Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технологический университет «СТАНКИН» (ФГБОУ ВО «МГТУ «СТАНКИН»)

На правах рукописи

Нежметдинов Рамиль Амирович

ПРИНЦИПЫ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ

Специальность 05.13.06 - «Автоматизация и управление технологическими процессами и производствами (технические системы)»

ДИССЕРТАЦИЯ
на соискание ученой степени доктора технических наук

Научный консультант:
д.т.н., профессор
Мартинов Георги Мартинов

Москва – 2019 г.

Оглавление

Введение

Актуальность темы исследования. Сегодняшний этап развития общества относят к постиндустриальному, основу которого составляет информация как средство и объект производства. В этих условиях изменились средства сбора, обработки и передачи информации: все большее количество людей для работы с информационными ресурсами используют мобильные устройства, а для доступа к большим объемам данных применяют глобальную сеть и облачные технологии. Эти изменения отразились и на промышленных технологиях, в которых произошел переход от концепции, направленной на автоматизацию отдельных машин и процессов к концепции, предусматривающей цифровое представление всех физических активов с последующей интеграцией в цифровую глобальную систему, выстроенную совместно с партнерами, участвующими в цепочке создания стоимости. В основе новой концепции лежит многоуровневая, сложная, глобальная технологическая и организационная система, которая подразумевает интеграцию в единое информационное пространство физических операций и сопровождающих их процессов. Эффект от её внедрения возможен в том случае, если структурирован процесс получения, анализа и обмена данными между технологическим оборудованием и различными уровнями производственной системы в целом. При этом на промышленных предприятиях происходит переход от классической модели производственных систем к «умным» или интеллектуальным производствам (англ. smart manufacturing), предусматривающим объединение производства в единую цифровую экосистему. Развитием идеи интеллектуальных производств является концепция цифровых платформ, которая предполагает наличие взаимосвязанных программных и аппаратных средств, позволяющих сократить циклы производства товара и срок вывода новых продуктов на рынок. При этом на цеховом уровне указанные изменения напрямую затрагивают системы логического управления, которые должны иметь механизмы взаимодействия с цифровой экосистемой производства. Под термином «системы логического управления» будем понимать совокупность систем управления (программируемые логические контроллеры - ПЛК, контроллеры автоматизации, системы управления безопасностью и управления движением), решающие логические задачи управления в технологических системах. Указанные системы относятся к типу систем программного управления и в работе рассматривается вариант их применения для управления технологическим оборудованием (рисунок В.1).

В промышленной среде появилась необходимость применения технологии анализа больших объемов данных (англ. «Big data»), которая предполагает использование большого количества датчиков, установленных на ключевых узлах объектов управления, при этом сбор и предварительную обработку данных с датчиков выполняют системы логического управления. Анализ

информации, полученной указанным путем, представляет объективные и точные данные о работе конкретного оборудования и предприятия в целом. Применение большого количества датчиков используют для получения цифрового образа конкретного физического изделия в процессе его эксплуатации, что является основой концепции «цифровых двойников». Анализ «цифровых двойников» позволяет корректировать как характеристики готовых изделий в ходе их работы, так и технологический процесс их изготовления.

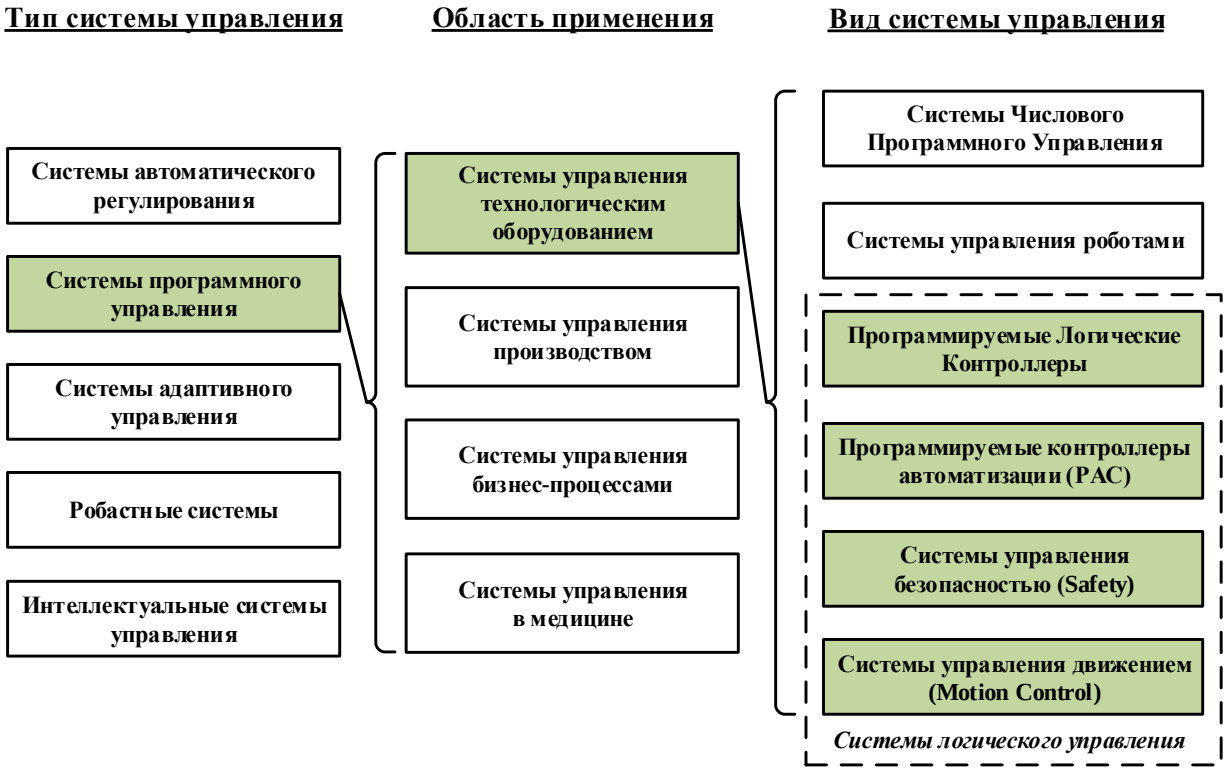


Рисунок В.1 Место систем логического управления в классификации систем управления

В связи с наличием кризисных явлений в экономике все большее внимание при проектировании оборудования уделяют вопросам энергоэффективности. В этом случае системы логического управлении объектами должны обеспечивать наличие режима ожидания (англ. standby) с уменьшенным потреблением энергии, переход в который осуществляется при простое энергоемкого оборудования.

Учитывая перечисленные факторы, проблема построения устройств логического управления, обеспечивающих согласованную работу механизмов и агрегатов, является одной из важнейших при решении задач автоматизации производственных процессов в различных отраслях промышленности, в первую очередь в машиностроении и станкостроении. На большинстве предприятий Российской Федерации на сегодняшний момент в качестве основного звена автоматизации используются классические ПЛК, называемые также внешне реализованными контроллерами, которые показали свою надёжность и широкий спектр применения. В то же время, на мировом рынке промышленной автоматизации наметился ряд тенденций в области развития ПЛК, среди

которых можно выделить следующие: применение высокоскоростных протоколов связи на базе технологии Ethernet для взаимодействия с системами управления и исполнительными устройствами в рамках цифровой экосистемы предприятия; реализация распределенного управления на базе многоранговых сетей, работающих по принципу «ведущий-ведомый» (англ. «master-slave»), для обеспечения согласованной работы разнородного оборудования в рамках единого технологического процесса; использование в качестве системного программного обеспечения операционных систем реального времени (ОСРВ), что позволяет реализовать систему логического управления на базе вычислительной платформы общего назначения; применение технологий анализа больших объемов данных в связи постоянным увеличением объемов информации получаемой с объекта управления; применение для связи с устройствами автоматизации единого интерфейса взаимодействия на базе технологии OPC, что позволяет реализовать коммуникационный канал между устройствами различных производителей; интеллектуализация периферийных модулей, в частности модулей ввода/вывода, что позволяет производить обработку информации непосредственно на объекте управления и др.

По оценкам аналитиков компании HIS Markit [1], объем мирового рынка программируемых логических контроллеров в 2017 году составил 8,5 млрд. $ и ожидается, что к 2021 году он превысит 9,5 млрд. $. На сегодняшний день в области систем промышленной автоматизации в целом и в области ПЛК в частности, Россия идет по пути «догоняющего развития» (англ. catch-up growth), следствием чего является отсутствие в стране производства контроллеров мирового уровня на собственной элементной базе и программном обеспечении отечественной разработки. Российские разработчики контроллеров «ОВЕН», «Fastwell», «Робокон» и др. в качестве аппаратной базы используют электронные компоненты зарубежного производства, часть из которых является OEM (original equipment manufacturer) решениями, а в качестве среды программирования применяются коммерческие пакеты, разработанные иностранными компаниями (например, CoDeSys). Такой подход не позволяет осуществлять контроль над функциями программного обеспечения и накладывает ограничения при выборе аппаратных комплектующих и по большей части не может считаться отечественным решением.

В свою очередь западные компании развивают решения в области автоматизации на новой программно-аппаратной базе. Среди наиболее перспективных решений стоит выделить следующие: PAC-системы (Programmable Automation Controller) – контроллеры на базе персональных компьютеров (ПК) промышленного исполнения, программно реализованные контроллеры (в западной терминологии Soft PLC) – программное обеспечение, устанавливаемое на вычислительную платформу (например, ПК) и на системном уровне реализующее функции контроллера и системы управления движением (в западной терминологии Motion Control). Указанный класс си-

стем позволяет с меньшими экономическими, технологическими и временными издержками реализовывать проекты по управлению сложным технологическим оборудованием и агрегатами. При этом основные алгоритмы и механизмы функционирования систем заложены в программном ядре, что позволяет российским производителям вывести разработки на современный уровень и избавиться от импортозависимости в основных вопросах проектирования и разработки. Успешность в значительной степени будет зависеть от правильности выбора архитектурной концепции и используемых технологий.

Системы управления ПЛК, РАС системы, контроллеры безопасности и контроллеры движения развивались самостоятельно, без взаимной связи между собой. Они имеют традиции, которые сформировали обособленные принципы построения каждой из этих систем управления. Для их проектирования используют разные: аппаратные решения, системное программное обеспечение (ПО), инструментарий проектирования и др. До недавнего времени указанные типы систем отвечали за реализацию узкоспециализированных задач, сегодня с развитием вычислительной техники и технологий каждый из них позволяет реализовывать в том числе не традиционные для себя функции (например, ПЛК с функцией управления движением). Все большее расширение функциональных возможностей систем позволяет применять единое решение при проектировании и реализации логической задачи управления вне зависимсоти от типа, применяемого оборудвоания. Такой подход наиболее актуален для нетривиальных задач, к которым можно отнести проектирование электроавтоматики сложных станков-комбайнов или больших технологических комплексов. (рисунок В.2) Системы логического управления применяются для работы со всеми видами технологического оборудования, используемого на машиностроительных предприятиях.

В Российской Федерации действуют государственные программы, которые предполагают проектирование и реализацию оборудования отечественного производства, такие как: цифровая экономика Российской Федерации (до 2024 года), развитие науки и технологий (до 31.12.2020), развития вооружений (до 31.12.2020), развитие авиационной промышленности (до 31.12.2025), развитие промышленности и повышение её конкурентоспособности (до 31.12.2020). В этой связи остро стоит проблема реализации систем управления независимых от импортных программных и аппаратных комплектующих.

Чтобы реагировать на перечисленные вызовы и с учетом тенденции развития современных промышленных предприятий, направленной на реализацию единой цифровой экосистемы с поддержкой концепций: цифровых двойников и платформ, работы с «большими данными», энергосбережения и др., требуется переосмысление логической задачи управления. Необходима новая концепция построения систем логического управления, которая должна базироваться на мо-

делях, учитывающих: применение высокоскоростных протоколов связи, реализацию распределенного управления на базе многоранговых сетей, использование в качестве системного программного обеспечения ОСРВ, применение единого интерфейса взаимодействия на базе технологии OPC, интеллектуализация периферийных модулей и др., что обуславливает актуальность диссертационной работы.

Рисунок В.2 Область применения систем логического управления

Степень разработанности темы исследования. Вопросы моделирования, проектирования, разработки, программирования и эксплуатации систем логического управления рассмотрены в работах таких российских и зарубежных ученых, как: Сосонкин В.Л. [2-11], Соломенцев Ю.М. [10-12], Мартинов Г.М. [4, 6-9, 13-14], Юдицкий С.А. [15-20], Гаврилов М.А. [21-22], Грейнер Г.Р. [23], Петров И.В. [24-25], Аршанский М.М. [26], Титаренко Ю.И. [27], Дубинин В.Н. [28], G. Olsson [29], G. Piani [29], Rullan A. [30], E.A. Parr [31], Frank D. Petruzella [32-33], Quan Liang [34-35].

