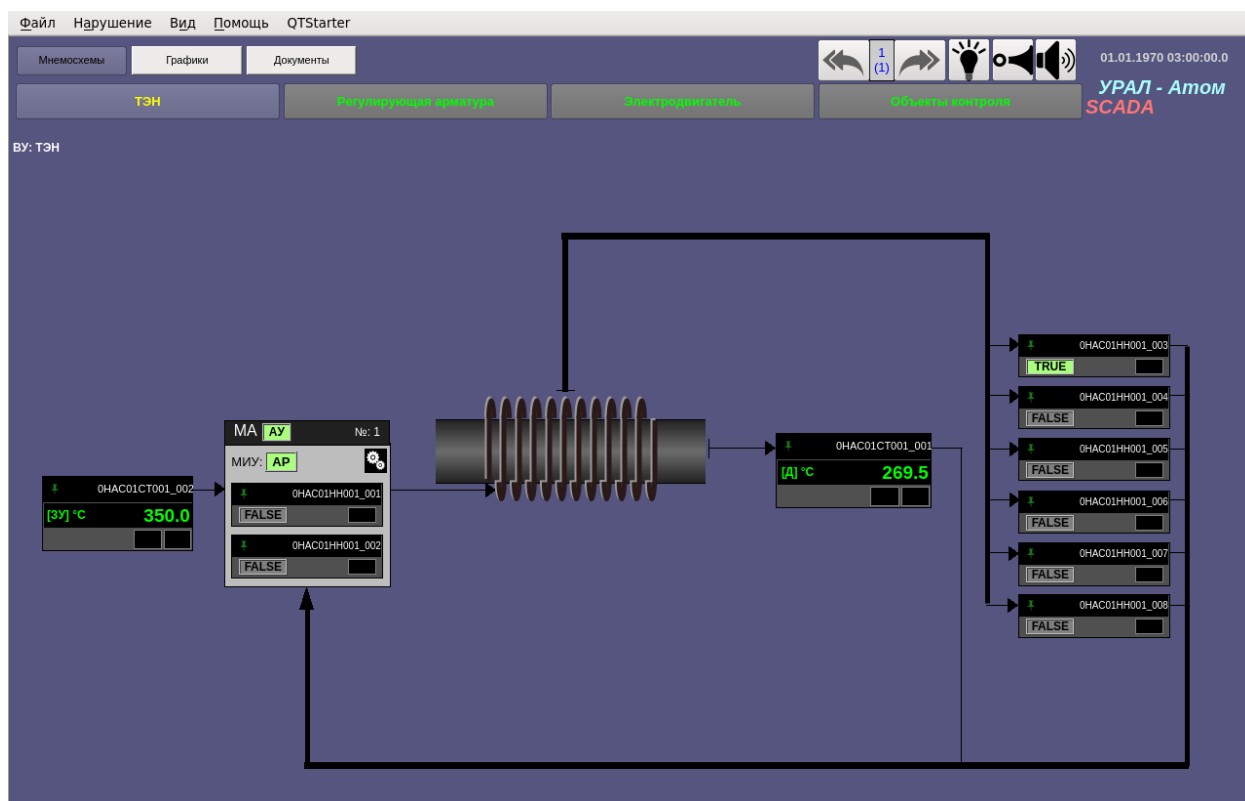


Рисунок В.11 – Недостоверное значение датчика температуры, режим блокировки



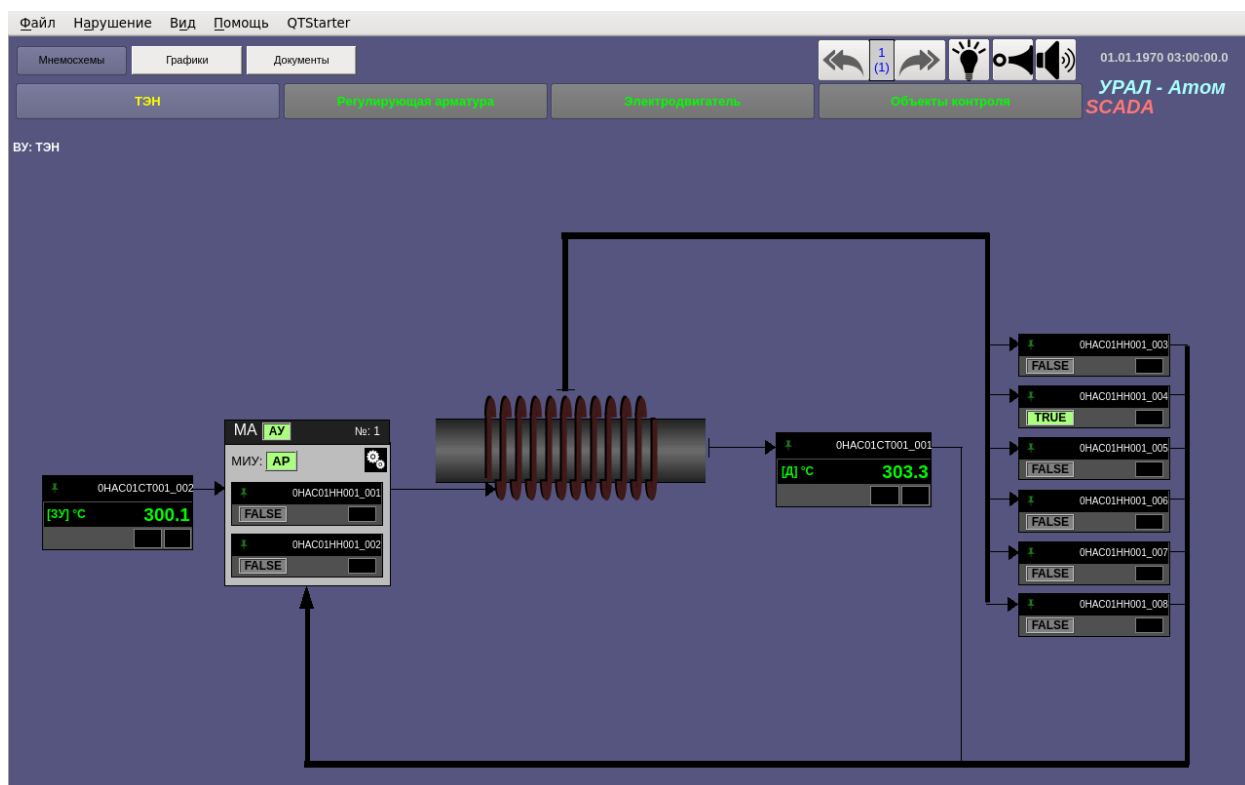


Рисунок В.13 – Датчик температуры больше задатчика температуры, режим автоматического регулирования

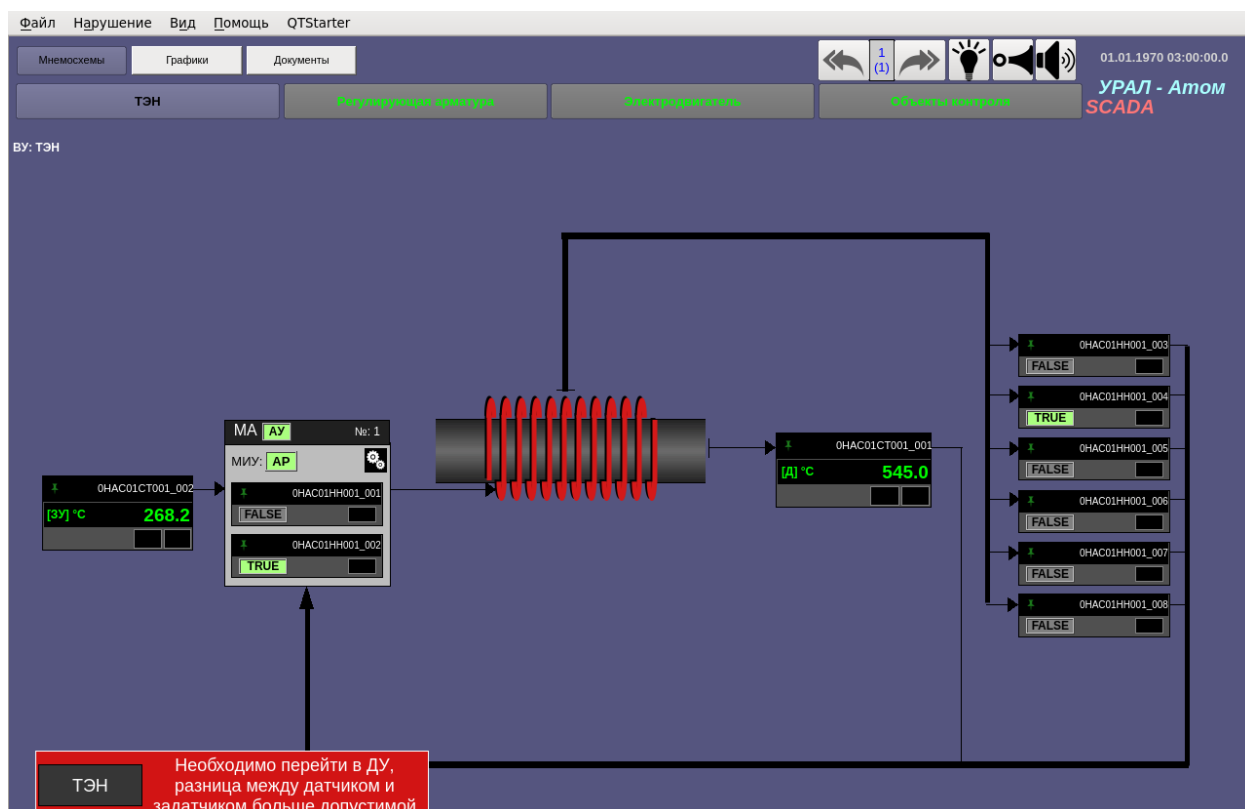


Рисунок В.14 – Разница между значениями датчика и задатчика температуры больше допустимой, режим автоматического регулирования

Приложение Г
АКТ О ВНЕДРЕНИИ РЕЗУЛЬТАТОВ
(рекомендуемое)

АКТ
о внедрении результатов
выпускной квалификационной работы
Сорокиной Надежды Викторовны

Модель и алгоритм поддержки адаптивного управления электрообогревом технологического оборудования атомной электростанции, разработанные инженером-программистом 3 категории НТЦ-1300 Сорокиной Н.В. – сотрудником АО «НПП «Рубин», в ходе выполнения выпускной квалификационной работы, включены в состав программного обеспечения опытного образца программно-технического комплекса для создания и модернизации систем контроля и управления технологическим оборудованием ПТК «УРАЛ-Атом» и активно используются при проведении его комплексной наладки, испытаний и эксплуатации.

Заместитель генерального
директора по науке

Главный конструктор
ОКР ПТК «УРАЛ-Атом»



И. В. Сериков

П. Ю. Кочегаров