В научных работах, выполненных до конца прошлого века, в качестве основной аппаратной единицы систем логического управления рассматривались ПЛК. Были предложены стандартные архитектурные решения для построения указанного типа систем, методы выбора программных и аппаратных компонент. В области программирования систем логического управления были предложены методы, основанные на применении математических аппаратов сетей

Петри, автоматных моделей, операторных формул и др. Однако, на рубеже веков были выдвинуты новые идеи построения систем логического управления, основанные на глубокой проработке математического обеспечения ядра системы управления и применении аппаратной платформы персональных компьютеров. Зарубежными учеными и разработчиками были предложены новые формы организации систем, среди которых можно выделить программно реализованные [30-33] и PAC контроллеры [30, 34, 35]. В то же время эти изменения пришлись на период, сопровождавшийся в России кризисом в реальном секторе экономики. В связи с этим комплексных исследований по вопросам построения систем логического управления в стране почти не проводилось, а те результаты, которые были получены, касались отдельных узких областей применения систем управления [27, 28, 36-40].

Объект исследования. Системы логического управления технологическим оборудованием.

Предмет исследования. Модели, методы, алгоритмы и программно-аппаратное обеспечение, используемое при разработке систем логического управления технологическим оборудованием.

Цель и задачи работы. В работе поставлена следующая **цель** - повышение эффективности функционирования систем логического управления технологическим оборудованием за счет формализации и инструментальной поддержки процессов их проектирования и программной реализации.

Для достижения цели работы должны быть решены следующие задачи:

- Проанализировать современные системы логического управления, выявить тенденции и перспективы их развития;
- Разработать теоретические основы для описания моделей построения программных систем логического управления технологическим оборудованием;
- Создать формальный аппарат построения подсистемы программирования и исполнительного ядра системы логического управления технологическим оборудованием;
- Разработать методологические основы построения систем логического управления технологическим оборудованием;
- На базе теоретических и методологических основ построения систем логического управления разработать практические аспекты реализации систем логического управления конкретным технологическим оборудованием.

Научная новизна работы

1. Установлены взаимосвязи между характеристиками технологического оборудования и задачами, функциями, параметрами систем логического управления технологическим оборудованием, влияющими на структуру системы управления и определяющими состав программно-аппаратных модулей логических контроллеров, как основного инструментария автоматизации.

2. На основе установленных взаимосвязей разработаны модели систем логического управления, которые составили теоретические основы проектирования систем логического управления, предполагающую последовательную трансформацию моделей для получения полного описания системы.

3. Разработаны теоретические основы построения исполнительного ядра систем логического управления технологическим оборудованием как системы реального времени.

4. Разработаны теоретические основы построения среды программирования систем логического управления технологическим оборудованием, соответствующей требованиям стандарта МЭК 61131-3.

5. Разработан методологический базис построения современных систем логического управления, соответствующий требованиям предъявляемым международными стандартами, обеспечивающий возможность применения: аппаратных модулей ввода/вывода использующих промышленные протоколы связи, стандартных средств операционных систем, стандартных и оригинальных инструментальных средств реализации программ логического управления.

6. Предложена методология построения систем логического управления технологическим оборудованием, в том числе:

- формализован процесс выбора аппаратных средств проектирования систем логического управления;

- систематизированы математические методы проектирования программ логического управления и предложены способы проектирования программ логического управления на его основе;

- предложена методика построения систем управления, которая в качестве входных данных использует техническое задание на разработку системы и предполагает прохождение последовательности шагов для получения готовой системы логического управления конкретным технологическим оборудованием.

Теоретическая значимость работы заключается в:

- установленных в результате анализа взаимосвязях между характеристиками технологического оборудования и задачами, функциями, параметрами систем логического управления технологическим оборудованием;
- разработанных теоретических основах построения исполнительного ядра и среды программирования систем логического управления технологическим оборудованием;
- предложенной совокупности моделей построения систем логического управления технологическим оборудованием;
- систематизации и обосновании математических методов проектирования программ логического управления.

Практическая значимость работы заключается в:

- программной реализации исполнительного ядра системы логического управления технологическим оборудованием;
- программной реализации подсистемы программирования системы логического управления технологическим оборудованием;
- разработке методики проектирования систем логического управления технологическим оборудованием, которая позволяет:

– производить адаптацию моделей и настройку систем логического управления под конкретный технологический объект;

– использовать готовые программные и аппаратные компоненты системы управления, имеющие стандартные интерфейсы подключения;

– получать систему управления, обладающую новыми качественными и количественными характеристиками, за счет конфигурирования программно-аппаратных компонент;

– использовать готовые библиотеки программ логического управления для поддержки функций отдельных узлов технологического оборудования.

- в разработке специализированных систем логического управления электроавтоматикой для: установки гидроабразивной резки, гаммы токарно-фрезерных высокоточных обрабатывающих центров с числовым программным управлением наклонной компоновки, вертикально-фрезерного обрабатывающего центра.

Методология и методы исследований. Теоретические исследования выполнялись с использованием математического аппарата булевой алгебры, автоматных моделей, теории графов, аппарата разностных уравнений, теории автоматического управления (теории регуляторов), теории алгоритмов и линейной алгебры. Проектирование программных компонент осуществлялось с использованием объектно-ориентированного подхода. Лабораторные испытания проводились на специально разработанных стендах с применением современных средств автоматизации. Экспериментальные исследования проводились в производственных условиях на современном технологическом оборудовании. Обработка результатов теоретических и практических исследований осуществлялась с использованием программных инструментов обработки информации.

Положения, выносимые на защиту:

1. Результаты комплексного анализа вопросов построения систем логического управления технологическим оборудованием.
2. Совокупность моделей построения систем логического управления технологическим оборудованием.

3. Формальный аппарат построения исполнительного ядра и подсистемы программирования системы логического управления.

4. Методика проектирования систем логического управления технологическим оборудованием.

5. Результаты практической разработки специализированных систем логического управления конкретным технологическим оборудованием.

Степень достоверности и апробация результатов. Достоверность результатов, полученных в ходе работы над диссертацией подтверждается применением системного подхода к решению поставленных задач, корректностью методов, применяемых для теоретических и экспериментальных исследований. Теоретические аспекты работы строятся на результатах, прошедших проверку и согласуются с имеющимися в открытом доступе экспериментальными данными по теме диссертации. Сбор, обработка, анализ и интерпретация экспериментальных данных проведена с применением современных методов статистического анализа и обработки информации.

Основные положения и результаты работы докладывались на международных и всероссийских конференциях: «Мехатроника, автоматизация, управление» (МАУ-2010) (г. Санкт-Петербург, 2010 г.); II Международная Интернет-конференция молодых ученых, аспирантов и студентов «Инновационные технологии: теория, инструменты, практика» (InnoTech 2010); X - XVII международные конференции "Системы проектирования технологической подготовки производства и управления этапами жизненного цикла промышленного продукта (CAD/CAM/PDM)" (ИПУ РАН, г. Москва, 2010-2018 г.г.); всероссийская научно-практическая конференция с элементами научной школы для молодежи "Проведение научных исследований в области информационно-телекоммуникационных технологий" (ВВЦ, г. Москва, 2010 г.); III научно-образовательная конференция "Машиностроение – традиции и инновации" (МТИ-2010) (ГОУ ВПО МГТУ "Станкин", г. Москва, 2010 г.); Международная научно-практическая конференция "Передовые информационные технологии, средства и автоматизации и их внедрение на российских предприятиях" AITA 2011 (ИПУ РАН, г. Москва, 2011 г.); III Международная научно-техническая конференция "Модернизация машиностроительного комплекса России на научных основах технологии машиностроения" ТМ-2011 (Брянский государственный технический университет, г. Брянск, 2011 г.); Всероссийская научная школа "Современные технические средства диагностики металлорежущих станков" (МГТУ им. Баумана, г. Москва, 2011 г.); IV Всероссийская мультиконференция по проблемам управления (МКПУ - 2011) (с. Дивноморское, Краснодарский край, 2011г.); XIX Международная научно-техническая конференция "Информационные средства и технологии" (г. Москва, 2011г.); Всероссийская молодежная конференция "Инновационные технологи в машиностроении" (ИТМ - 2011) (ФГБОУ ВПО МГТУ "СТАНКИН", г. Москва 25-26 октября 2011 г.); XII

Всероссийское совещание по проблемам управления (ИПУ РАН, г. Москва, 2014 г.); XI Всероссийская научно-практическая конференция "Современные информационные технологии в науке, образовании и практике" (Оренбургский государственный университет, г. Оренбург, 2014 г.); VII Международная научно-образовательная конференция "Машиностроение - традиции и инновации" (МТИ - 2014) (МГТУ «СТАНКИН», г. Москва, 2014 г.).

Работа выполнялась на кафедре компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН». Результаты использованы в рамках:

- работ по ФЦП «Научные и педагогические кадры инновационной России» (Минобрнауки РФ) по государственным контрактам: П-1313 от 09.06.2010 «Создание распределенных вычислительных систем управления производственно-технологическим оборудованием на базе логических контроллеров и контроллеров автоматизации» (руководитель проекта), П-1368 от 11.06.2010 «Обработка, хранение, передача и защита технологической информации в распределенных системах управления» (руководитель проекта), 16.740.11.0228 от 22.09.2010 «Распределенная компьютерная система управления механолазерным комплексом прецизионной обработки», 02.740.11.0488 от 18.11.2009 «Создание гетерогенной распределенной компьютерной системы для управления в реальном времени децентрализованными высокотехнологичными производствами, объединенными в виртуальные корпорации» (исполнитель в проекте);
- выполнения государственных контрактов по заданию Минобрнауки РФ: № 1357 «Методы, модели и алгоритмы построения компьютерных систем управления для цифрового производства» (исполнитель в проекте), 2.1237.2017/ПЧ «Исследование и разработка высокопроизводительных распределенных систем управления на базе микропроцессоров «Эльбрус» для цифровых производств» (исполнитель в проекте);
- выполнения работ по грантам Президента РФ в поддержку молодых ученых кандидатов наук по государственным контрактам: 16.120.11.323-МК от 18.02.2011 «Создание кроссплатформенного программно реализованного логического контроллера для управления технологическим оборудованием» (руководитель проекта), 14.124.13.4353-МК от 04.02.2013 «Организация управления децентрализованными производствами с применением программно реализованных средств электроавтоматики» (руководитель проекта);
- работ по ФЦП "Национальная технологическая база" (Минпромторг РФ) в рамках государственных контрактов: 9411.1003702.05.010 от 23.09.09 при создании установки гидроабразивной резки (совместно с ОАО НИАТ и ООО «Савеловский машиностроительный завод», исполнитель в проекте), 252и/60М от 29.09.2011 при создании гаммы токарно-фрезерных высокоточных обрабатывающих центров с числовым программным управлением наклонной компоновки (совместно с ОАО «САСТА», исполнитель в проекте);

- работ по договору 14-45/х от 10.07.2014 с ОАО «Ковровский электромеханический завод» по разработке электрооборудования и изготовлении комплектной системы ЧПУ для оснащения станка MV184P/15C (исполнитель в проекте).

Результаты работы в виде программ и методических материалов внедрены в учебный процесс кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» в рамках освоения дисциплин:

- «Автоматика и управление движением», «Автоматные модели в системах управления» подготовки бакалавров по направлению 15.03.04 «Автоматизация технологических процессов и производств»;
- «Программируемые логические контроллеры в системах управления» подготовки магистров по направлению 15.04.04 «Автоматизация технологических процессов и производств»;
- «Информационные системы в автоматизированном производстве» подготовки аспирантов по научной специальности 05.13.06 «Автоматизация и управление технологическими процессами и производствами».

Соответствие диссертации паспорту научной специальности. Содержание материалов диссертационной работы соответствует пунктам 3, 5 и 15 раздела «Области исследований» паспорта специальности 05.13.06 «Автоматизация и управление технологическими процессами и производствами».

Публикации. По теме диссертационной работы опубликовано 91 печатных работы, из которых: 34 статья в журналах из перечня научных рецензируемых журналов ВАК РФ, 7 печатных работ, входящих в реферативные базы Web of Science и SCOPUS, 39 материалов конференций, 11 свидетельств о регистрации программ для ЭВМ, 1 патент на полезную модель.

Структура и объем работы. Диссертация состоит из введения, пяти глав, заключения, списка сокращений, списка использованной литературы и приложений. Работа изложена на 251 страницах машинописного текста, содержит 87 рисунков, 28 таблиц, 2 приложения. Список литературы содержит 206 наименований.

Глава 1 Анализ современных систем логического управления. Выявление тенденций и перспектив развития

Традиционно контроллеры были предназначены для выполнения простых последовательных операций с двоичными сигналами, отсюда и пошло их название – логические. Современные логические контроллеры помимо этого выполняют и другие операции, например, для решения задачи управления цикловой электроавтоматикой совмещают функции счётчика и интервального таймера, обрабатывают задержку сигнала и т.д. При этом в их работе выделяют три фазы управления. Первая фаза содержит ввод программы логического управления, конфигурирование и настройку программно-аппаратных компонент и выполняется в машинном времени. Вторая фаза содержит вычисления в реальном времени и производится в ядре логического управления. В рамках этих вычислений осуществляется: обработка данных, поступивших с технологического оборудования, исполнение алгоритмов управления электроавтоматикой, формирование массива управляющих воздействий для передачи на технологическое оборудование. К третьей фазе относится получение данных с аппаратных модулей ввода и передача управляющих воздействий на аппаратные модули вывода. В зависимости от состава аппаратного обеспечения вычислительного ядра и объема задач, решаемых на каждой из фаз управления, формируется состав конкретной системы логического управления. Адаптация системы к конкретному объекту управления осуществляется за счет реализации программ логического управления.

Анализ современных тенденций в области построения систем логического управления (рисунке 1.1) показывает, что за последние годы существенно изменилась их: архитектура, потребительские свойства, аппаратное и программно-математическое обеспечение. К причинам изменений можно отнести следующие: ориентация на концепцию цифрового производства (распределенное управление, высокоскоростные протоколы связи, интеллектуализация входов/выходов, удаленная диагностика и др.); появление новых типов аппаратных устройств вычислительной техники (одноплатные компьютеры, микроЭВМ, мобильные устройства и др.); появление общедоступных ОСРВ, в том числе и для установки на мобильные платформы (Linux RT, Windows Embedded и др.); развитие свободно распространяемых и коммерческих программных библиотек, которые могут быть использованы при реализации систем управления. Представленные изменения требуют пересмотра методологической базы проектирования систем логического управления. При этом опыт, накопленный в проектировании, программировании и эксплуатации систем остается актуальным для автономных систем, реализованных на базе ПЛК.

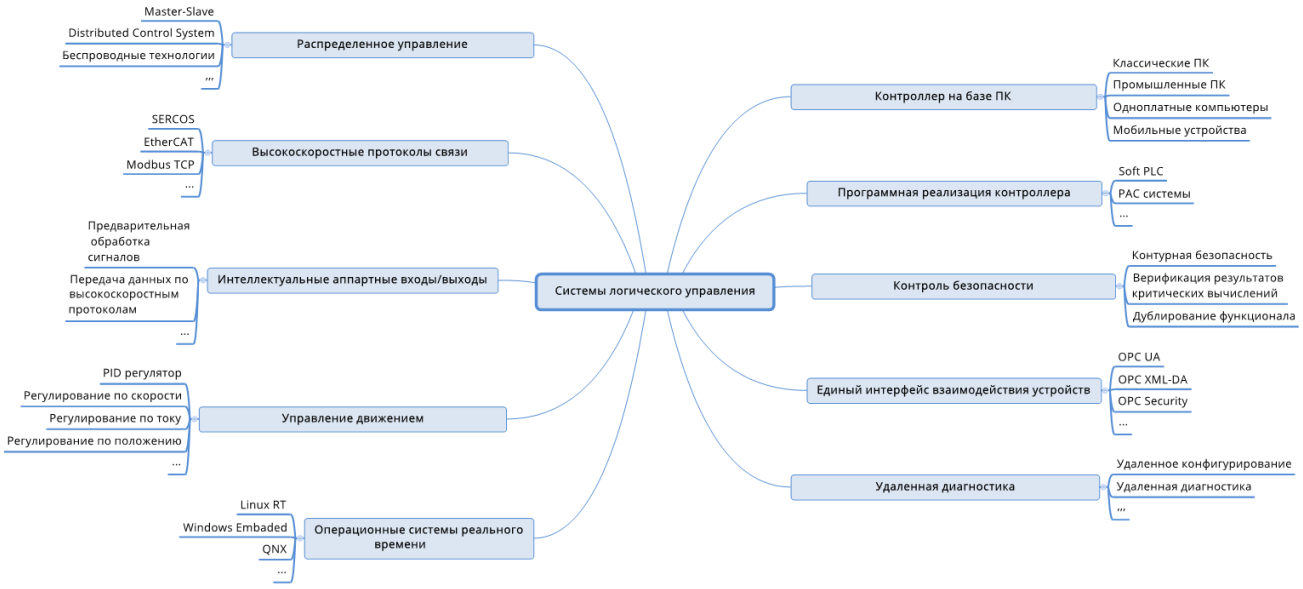


Рисунок 1.1 Интеллект-карта ключевых направлений развития систем логического управления

1.1 Накопленный научным сообществом опыт в реализации систем логического управления технологическим оборудованием

Развитие дискретной автоматики в нынешнем её виде началось с изобретения электромагнитного реле. Их использование расширилось в 50-е годы прошлого века, чему способствовал возросший спрос на автоматизацию различных промышленных процессов. Успехи технологии привели к созданию надежных малогабаритных реле, а также развитию теории синтеза контактных логических сетей. В нашей стране теорией и практикой релейно-контактных схем автоматизации занимались Гаврилов М.А., Копыленко В.М., Грейнер Г.Р. [21-23]. На сегодняшний день каждая из систем автоматического управления имеет в своем составе одно из разновидностей реле. На основе изображения релейно-контактных схем был создан графический язык программирования ПЛК, получивший широкое распространение и вошедший в стандарт 61131-3 Международной Электротехнической комиссии (МЭК) и имеющий название LD (ladder diagram, русская аббревиатура РКС – язык релейно-контактных схем). [24-25]

Большой вклад в развитие теории релейно-контактных схем внес М.А. Гаврилов. Он создал научную методику проектирования релейно-контактных схем на основе математического аппарата Булейвой алгебры, им была предложена общая теория анализа и синтеза одно- и многотактных РКС. В качестве инструмента реализации был предложен язык «таблиц включений», по которым можно получить структурную формулу многоконтактных схем. Разработана методика преобразования параллельно-последовательных и «мостиковых» схем (схем «Н»), как с ре-

лейно-контактными элементами общего вида, так и со специализированными элементами (искателями, поляризованными и амплитудными реле). Результаты этих исследований М. А. Гаврилов подытожил в своей монографии «Теория релейно-контактных схем» (1950 г.).

Развил теорию синтеза релейно-контактных схем Г.Р. Грейнер в своей работе 1972 года «Проектирование бесконтактных управляющих логических устройств промышленной автоматики» [23]. Основное внимание в работе было уделено вопросам проектирования бесконтактных управляющих логических устройств промышленной автоматики. Были развиты математические основы теории проектирования логических устройств и инженерные методы их синтеза. Предложены основы унификации функциональных узлов и блоков. Были разработаны методы построения контролирующих и диагностических тестов и их применение при наладке и техническом обслуживании систем логического управления. Уделено внимание расчетам надежности и экономической эффективности бесконтактных логических устройств.

В системах управления, выполненных на основе реле или микросхем, невозможно изменить логику работы, не подвергнув их существенному изменению. Со временем развития технологий встал вопрос о создании устройства, которое можно было бы перепрограммировать, не затрачивая больших усилий и не меняя аппаратной базы. Такими устройствами стали ПЛК, появившиеся в 1968 году. ПЛК на основе микропроцессора был впервые создан в США в 1977 году компанией Allan-Bradley Corporation. [41, 42]

Новая аппаратная платформа требовала пересмотра теоретических основ построения систем логического управления. В этом отношении большой вклад в теорию синтеза систем логического управления внес С.А. Юдицкий. В работе «Пневматические системы управления приводом машин-автоматов» были положены основы синтеза систем логического управления применительно к пневматическим приводам машин-автоматов. Рассмотрены вопросы проектирования систем управления машин-автоматов, рабочие органы которых циклически, в заранее известной последовательности, перемещаются в пространстве. С.А. Юдицким были предложены способы описания условий работы машин-автоматов, позволяющие стандартным образом сформулировать задание при проектировании системы логического управления. На основе анализа структуры системы логического управления разработаны методы их инженерного синтеза.

В диссертационной работе С.А. Юдицкого «Разработка принципов и методов построения устройств логического управления дискретными технологическими процессами на основе сетей Петри» предложены основы теории проектирования систем логического управления на базе математического аппарата сетей Петри. Изначально специализированный математический аппарат на базе сетей был предложен Карлом Адамом Петри в 1962 году в рамках диссертации

«Kommunikation mit Automaten» (нем., взаимодействие с автоматами). Труд К.А. Петри стал классическим и позволил развить теорию параллельных и распределённых вычислений и послужил основой при построении сложных систем и потоков работ.

Благодаря продуктивному развитию средств сетевой интеграции появилась возможность создания распределенных систем управления. В 80-х гг. XX в. доминировали ПЛК с числом входов-выходов в несколько сотен. В настоящее время большим спросом пользуются микро ПЛК с количеством входов-выходов до 64. В распределенных системах каждый ПЛК решает локальную задачу. Задача синхронизации управления выполняется компьютерами среднего звена автоматизированных систем управления. Распределенные системы выигрывают по надежности, гибкости монтажа и простоте обслуживания [43-45].

Построение систем логического управления в рамках машиностроительных производств имеют свои особенности, в частности, это согласованная работа контроллеров в рамках систем ЧПУ, в качестве головного устройства в гибких производственных модулях (ГПМ), ячейках (ГПЯ) и системах (ГПС). Развитием теории проектирования систем логического управления в машиностроении занимались Соломенцев Ю.М., Сосонкин В.Л., Аршанский М.М., а позднее и Мартинов Г.М.

В работе «Программное управление станками», переведенной на многие языки мира и получившей заслуженную популярность, Сосонкиным В.Л. была предложена классификация задач в рамках систем ЧПУ, среди которых была выделена логическая задача. Логическая задача определялась как задача автоматизации «большого числа многообразных вспомогательных простых или циклических операций: зажимы-разжимы, подводы-отводы, переключения, пуски-остановы, автоматическая смена инструмента и др.» [202] Основным объектом управления для логической задачи ЧПУ была определена цикловая электроавтоматика станка, под которой понимают «систему автоматического управления механизмами и группами механизмов, поведение которых определяется множеством дискретных операций с отношениями следования и параллелизма.» [202] Для решения логической задачи ЧПУ было предложено применение математического аппарата автоматных моделей, приведены практические аспекты реализации автоматической сменой инструмента. Для описания циклов и операций автоматики в работе было предложено применение формализма сетей Петри. Развитие теории построения систем управления в машиностроении получило в последующих работах В.Л. Сосонкина - «Микропроцессорные системы числового программного управления станками» [203] и «Программное управление технологическим оборудованием» [205].

С появлением систем числового программного управления (ЧПУ) класса PCNC (Personal Computer Numerical Control) появились ПЛК для управления электроавтоматикой станка в одно-

платном исполнении. Сетевой контроллер электроавтоматики при таком исполнении представляет собой электронное устройство, снабженное собственным микропроцессором и встроенной ОС РВ [117, 131, 132].

Теоретические основы проектирования гибких производственных систем и применение в рамках гибкого производства контроллеров было раскрыто в совместной работе Соломенцева Ю.М. и Сосонкина В.Л. «Управление гибкими производственными системами» [204]. В работе предложен новый подход к автоматическому управлению сильно неопределенными объектами гибкого производства. Доказано, что проект системы управления зависит от параметров и характеристик объекта управления, в качестве объекта управления выступает технологический процесс. Задачи, функции, способы организации и вопросы программно-математического обеспечения системы логического управления рассмотрены на двух уровнях – гибкого модуля и гибкой системы.

На очередном витке развития технологий Г. Бучем была предложена объектно-ориентированная парадигма проектирования и программирования [206], которая нашла применение и в промышленных системах управления, в том числе и в логическом управлении. Развитием теории построения промышленных систем управления на основе объектно-ориентированной парадигмы на западе занимались A. Rullan [30], E. A. Parr [31], F.D. Petruzella [32], в России Шалыто А.А. [193], Аршанский М.М. [26], Мартинов Г.М. и др.

Мартинов Г.М. развил теорию построения систем ЧПУ, предложенную Сосонкиным В.Л., с использованием объектно-ориентированной парадигмы, что нашло свое отражение в работах «Системы числового программного управления: учебное пособие» [10] и «Программирование систем числового программного управления: учебное пособие» [9]. В указанных работах использован объектный подход к управлению электроавтоматикой, выделены особенности управления электроавтоматикой станков с ЧПУ, предложено использование программно-реализованного контроллера (Soft PLC), встроенного в программно-математическое обеспечение системы ЧПУ.

В промышленности широкое применение также нашли системы управления движением (англ. Motion Control), для проектирования и реализации которых используют классическую теорию автоматического управления и теорию регуляторов.

А для обеспечения безопасной работы применяют противоаварийные системы и системы безопасности, которые в английской литературе называются Safety. Теория построения надежных и безопасных систем базируется на математическом аппарате теории вероятности и математической статистики. Прототипом современных систем безопасности стали релейные схемы защиты, разработкой теории построения которых занимались М.А. Беркович [200], Н.В. Чернобро-

вов [201], А.М. Федосеев [199]. В России вопросами математической теории надежности занимались следующие ученные: Барзилович Е. Ю. [198], Байхельт Ф. [197], Базовский И. [196], Рогов С.Л. [194]

В таблице 1-1 представлены методы построения, применяемые для систем управления, рассматриваемых в работе.

Таблица 1-1. Систематизация методов построения специализированных систем управления

Программируемые логические контроллеры (ПЛК):	**Программируемые контроллеры автоматизации (PAC):**
- булева алгебра (Гаврилов М.А., Грейнер Г.Р.) - автоматные модели (Сосонкин В.Л., Юдицкий С.А.) - теория релейно- контактных схем (Гаврилов М.А., Копыленко В.М., Грейнер Г.Р.); - теория сетей Петри (K. A. Petri, Юдицки С.А.)	- булева алгебра (Гаврилов М.А., Грейнер Г.Р.); - теория сетей Петри (K. A. Petri, Юдицки С.А.); - теория графов (Сосонкин В.Л., Соломенцев Ю.М., Г. Олссон, Д. Пиани); - объектно-ориентированная парадигма (Г. Буч, А.А. Шалыто, Аршанский М.М., Мартинов Г.М., A. Rullan, E. A. Parr, F.D. Petruzella).
Контроллеры движения (Motion Control):	**Контроллеры безопасности (Safety):**
- теория автоматического управления; - теория регуляторов.	- теория вероятности; - математическая статистика; -математическая теория надежности (Барзилович Е. Ю., Байхельт Ф., Базовский И., Рогов С.Л.); - теория релейных схем защиты (М.А. Беркович, Н.В. Чернобровов, А.М. Федосеев); - теория резервирования.

С развитием ПЛК появилась необходимость в стандартизации всех средств проектирования и программирования контроллеров. В рамках МЭК была создана специальная группа технических экспертов по проблемам ПЛК, включая аппаратные средства, монтаж, тестирование, документацию и связь. Первый вариант стандарта был опубликован в 1982 году. Традиции, сложившиеся на протяжении долгих лет развития, оказывают влияние на тенденции построения современных систем логического управления. Особенно велико влияние в области проектирования и реализации программ логического управления, где используют традиционный математический аппарат и стандартные языки прикладного программирования неизменные на протяжении десятилетий [46-50]. На сегодняшний момент системы на базе традиционных ПЛК и контроллеров на базе промышленных компьютеров заняли свою нишу среди средств автоматизации. Согласно ис-

следованиям журнала Control Engineering [51, 52] и агентства Reed Research (рисунки 1.2) большинство респондентов применяют логическое управление для контроля электроавтоматики станков и технологических процессов.

Однако, есть целый ряд областей, в которых количество применяемых систем логического управления растет с каждым годом, среди которых: системы управления движением (Motion Control), системы диагностики, системы безопасности (Safety) и др.

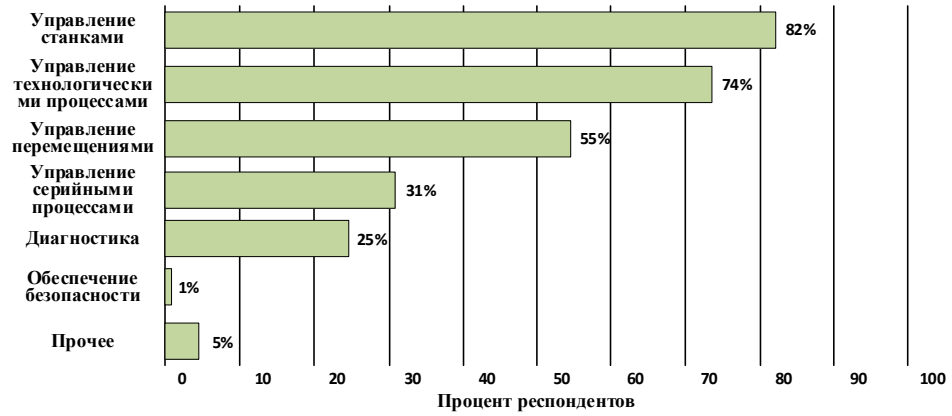


Рисунок 1.2 Применения систем логического управления в машиностроении

1.2 Анализ современных тенденций в области построения систем логического управления технологическим оборудованием

Развитие систем логического управления привело к изменению их организационной структуры, был осуществлен переход от крупных систем на несколько тысяч входов/выходов к локальным узкоспециализированным системам с количеством входов/выходов до нескольких сотен. Этот переход отражают исследования журнала Control Engineering [51, 52] и агентства Reed Research, результаты которых приведены на рисунке 1.3.

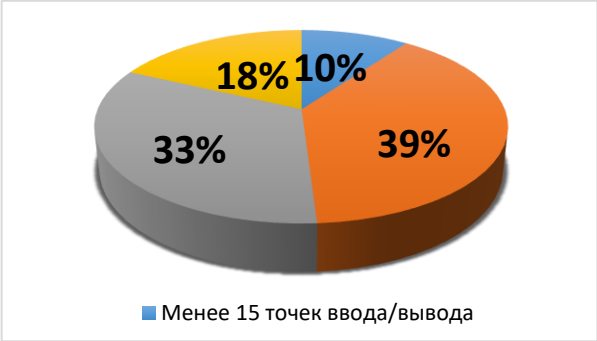


Рисунок 1.3 Количество аппаратных входов/выходов, используемых в системах логического управления

По оценке указанных источников, доля систем с количеством аппаратных модулей входов/выходов более 512 точек до 2000-х годов составляла более половины рынка промышленной

автоматики. Небольшие системы логического управления могут быть использованы как один из узлов автоматизации при внедрении в цифровое машиностроительное производство или для построения распределённых систем управления. При такой форме организации система, кроме основного функционала, должна реализовывать ряд вспомогательных функций, таких как: поддержка высокоскоростных протоколов связи, взаимодействие с остальными узлами автоматизации сети предприятия, в том числе и при многоуровневой организации (например, реализация принципа «ведущий-ведомый»), удаленное конфигурирование, настройка и диагностика, а также удаленная загрузка и отладка программ логического управления и др.

Анализ тенденций развития систем логического управления (рисунок 1.1) показывает, что они направлены на снижение себестоимости конечного продукта, с одновременным повышением степени гибкости и открытости. Выделенные тенденции позволяют сформировать новое представление о системах логического управления (таблица 1-2) и в то же время раскрывают задачи, которые должны быть решены. Далее рассмотрены представленные тенденции.

Таблица 1-2 Систематизация современных тенденций построения систем логического управления

Уровень системы	**Тенденция**	**Пример**	**Открываемые возможности**
Архитектура системы	Распределенный принцип построения систем логического управления		- Возможность использования готовых решений для организации ввода/вывода; - Возможность организации многоуровневой структуры управления.
Прикладное ПО	Программная реализация логического контроллера	Soft PLC	- Реализация СЛУ без привлечения дополнительных аппаратных вычислительных устройств.
	Единый интерфейс взаимодействия устройств на базе OPC	OPC UA	Возможность автоматизированного обмена данными с устройствами, поддерживающими технологию OPC.
	Реализация управления движением	Motion Control	Возможность управления электрическими приводами без привлечения дополнительных программных и аппаратных устройств
	Удаленная диагностика и настройка по Internet		- Сокращение времени на диагностику и настройки систем, находящихся на значительных расстояниях от сервисных служб.
Системное ПО	Применение ОС РВ	Linux RT, Windows Embedded	- Унифицированный стиль программирования и работы оператора; - Использование стандартного инструментария ОС для поддержки функционала системы; - Наличие инструментария и окружения разработки ПО.

Продолжение таблицы 1-2.

Уровень системы	Тенденция	Пример	Открываемые возможности
Аппаратная платформа	Применение вычислительной платформы общего назначения	ПК, одноплатный компьютер, микроЭВМ	- Низкая себестоимость; - Взаимозаменяемость комплектующих; - Постоянное развитие вычислительной базы.
	Применение технологии обеспечения безопасности	Safety	- Унификация программно-аппаратных решениях для обеспечения безопасности; - Аппаратное обеспечение требований к безопасности.
Аппаратная платформа	Применение интеллектуальных аппаратных устройств ввода/вывода		- Предварительная фильтрация и обработка сигналов; - Передача информации по высокоскоростным протоколам.
	Использование высокоскоростных протоколов связи на базе технологии Ethernet	EtherCAT, SERCOS, Modbus TCP	- Унификация аппаратных решений.

Использование в качестве аппаратной базы вычислительных платформ общего назначения. Прогресс в области технологий компьютерной и IT индустрии оказывает прямое влияние на развитие систем логического управления. Изменение форм-факторов и увеличение мощности вычислительных платформ расширяет, в том числе, и возможности систем управления. Программные решения в области построения систем в виде готовых модулей и библиотек на сегодняшний день позволяют превратить любую серийно выпускаемую вычислительную платформу, поддерживающую работу с ОСРВ, в универсальную систему управления. Увеличение мощности вычислительных ресурсов и изменения в программных и аппаратных технологиях происходят постоянно, при этом в новых продуктах осуществлена поддержка функционала предшествующих решений. Внедрение новинок IT-технологий в области систем промышленной автоматизации происходит с серьезными временными задержками. По представленной в каталогах фирм-производителей ПЛК информации можно судить, что тактовая частота самых производительных микропроцессоров ПЛК оценивается десятками МГц, тогда как в современных ПК установлены многоядерные процессоры с тактовой частотой в несколько ГГц. Проблема применения менее производительных микропроцессоров связана со сложностью отвода тепла в промышленных системах, в связи с чем используются элементы с низким коэффициентом потребления энергии. Эти показатели отражают задержку развития аппаратного обеспечения ПЛК по сравнению с ПК в пределах 3-4 лет. Помимо этого, ПК выпускаются массовым тиражом, что удешевляет стоимость каждого изделия, тогда как ПЛК выпускаются ограниченным тиражом (по сравнению с ПК). В этой связи использование в качестве аппаратной платформы вычислительных систем

общего назначения позволит реализовывать продукты, соответствующие передовым решениям в области компьютерных технологий и приведет к снижению себестоимости систем управления. Аппаратное обеспечение ПК также обладает свойством взаимозаменяемости, это позволяет всегда иметь аналоги комплектующих, которые снимаются с производства. При этом, конечные пользователи систем управления будут иметь возможность модернизации (англ. upgrade) и ремонта аппаратной базы без капитальных затрат и изменения системы управления в целом.

Использование программно-реализованного логического контроллера для реализации логической задачи управления. Большинство мировых лидеров в области производства логических контроллеров имеют в списке средств автоматизации программно реализованные логические контроллеры. Эта тенденция напрямую связана с использованием в качестве аппаратной базы вычислительных платформ общего назначения. Для реализации принципа «контроллер на базе компьютера» необходимо иметь аппаратный вычислитель и программное обеспечение. В этом случае программное обеспечение логического контроллера на системном уровне реализует цикл работы контроллера. Среди мировых лидеров в области промышленной автоматизации, которые имеют программно реализованные контроллеры можно выделить: Siemens, с контроллером «S7-1500 Software Controller», Bosch-Rexroth, с продуктом «Soft Control», Rockwell, с контроллером SoftLogix и др.

Использование специализированных аппаратных средств для реализации контроля безопасности. Современные системы автоматического управления должны быть оборудованы средствами противоаварийной защиты (ПАЗ). Указанный тип систем должен быть реализован согласно ГОСТ Р МЭК 61508-1-2007 (международный стандарт IEC 61508-1:1998, Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью). [53] Системы ПАЗ предполагают: желтую маркировку аппаратных средств, наличие устройств прямого размыкания электрических цепей (например, аварийный грибок), дублирование и резервирование критичных процессов и процессов обеспечения безопасности, реализация алгоритмов верификации критичных вычислений, использования специализированного промышленного протокола передачи данных, который предполагает гарантированную доставку пакетов за обозначенный период времени (например, протокол Safety over EtherCAT). В отличии от России, европейское и американское законодательство в части использования систем ПАЗ предъявляет более жесткие требования к разработчикам систем управления, поэтому «Safety» системы западных производителей отличаются более широким функционалом. Среди производителей мирового уровня можно выделить ряд продуктов, применяемых для обеспечения безопасности в системах управления, среди которых: «Safety integrated» фирмы Siemens, «Safety technology» фирмы Bosch-Rexroth, «Guard PLC» фирмы Rockwell, «CC-Link Safety» фирмы Mitsubishi, «openSAFETY» фирмы B&R.

Использование интеллектуальных аппаратных устройств ввода/вывода. В современных условиях развития техники происходит постоянное увеличение объемов данных, получаемых с объекта управления, что является отражением концепции «Индустрия 4.0», одной из основополагающих особенностей которой является работа с «большими» данными (анг. «Big data»). Такой подход предполагает обработку информации на всех этапах работы с ней, в том числе непосредственно на объекте управления. До недавнего времени в системах логического управления предварительная фильтрация и обработка сигналов производились только для аналоговых типов входов. В современных системах управления появилась возможность использования аппаратных входов/выходов, имеющих дополнительные функции, значительно расширяющие интеллектуальные возможности обработки информации непосредственно на объекте управления. К возможностям интеллектуальных устройств ввода/вывода можно отнести следующие: фильтрация входных сигналов и устранение эффектов «дребезга» контактов, поддержка специализированных протоколов связи с датчиками (например, EnDat), возможность объединения различных типов входов/выходов в единые группы для предварительной обработки поступающей информации и др. Для реализации работы с интеллектуальными устройствами ввода/вывода фирмы-производители систем автоматизации предлагают различные технологии, среди которых можно выделить: IndraControl InLine фирмы Bosch-Rexroth, SIMATIC IOT 2000 фирмы Siemens, DeviceLogix фирмы Rockwell Automation, IoT фирмы B&R.

Использование высокоскоростных протоколов связи на базе технологии Ethernet. На сегодняшний момент на рынке доступны пассивные устройства для организации ввода/вывода, которые поддерживают обмен данными по одному из стандартных высокоскоростных протоколов связи, использующих технологию Ethernet, которая на физическом уровне промышленных систем стала стандартом «де-факто». Указанные устройства стоят на порядок дешевле стандартных ПЛК в связи с тем, что не содержат интеллектуальных модулей способных организовать обработку алгоритмов управления оборудованием. Если логический контроллер, на базе которого реализуется система управления, поддерживает стандартный высокоскоростной протокол связи, то к нему можно подключить ряд пассивных аппаратных модулей ввода/вывода, в том числе расположенных на значительном удалении от основных элементов системы управления, и таким образом организовать полноценную систему логического управления. Ведущие мировые производители систем управления поддерживают один из распространённых высокоскоростной промышленный протокол связи. Например, на базе технологии Ehternet (SERCOS - производители, входящие в ассоциацию SERCOS [54-55], EtherCAT - фирма Beckhoff [56-57], Profibus и Profinet - производители, входящие в ассоциацию PI, Ethernet IP поддерживается компаниями Rockwell Automation и Opto 22, Modbus TCP [58], компаниями Fastwell и ОВЕН).

Распределенный принцип построения систем логического управления. Распределенные системы управления (англ. Distributed Control System) – это решение в области проектирования систем управления, появившееся в конце прошлого века в связи с: увеличением количества контролируемых у объекта управления параметров (датчиков), увеличением территории, на которой расположены отдельные элементы объекта управления, и усложнением алгоритмов управления. Современные распределенные системы управления, помимо основных функций, позволяют реализовать:

- Один или несколько высокоскоростных протоколов связи для подключения удаленных аппаратных модулей входов/выходов.
- Многоранговые промышленные сети по принципу «master-slave», что позволяет управлять разнородным технологическим оборудованием в рамках единой системы с возможностью распределения функций управления между узлами в рамках указанной сети.
- Работу в локальных и глобальных вычислительных сетях. Контроллер, в котором предусмотрена поддержка технологии Ethernet и реализован протокол TP/IP, способен работать в рамках вычислительной сети предприятия. В этом случае предусмотрена поддержка работы со стандартными сетевыми аппаратными устройствами (коммутаторы, маршрутизаторы, шлюзы и т.д.), что позволяет реализовывать в том числе многоранговую сеть.
- Поддержку беспроводного доступа. На сегодняшний день полезной опцией аппаратной платформы контроллеров является поддержка беспроводного доступа (например, WI-FI). Это позволяет, используя мобильные терминалы, иметь доступ к вспомогательным функциям систем логического управления, таким как конфигурирование, настройка, задание постоянных величин и т.д. В этом случае оператор, имея один мобильный терминал, может осуществлять поочередное профилактическое обслуживание узлов автоматизации.
- Удаленное конфигурирование и диагностику (в том числе по сети Internet, находясь на удалении от объекта управления). На основе канала доступа к глобальной сети реализуется связь с диагностическим центром, который может находится в любой точке земного шара. Специалисты осуществляют удаленный сбор информации об объекте управления, в том числе во время его работы.
- Возможность удаленной загрузки и отладки программ логического управления, что позволяет реализовать единый терминал, с которого отлаживается программа логического управления для группы систем, обычно находящихся в пределах одного цеха.

Использование операционных систем реального времени. На сегодняшний момент на рынке системного программного обеспечения доступен ряд операционных систем реального вре-

мени, которые устанавливаются на вычислительные платформы общего назначения. В результате получается полноценный логический контроллер, который используется в качестве узла систем автоматизации для решения широкого круга задач. В качестве таких операционных систем фирма Siemens использует Windows 7 версии Legasy (адаптированная версия под контроллеры Siemens с расширением реального времени) и Embaded, фирма Rockwell Automation - Windows 10 IoT, фирма B&R – Windows 7 Embaded, Windows 10 IoT, Debian 8, российская фирма Fastwell – Windows XPe, Linux RT, QNX. [59-61] и др. Указанные ОС поддерживают, в том числе, работу в режиме реального времени на мобильных платформах. Общедоступные ОСРВ развиваются бурно, динамично и быстрее узкоспециализированных ОС предназначенных для работы с конкретными вариантами оборудования.

При использовании общедоступных ОСРВ в промышленных системах управления необходимо учитывать юридический аспект, в соответствии с которым лицензия на ОСРВ может приобретаться только разработчиками систем управления, которые продают готовое оборудование с предустановленными ОС конечным пользователям. Продажа и перепродажа ОС отдельно от готового устройства невозможна. Применение общедоступных ОСРВ дает ряд преимуществ, среди которых можно выделить следующие:

- Работа в привычном пользователям интерфейсе на основе оконных приложений при проектировании и отладке программного кода. Большинство систем ОСРВ имеют аналоги среди ОС общего назначения (например, Linux – Linux RT, Windows – Windows Embaded и др.), в которых может вестись разработка и отладка приложений. Стиль работы пользователя при этом не будет отличаться от стиля работы с домашними ПК и будет интуитивно понятен. В то же время, для программистов интерфейсных приложений (например, терминалов оператора) доступны стандартные библиотеки элементов, которые значительно облегчают разработку. Для операционных систем общего назначения на высоком уровне находится инструментальное сопровождение разработки программного обеспечения, существуют хорошо зарекомендовавшие себя средства разработки, в том числе применяемые для больших проектов [62-64].
- Использование единой исполнительной среды, в которой работают основные функциональные модули системы управления. В рамках единой исполнительной среды предполагается работа ядра логического управления, терминала оператора, подсистемы программирования и других модулей, взаимодействие которых обеспечивается по одному из стандартных механизмов (разделяемая память, протокол TCP/IP на локальной шине).
- Система управления позволяет реализовать подключение на основе поддерживаемых сетевых протоколов (например, TCP\IP, UDP и др.) внешних устройств, таких как: мобильный терминал оператора, система удаленного мониторинга и настройки и т.д.

- Снижение себестоимости системы управления, за счет снижения издержек на разработку специализированного системного ПО.
- Для ОС, устанавливаемых в системы управления оборудованием, действует длительный цикл продаж и поддержки. В среднем, учитывая данные по различным производителям ОСРВ, полный цикл поддержки длится около пятнадцати лет с моменты выхода последнего обновления. При этом полноценная поддержка с предоставлением средств разработки и лицензий длится около десяти лет, далее производится частичная поддержка, которая включает передачу критических обновлений и доступность лицензий, ограниченных по времени.

Однако, применение ОСРВ в рамках систем управления имеет и свои недостатки, к которым можно отнести следующие:

- В себестоимость системы управления обязательно должна быть включена стоимость ОСРВ.
- Совместимость версий программного обеспечения используемого в системе управления, обеспечивается разработчиком системы.
- Техническая поддержка конечных пользователей, в том числе по вопросам работы ОСРВ, осуществляется производителем системы управления. Это предполагает наличие сертифицированных специалистов, отвечающих за работу ОСРВ в отделе технической поддержки компании разработчика системы управления.

Использование единого интерфейса для взаимодействия устройств на базе технологии OPC UA. В начале 90-х годов прошлого века возникла идея разработки универсального инструментария коммуникации устройств различных производителей, работающих на базе отличающихся между собой промышленных протоколов. В этой связи возник стандарт OPC (аббревиатура от английского Open Platform Communications, ранее OLE for Process Control). Технология OPC предоставляет разработчикам приложений промышленной автоматизации универсальный интерфейс обмена данными с промышленными устройствами, поддерживающими указанную технологию. Разработчики промышленных устройств для поддержки технологии OPC реализуют специальное приложение OPC-сервер, который выступает промежуточным звеном между технологическим оборудованием и внешними приложениями автоматизации. Ранние решения в области технологии OPC были ориентированы на работы в рамках ОС Windows (COM/DCOM, ActiveX, OLE и др.), современные решения являются платформонезависимыми (OPC XML DA и OPC UA). Для координации работ по разработке и поддержке OPC стандартов в 1994 году ведущими разработчиками средств автоматизации была создана международная некоммерческая организация OPC Foundation [65]. На сегодняшний день наиболее часто используемой является кроссплатформенная технология OPC Unified Architecture (OPC UA). Это первый протокол OPC, который базируется не на технологии COM/DCOM, поддерживает безопасное соединение за счет

валидации клиентов и серверов и имеет средства противодействия внешним атакам. Принцип работы OPC UA основан на технологии клиент-сервер. Система может иметь много клиентов и серверов, при этом клиент может быть подключен к нескольким серверам и сервер может поддерживать нескольких клиентов. Приложение, реализованное пользователем, может объединять группы клиентов и серверов для пересылки сообщений. Клиент выполняет запросы к OPC серверу через внутренний интерфейс, коммуникационный стек конвертирует запросы в сообщения для вызова необходимого сервиса, которые посылает серверу. После получения ответа на запросы коммуникационный стек передает их в клиентскую программу. Адресное пространство OPC сервера – это множество узлов, доступных клиентской программе с помощью сервисов OPC UA. "Узлы" представляют реальные объекты, их определения и перекрестные ссылки.

Использование программно-аппаратных средств управления движением. Системы управления движением (англ. Motion Control) – это отдельный класс устройств автоматизации, имеющий программные и аппаратные компоненты, направленный на решение задач управления перемещениями узлов технологического оборудования. Средства управления движением ведущих производителей обычно поставляются как комплектные решения совместно с приводной техникой, средствами визуализации, обеспечения безопасности и др. Развитие современных систем управления привело к расширению функционала специализированных систем (ПЛК, PAC, системы управления движением, ЧПУ, робототехнические системы), что в свою очередь привело к размытию границ между типами систем управления. Сегодня высокопроизводительные ПЛК и PAC системы имеют специализированные программно-аппаратные решения для управления движением, а Motion Control системы могут реализовывать логическое управление по заданным алгоритмам. Для управления движением производители систем логического управления предлагают специализированные технологии, среди которых можно выделить следующие: SIMOTION фирмы Siemens, IndraMotion фирмы Bosch-Rexroth, Motion Control Integrated фирмы Rockwell Automation, программно-аппаратные ШИМ модули фирмы B&R.

Удаленная настройка и диагностика системы. Современные системы управления – это сложный технический объект, при возникновении неисправностей на котором необходимо привлечение квалифицированных инженеров по сервисному обслуживанию. Выезд указанных специалистов стоит дорого и не всегда возможен. Существует тенденция дистанционной диагностики, настройки, конфигурирования и отладки программного обеспечения систем управления. Эта возможность появляется, если система управления имеет канал доступа к сети Internet, на основе которого реализуется связь с диагностическим центром, который может находится в любой точке мира. Специалисты осуществляют удаленный сбор информации об объекте управления, в том числе во время его работы. Система управления в этом случае выступает в качестве

сервера, к которому подключается диагностическое приложение-клиент. Интерфейс приложения-клиента может быть реализован на основе web-технологий и запускаться посредством стандартного браузера. Ведущие мировые производители систем логического управления имеют следующие решения для реализации дистанционной диагностики: SIMATIC NET на базе технологии Industrial Ethernet CP фирмы Siemens, IndraControl Service Tool на базе web-технологий фирмы Bosch-Rexroth, Smart RTU фирмы Mitsubishi, Secure Remote Maintenance фирмы B&R.

1.3 Анализ возможных путей эволюции программно-аппаратных средств логического управления

На сегодняшний день основная часть систем логического управления реализуется на основе внешне реализованных ПЛК, однако с развитием вычислительных систем большее внимание уделяется контроллерам, которые в качестве аппаратной базы используют стандартные вычислительные платформы. Принципы, лежащие в основе систем, организованных на базе персонального компьютера, были сформулированы в работах таких ученных как: Rullan A., Basset E.W., Gee D., Hohman T. [30, 66-69]. Наиболее перспективными среди новых форм организации систем логического управления являются программно реализованные контроллеры и PAC системы.

1.3.1 Программно реализованные логические контроллеры

Появление программно реализованных логических контроллеров связывают с появлением в 1995 году системы SCADA TRACE MODE, интегрированной со средой программирования контроллеров. Впервые пользователь получил инструмент, позволяющий как разрабатывать операторские рабочие места, так и программировать контроллеры, что позволило увеличить производительность труда разработчиков. Наибольший эффект идея программной реализации контроллеров дает в системах управления, в которых контроллеры применяются как вспомогательные устройства для решения логической задачи управления, например, в устройствах числового программного управления (ЧПУ). В этом случае программно-математическое обеспечение ядра логического управление устанавливается на имеющуюся аппаратную платформу и работает в одной среде с основной системой управления на базе стандартной ОСРВ.

Такого рода системы управления зачастую построены в соответствии с клиент-серверной архитектурой, при этом клиентская и серверная часть могут работать на базе одного компьютера.

С клиентского приложения пользователь может ввести программу логического управления и необходимые значения переменных или констант. Серверная часть контроллера представляет из себя исполнительное ядро, на уровне которого происходит изменение переменных и формирование управляющих сигналов согласно заданному алгоритму.

Стандартный ПЛК предоставляет для размещения прикладной программы память размером в несколько килобайт и способен обрабатывать эту программу с тактом в несколько миллисекунд. Ядро программно реализованного логического контроллера устанавливается на вычислительную платформу, обладающую значительными ресурсами (мегабайтами оперативной памяти, гигагерцовым процессором, жестким диском объемом в десятки гигабайт, платами расширения с различными интерфейсами и т.д.). При этом вычислительные ресурсы могут быть использованы для замены аппаратных средств на программно реализованные функциональные модули в ядре контроллера, которые могут вызываться средствами прикладных программ произвольное число раз. Указанный тип систем обладает достаточной надежностью за счет использования платформ, выпускаемых массовыми тиражами (по сравнению с мелкосерийными специализированными решениями), в которых ошибки сведены к минимуму за счет отлаженной технологии производства. Вычислительное ядро программно реализованного контроллера, установленное на аппаратную платформу одноплатной микроЭВМ, позволяет получить автономный универсальный инструмент для автоматизации отдельных узлов или небольших технологических объектов.

1.3.2 Программируемые контроллеры автоматизации (PAC системы)

Современные задачи автоматизации предъявляют ряд требований к применяемым в них контроллерам, среди которых можно выделить: интеграцию в общую сеть предприятия и обмен данными в её рамках, взаимодействие между системами управления различного уровня, поддержка стандартных интерфейсов подключения периферийных устройств и т.д. Указанные задачи решаются на базе платформы современных персональных компьютеров, в то время как для традиционных ПЛК эти задачи могут быть решены лишь за счет подключения дополнительных аппаратных средств. Применение ПК в качестве вычислительной платформы для реализации систем автоматизации представляется затруднительным в связи с тем, что в них не решены проблемы реализации цикла логического управления в режиме реального времени и аппаратное обеспечение не имеет защиты от вредных воздействий промышленной среды. В качестве решения, которое совмещает в себе преимущества современных ПЛК и ПК (рисунок 1.4) производи-

телями была предложена концепция нового типа контроллеров, для которых в 2001 году аналитики ARC Advisory Group ввели термин Программируемые Контроллеры Автоматизации (англ. Programmable Automation Controllers PAC). [70]

Рисунок 1.4 Возможности PAC контроллеров

PAC контроллеры – это гибкие и перестраиваемые средства автоматизации, которые могут быть модифицированы под конкретную систему управления по требованию конечного пользователя и позволяют максимально расширить возможности программной и аппаратной интеграции. Совмещение возможностей программной реализации и инженерного подхода позволяет: получить доступ ко всем параметрам и функциям в рамках единой системы управления; объединяет классический ПЛК, программно реализованный контроллер, аппаратные устройства ввода/вывода сигналов, системы управления движением, ПИД регуляторы, системы визуального наблюдения и системы обработки информации. Использование технологии Ethernet, протокола TCP/IP, глобальной сети Internet и IT стандартов позволяет интегрировать PAC контроллеры в единую среду цифрового машиностроительного предприятия. Решения, применяемые в PAC контроллерах, позволяют осуществлять согласованную работу контроллера с системами управления предприятием, например, обмен производственной информацией и подключение систем управления оборудованием к системам ERP (Enterprise Research Planning) и CPM (Corporate Performance Management). В таблице 1-3 приведена систематизация структуры и функционала программного и аппаратного обеспечения PAC контроллеров от компании National Instruments [71].

Программное обеспечение PAC контроллеров на системном уровне поддержано функциями ОСРВ, на уровне пользовательских приложений реализованы функции управления и анализа данных. Аппаратное обеспечение включается в себя: интерфейсы взаимодействия с внешней средой (модули удаленных входов/выходов, аппаратные устройства реализации связи с двигателями, терминалы оператора), устройства реализации связи (сетевые карты и карты реализации

промышленных протоколов) и вычислительные модули (сопроцессор, ОЗУ, энергонезависимая память).

Таблица 1-3 Структура и функции программного и аппаратного обеспечения РАС контроллеров

<table>
<tr><th colspan="12">Гибкое открытое программное обеспечение</th></tr>
<tr><th colspan="6">Функции управления и анализа</th><th colspan="6">ОС РВ</th></tr>
<tr><td>Алгоритмы управления</td><td>Анализ сигналов</td><td>Регистрация данных</td><td>Реализация сетевых протоколов</td><td>Интеграция внешнего ПО</td><td>Информационная и технологическая безопасность</td><td>Организация входов /выходов и системных таймеров</td><td>Реализация цикла логического управления</td><td colspan="2">Обработка исключений</td><td>Встроенные сервисные функции</td></tr>
<tr><th colspan="12">Надежное модульное аппаратное обеспечение</th></tr>
<tr><th colspan="5">Входы/выходы</th><th colspan="4">Реализация связи</th><th colspan="3">Вычислительные модули</th></tr>
<tr><td>Аналоговые и цифровые входы/выходы</td><td>ПЛИС</td><td>Управление движением</td><td>Терминалы оператора</td><td>Safety модули</td><td>Ethernet</td><td>Удаленная диагностика</td><td>Промышленные протоколы</td><td>Сопроцессор</td><td>ОЗУ</td><td>Flash память</td><td>Энергонезависимая память</td></tr>
</table>

Применение РАС контроллеров позволяет получить следующие преимущества при реализации систем логического управления:

- Повышение производительности. Согласно эмпирическому закону Гордона Мура количество транзисторов на кристалле интегральной схемы удваивается каждые 24 месяца, а производительность микропроцессоров удваивается каждые 18 месяцев. В связи с тем, что РАС системы используют в качестве аппаратной базы современные вычислители, их производительность увеличивается за счет естественной эволюции вычислительных платформ, не требуя для этого дополнительных вложении со стороной разработчиков систем управления.
- Уменьшение издержек за счет применения общей аппаратной базы на основе стандартов построения систем управления и промышленных сетей. Например, в качестве периферийного оборудования для РАС систем могут использоваться устройства, поддерживающие стандартизованные интерфейсы взаимодействия (USB, PCI и др.). За счет стандартизации механизмов сетевого взаимодействия РАС система может быть интегрирована в локальную сеть предприятия.
- Открытость на уровне конечного пользователя. РАС системы обеспечивают гибкость в выборе аппаратных средств и языков программирования, которые соответствуют требованиям контурного объекта управления.
- Возможность решения различных типов задач на одной платформе: логические задачи, задачи управления движением, задачи управления процессами, задачи реализации пользовательского интерфейса и др.

1.3.3 Современные системы логического управления, предлагаемые ведущими производителями

Системы логического управления на сегодняшний день реализуются на базе трех наиболее распространенных типов контроллеров: ПЛК, программно реализованный контроллер и PAC. Каждый из производителей поддерживает один или несколько из перечисленных типов контроллеров. Производителей систем автоматизации по количеству выпускаемого в год оборудования условно можно разделить на: крупных, средних и небольших.

К крупным разработчикам систем автоматизации относят тех производителей, чьи системы имеют массовое применение в различных сферах (станкостроение, машиностроение, энергетика и др.). Наиболее яркими представителями крупных компаний разработчиков систем автоматизации являются фирмы Siemens Rockwell Automation (Allen-Bradly), Mitsubishi и др. Особенностью работы таких производителей является принцип комплектной поставки средств автоматизации. Фирма-производитель гарантирует работоспособность средств автоматизации и их соответствие заявленным характеристикам только при использовании полного комплекса представленных программно-аппаратных решений (контроллеры, периферийное оборудование, кабели и т.д.). Во всех остальных случаях потребитель, реализуя систему управления берет на себя риски несовместимости отдельных узлов. Такой подход позволяет производителю устанавливать высокие цены, в том числе на комплектующие изделия.

Средства автоматизации, производимые компаниями средней величины (Bosch Rexroth, B&R), обычно имеют сектора промышленного производства, в которых они наиболее востребованы. Система автоматизации и оборудование производителей средней величины отличаются большей унификацией и допускают открытость для интеграции программных и аппаратных решений сторонних разработчиков или конечных пользователей. Помимо этого они могут использовать в качестве инструментальных средств программные и аппаратные решения сторонних производителей. Например, компания Bosch Rexroth в качестве среды разработки программ логического управления использует доработанную версию среды CoDeSys, что позволяет компании иметь современное программное решение и при этом не содержать большой штат высокооплачиваемых специалистов.

Небольшие производители характеризуются направленностью на решение определенного класса задач и не имеют ресурсов для развития общепромышленных сфер применения систем автоматизации. Одним из представителей небольших фирм производителей систем автоматизации является компания Opto 22, которая специализируется на разработке PAC контроллеров. Разрабатываемые компанией системы, применяются для решения задач не имеющих повышенного уровня сложности, и в проектах, не предполагающих большой серийности выпускаемых изделий.

При этом программное и аппаратное обеспечение проектируется и разрабатывается небольшими фирмами самостоятельно и характеризуются: узкой номенклатурой, ограниченным функционалом и решениями, которые концептуально не меняются на протяжении долгого периода времени.

Традиции разработки и продвижения систем логического управления на рынок в нашей стране и за рубежом различаются. Зарубежные компании предлагают комплексные решения, в которые помимо программируемого логического контроллера входят: системы ЧПУ, системы управления движением (Motion Control) и периферийное оборудование. Российские производители (ОВЕН, Fastwel) предлагают решения, ориентированные только на логическое управление, и не имеют предложений по комплексной автоматизации. Далее представлен анализ программно-аппаратных решений в области систем логического управления, в том числе решения, предложенного в рамках диссертационной работы.

Системы логического управления Siemens. Одним из мировых лидеров в области промышленной автоматизации является компания Siemens AG (Германия) [72], которая предлагает решения для автоматизации широкого круга задач. Анализ программно-аппаратных систем логического управления показывает, что компания развивает собственные решения в области автоматизации производства и использует продукцию крупных компаний производителей OEM решений в качестве комплектующих для систем автоматизации. В качестве программируемого контроллера в системе управления может быть использован ПЛК или программно реализованный контроллер, которые снабжаются модулями аппаратных входов/выходов производства Siemens или одного из OEM производителей (например, VIPA). В качестве аппаратной платформы используются ПЛК или промышленные ЭВМ производства Siemens, которые разделяются на серии, в которые входит ограниченный набор программно-аппаратных решений. Контроллер подключается в промышленную сеть на базе открытых протоколов Profibus или Profinet, которые реализованы разработчиками фирмы Siemens таким образом, чтобы скрыть свои решения. Протоколы позволяют реализовать связь: с терминалом оператора, системами управления верхнего уровня и шлюзами, позволяющими осуществить связь с сетями, работающими на базе стандарта Ethernet (рисунок 1.5).

Разработка программ логического управления производится в инструментальном средстве собственной разработки «Simatic Step 7», поддерживающем три языка программирования стандарта МЭК 61131-3 (LD, FBD, IL). [73] Разработанные программы не совместимы с другими средствами программирования ПЛК.

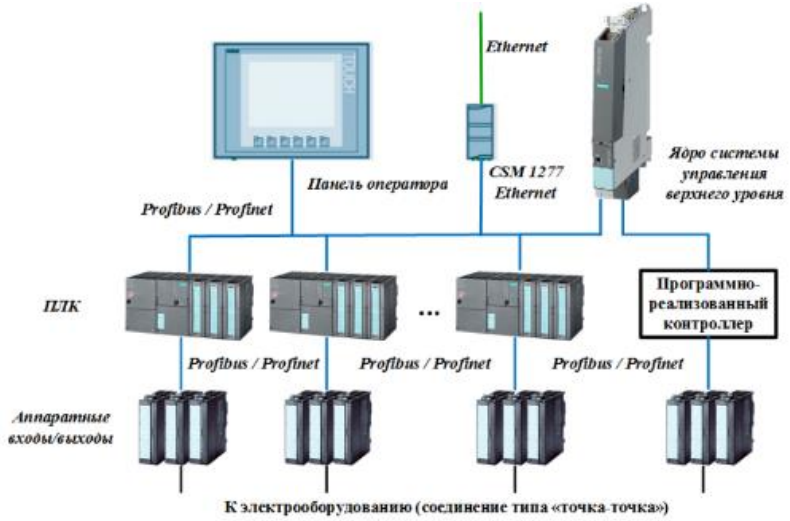


Рисунок 1.5 Сетевая структура систем логического управления Siemens

Системы логического управления Rockwell Automation. Фирма Rockwell Automation [74] является одним из лидеров мирового рынка систем логического управления и предлагает от малых ПЛК до масштабируемых PAC систем.

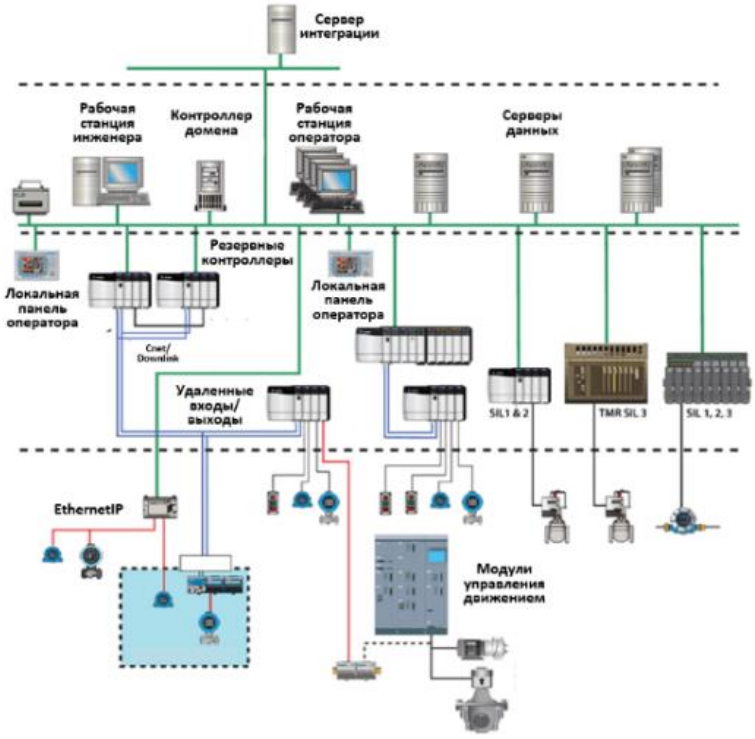


Рисунок 1.6 Сетевая структура систем управления Rockwell Automation

Компания развивает собственные решения в области автоматизации, основанные на лицензированных технологиях: FLEX IO (аппаратные входы/выходы), Factory Talk (визуализация и терминальная задача управления), Kinetix (управление перемещениями), PlantPAx (распределенное управление). В качестве высокоскоростного протокола связи системы используют протокол EthernetIP. В качестве среды разработки программ логического управления используется

STUDIO 5000 LOGIX DESIGNER, который использует языки программирования LD, ST, FBD стандарта МЭК 61131-3.

Системы логического управления Mitsubishi. Согласно исследованиям компании ARC (США), Mitsubishi Electric является мировым лидером по производству ПЛК. Распределенные системы управления на базе контроллеров Mitsubishi [75] реализуются по собственной технологии Melsec, основу которой составляет собственный высокоскоростной протокол SSCNET, модификация которого CCLink Safey используется совместно со специализированными модулями ввода/вывода для организации сетей безопасности (рисунок 1.7). Системы логического управления производства Mitsubishi в рамках единой сети объединяют разнородные решения, среди которых можно выделить следующие: ПЛК, контроллеры движения, комплектные приводы, терминалы визуализации, системы управления промышленными роботами, устройства контроля мощности и управления электрическими сигналами и др. Инструментарий программных средств для работы с продукцией Mitsubishi объединен под единым брендом MELSoft, в который входит среда программирования GX Works2, поддерживающая все языки стандарта МЭК 61131-3.

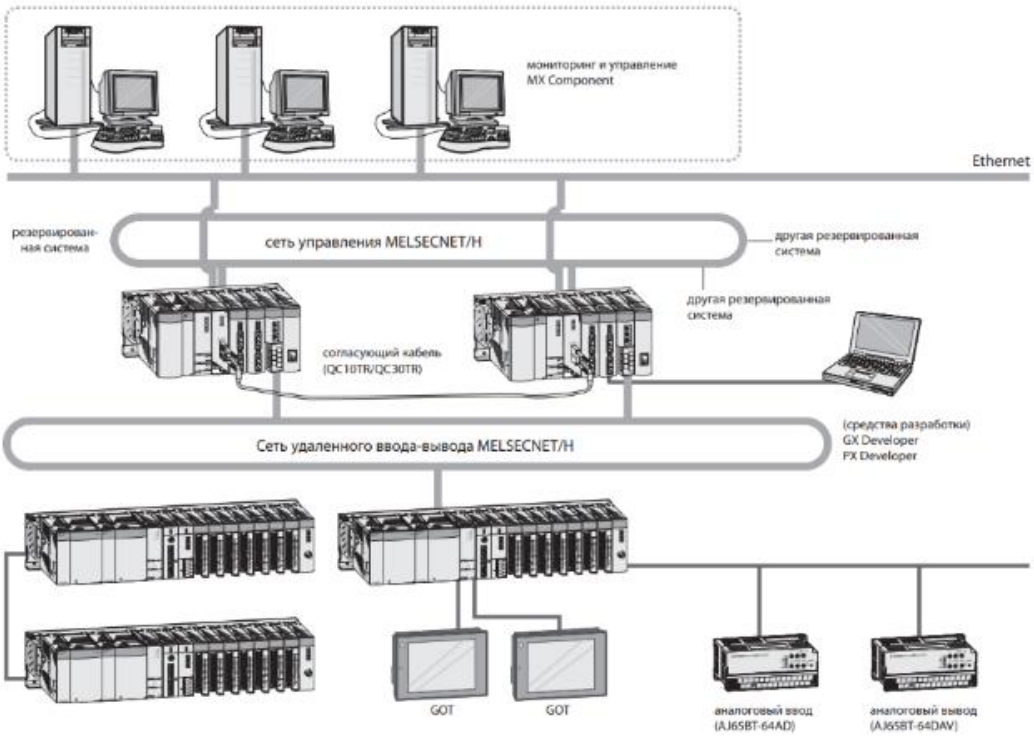


Рисунок 1.7 Сетевая структура систем логического управления Mitsubishi

Системы логического управления Bosch Rexroth. Архитектурные решения, лежащие в основе построения систем логического управления фирмы Bosch Rexroth (Германия), основаны на использовании открытого протокола SERCOS (рисунок 1.8). [54-55,76]

Указанный протокол позволяет организовать связь между системами управления разного назначения, таких как: Indra Control – логические контроллеры и системы управления движением

(Motion Control), Indra Drive - системы управления приводами, Inline - аппаратные модули входов/выходов (включая модули безопасности - Safety), системы управления пневматикой и гидравликой. Предусмотрена возможность интеграции в систему управления модулей входов/выходов OEM производителей (например, Phoenix Contact, Wago) и стороннего оборудования, поддерживающего протоколы DeviceNet, Profibus/ProfiNET и Ethernet/IP. Инструментарий разработки программ логического управления «IndraWorks» базируется на ядре среды CoDeSys и поддерживает все языки программирования стандарта МЭК 61131-3.

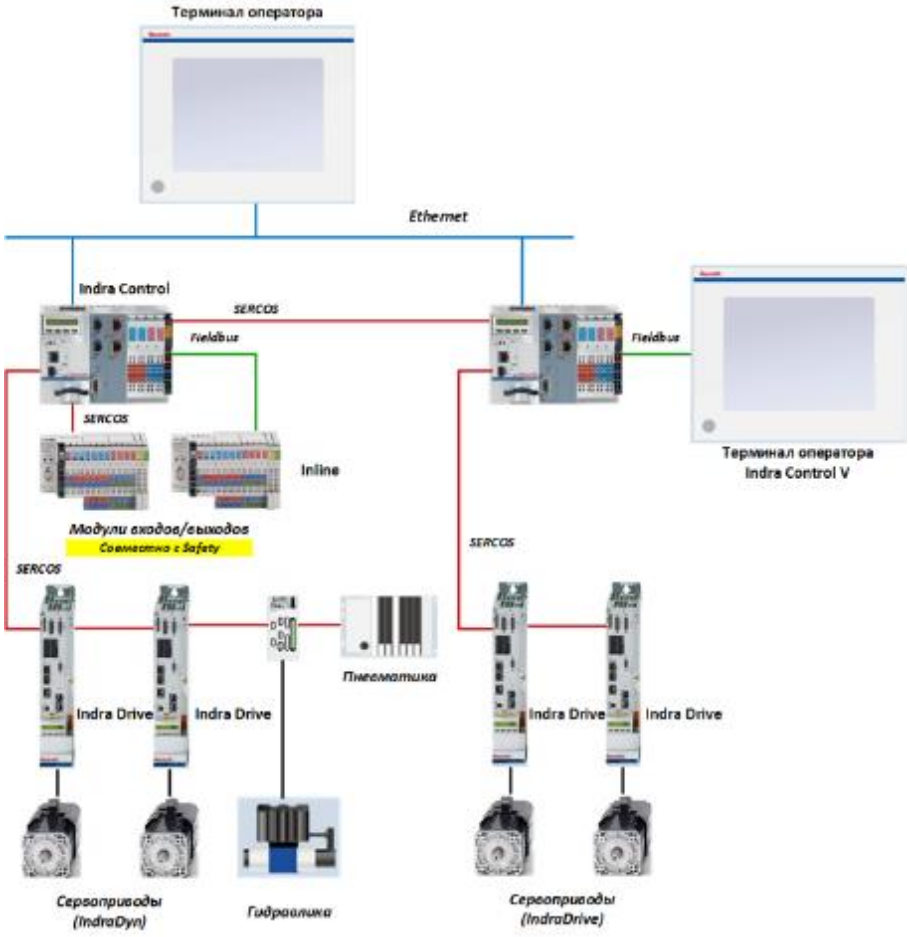


Рисунок 1.8 Сетевая структура систем логического управления Bosch Rexroth

Системы логического управления B&R. Системы логического управления производства B&R (Австрия) позиционируются как мультипротокольные решения, поддерживающие несколько промышленных высокоскоростных протоколов коммуникации. (рисунок 1.9) [77].

В качестве основного протокола связи используется PowerLink, но при конфигурировании можно выбрать иной протокол для устройств других производителей. К промышленной сети может быть подключено более одного вычислительного модуля (ПЛК или PAC контроллер) и аппаратных входов/выходов, количество которых определяется топологией сети. В качестве инструментария разработки программ логического управления используется среда Automation Studio, поддерживающая все языки стандарта МЭК 61131-3 и дополнительно языки: ANSI C,

CFC, Automation Basic. Пользователи могут создавать библиотеки пользовательских программ и использовать их повторно, но не предусмотрена возможность импорта программ в другие среды разработки.

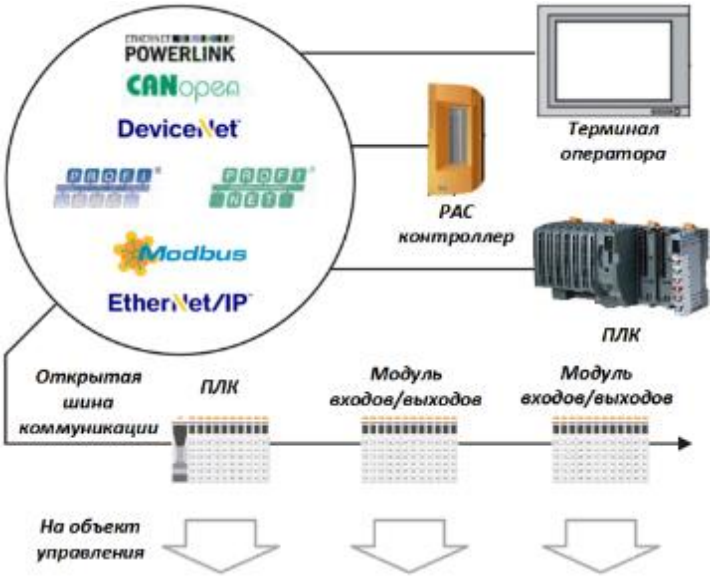


Рисунок 1.9 Сетевая структура систем логического управления B&R

Системы логического управления ОВЕН. Фирма ОВЕН (Россия) занимается разработкой программно-аппаратных решений для систем автоматизации. В качестве базового протокола связи между узлами сети используется протокол физического уровня RS-485, который позволяет связать в единую сеть ПЛК, модули аппаратных входов/выходов, терминалы оператора и др. [78] При значительном удалении оборудования друг от друга используются модули-повторители сигнала для сети RS-485. (рисунок 1.10)

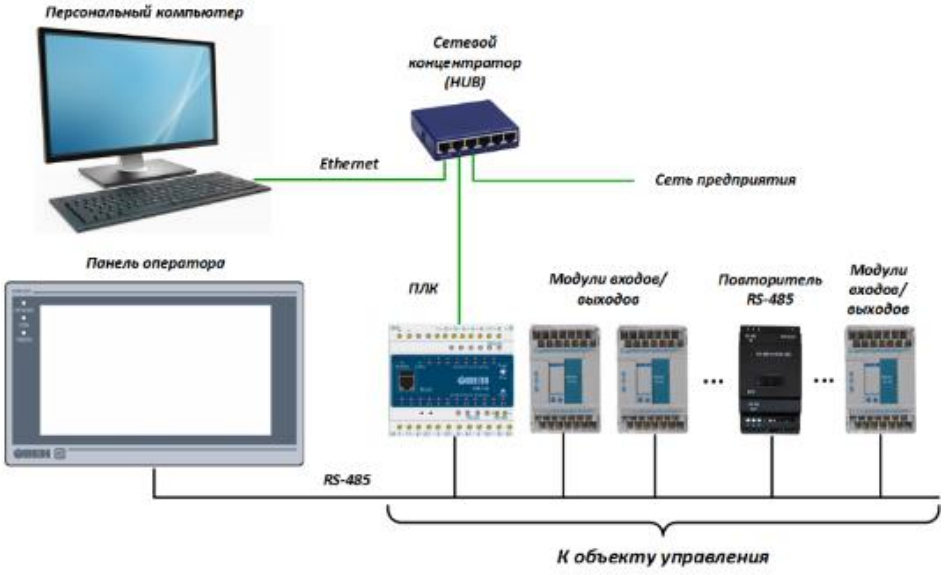


Рисунок 1.10 Сетевая структура систем логического управления ОВЕН

Для взаимодействия с сетью предприятия и системами управления верхнего уровня используется технология Ethernet и стандартное оборудование для реализации локальной вычислительной сети (например, сетевой концентратор). Программирование контроллеров осуществляется в среде CoDeSys версии 2.3, это старая версия продукта. На сегодняшний момент ведущие производители систем автоматизации используют версию 3.5, которая предлагает дополнительный набор библиотек программирования.

Системы логического управления Fastwel. Компания Fastwel (Россия) разрабатывает аппаратные решения для систем автоматизации. [79] Базовым протоколом связи между узлами сети автоматизации является протокол FBUS, который на физическом уровне использует стандарт RS-232. (рисунок 1.11)

Рисунок 1.11 Сетевая структура систем логического управления Fastwel

Помимо базового протокола имеется возможность выбрать модули связи, которые позволяют реализовать следующие промышленные протоколы: CAN, Modbus RTU/TCP и PROFIBUS. Взаимодействие с сетью предприятия возможно посредством подключения к персональному компьютеру, который имеет выход в указанную сеть. Программирование контроллеров осуществляется в среде CoDeSys версии 2.3.

Системы логического управления Opto22. Opto22 (США) - компания, профилем которой являются аппаратные и программные решения для систем промышленной автоматизации, удаленного мониторинга и сбора данных. Фирма одной из первых начала проводить исследования о возможностях реализации контроллеров на платформе персональных компьютеров и разрабатывать PAC контроллеры. [81-82] Базовой технологий, на основе которой осуществляется

связь всех аппаратных решений компании Opto22 является технология Ethernet, на основе которой реализована в том числе связь с промышленной сетью предприятия и системами управления верхнего уровня. В сети могут использоваться контроллеры, подключенные к общей сети на основе беспроводных технологий доступа. (рисунок 1.12)

Программирование систем осуществляется в инструментарии собственной разработки PAC Project, которая имеет две версии Basic и Professional. В качестве языка разработки программ используется язык Flowchart, который не соответствует стандарту МЭК 61131-3. Разработанные программы совместимы только с системами управления компании Opto22.

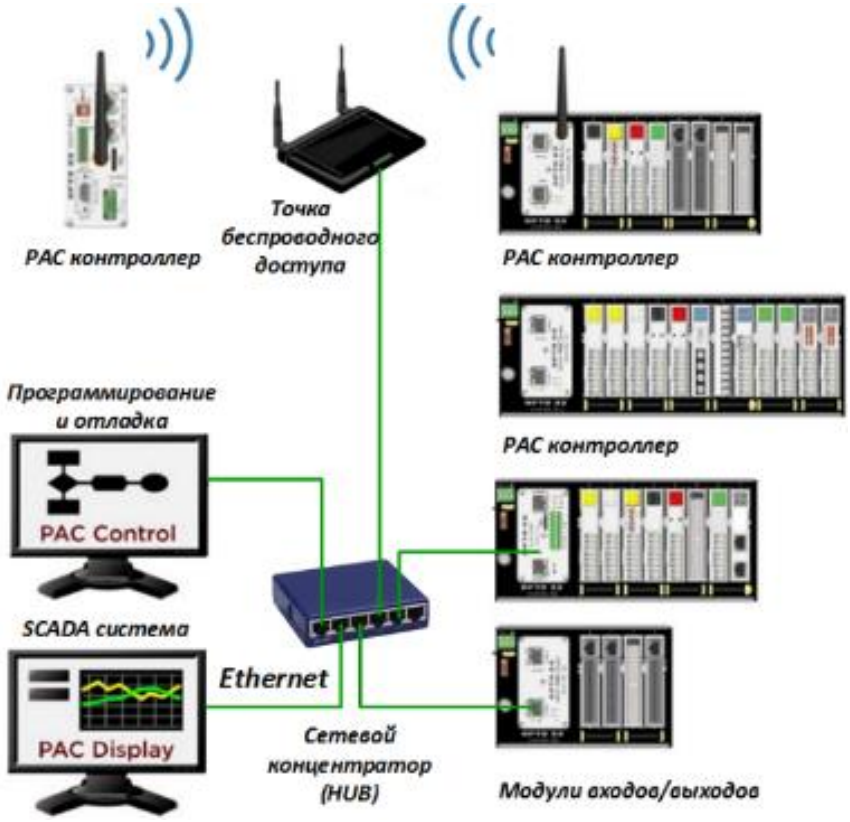


Рисунок 1.12 Сетевая структура систем логического управления Opto22

Системы логического управления на базе программно реализованного контроллера CoDeSys. Программные продукты CoDeSys производства компании Smart Software Solution (Германия) делятся на два класса: среда программирования ПЛК и система исполнения CoDeSys SP. [83-84] Среда исполнения, установленная на вычислительную платформу, позволяет получить программируемый логический контроллер, работающий в режиме реального времени. Для OEM производителей среда исполнения поставляется в виде исходного кода и может быть доработана разработчиками аппаратной платформы под свои нужды. Для систем, совместимых с ПКЮ существуют готовые решения и конфигурации, которые не требуют дополнительной адаптации. Режим поддержки протоколов связи и технологий коммуникации зависит от аппаратных возмож-

ностей вычислительной платформы, на которую установлена исполнительная среда. Среда программирования и исполнения полностью совместимы между собой, для разработки доступны все возможности и языки программирования, которые поддерживаются инструментарием CoDeSys. Разработанные программы логического управления могут быть повторно использованы, в том числе, на контроллерах сторонних производителей, использующих CoDeSys в качестве среды программирования.

1.3.4 Систематизация аналитических данных о системах логического управления

Результаты проведенного анализа систем логического управления от крупных, средних и мелких производителей систематизированы в таблице 1-4. Проведенный анализ позволил сделать следующие выводы:

- крупные производители развивают комплексные решения на аппаратуре собственного производства и малого числа OEM производителей;
- производители средней величины используют стандартные механизмы коммуникации, что позволяет подключать оборудование сторонних производителей;
- стирается грань между аппаратными решениями на базе ПЛК и ПК;
- крупные и средние производители ориентированы на интеграцию покупных программных продуктов, которые позволяют создать конкурентоспособный инструментарий разработки и поддержки систем логического управления;
- отсутствует механизм переноса программ логического управления из одного инструментария разработки в другой;
- современные системы логического управления ориентированы на интеграцию в производственную сеть предприятия на базе технологии Ethernet;
- при применении промышленных протоколов связи, выбор делается в пользу высокоскоростных протоколов на базе технологии Ethernet.

Требования к системам можно разделить на общие, которые едины для всех типов систем управления, и частные, которые зависят от объекта управления и решаемых задач. К общим требованиям отнесены:

- использование производителями PAC систем в качестве вычислительной платформы компьютеров (персональных, одноплатных, промышленных), что обеспечивает повышение вычислительных возможностей систем управления;

Таблица 1-4 Анализ систем логического управления различных производителей

Фирмы / Свойства	Siemens	Rockwell	Bosch Rexroth	B&R	Mitsubishi	ОВЕН	FASTWEL	Opto 22	CoDeSys	Решение
Поддержив. типы контроллеров	ПЛК, Soft PLC, PAC	ПЛК, Soft PLC, PAC	ПЛК	ПЛК, PAC	ПЛК, Soft PLC	ПЛК	ПЛК	PAC	Soft PLC	Soft PLC, PAC
Программно-аппаратная база	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	Ограниченная линейка контрол.	ПО без аппаратной платформы	Вариативная платформа
Интерфейсы связи	Ethernet	Ethernet, USB, слот для карт памяти microSD	Ethernet, USB, слот для SD карт	Ethernet, RS-232, USB	Ethernet, Melsec, USB, RS232, ...	Ethernet, RS-232, RS-485	CAN, RS-485, Profibus	Ethernet, RS-232	отсутствует	Ethernet, RS-232, RS-422, RS-485, USB, ...
Промышленные протоколы	Profibus, Profnet (закрытые протоколы)	EtherNet/IP, IO-Link	SERCOS, Profibus, Profinet, DeviceNET	Powerlink	MelsecNET, CC-Link, CC-Link IE, SSCNET III	ОВЕН, Modbus (RTU, ASCII, TCP), DCON	CAN, Modbus (RTU, TCP), Profibus	EtherNet/IP, PPP, Modbus/TCP, OptoMMP	SERCOS, EtherCAT, …	SERCOS, EtherCAT, Modbus (RTU, TCP), CAN
Протоколы общего назначения	TCP/IP	TCP/IP	TCP/IP	Протоколы, поддерживаемые ПК (для PAC)	TCP/IP	отсутствует	отсутствует	TCP/IP, SNMP, SMPT	Протоколы, поддерживаемые ПК	Протоколы, поддерживаемые ПК
Языки программир., МЭК 61131-3	LD, FBD, IL	LD, ST, FBD	FBD, LD, IL, ST, SFC, CFC	IL, ST, LD, FBD, SFC, ANSI C, CFC, ABasic	FBD, LD, IL, ST, SFC	FBD, LD, IL, ST, SFC, CFC	FBD, LD, IL, ST, SFC, CFC	Flowchart (не стандарт МЭК 61131-3)	FBD, LD, IL, ST, SFC, CFC	FBD
Инструмент. разработки	Simatic STEP7	STUDIO 5000	IndraWorks (на базе CoDeSys)	Automation Studio	GX Works2	CoDeSys 2.3	CoDeSys 2.3	PAC Project	CoDeSys	FBEditor (собственная разработка)

- поддержка нескольких высокоскоростных протоколов коммуникации (мультипротокольность), что позволяет интегрировать в состав систем логического управления широкую номенклатуру периферийного оборудования;
- предоставление возможности повторного использования программ логического управления, разработанных ранее и оформленных в виде отдельных библиотек, что позволяет сократить время, необходимое на проектирование и отладку систем управления;
- возможность конфигурирования как программной, так и аппаратной составляющей систем логического управления, для адаптации системы управления под конкретный объект;
- возможность конфигурирования как программной, так и аппаратной составляющей систем логического управления, что позволяет адаптировать систему управления под конкретный объект управления;
- применение специализированных аппаратных модулей и узлов управления для обеспечения противоаварийной защиты (safety модули).

К частным требованиям относятся:

- количество и тип применяемых модулей ввода/вывода;
- размер вычислительных ресурсов аппаратного обеспечения системы логического управления;
- возможность использования аппаратного обеспечения для работы в агрессивной внешней среде (радиация, высокая влажность и т.д.);
- возможность подключения специализированных устройств (дисплей, панель и др.);
- функции, реализуемые программным обеспечением.

Область применения систем логического управления определяет частные требования, предъявляемые к программной и аппаратной составляющей систем. Рассмотрим подробнее области применения контроллеров.

Управление электроавтоматикой станков. При управлении станками применяются ЧПУ, в составе которых за решение логической задачи отвечают системы логического управления. В качестве автономного узла автоматизации в рамках системы ЧПУ целесообразно использовать либо программно реализованный контроллер, работающий в рамках единого программно-математического обеспечения системы ЧПУ, либо ПЛК. Применение РАС контроллеров не оправдано с экономической точки зрения в связи с тем, что они предоставляют расширенные вычислительные возможности, которые не востребованы при управлении электроавтоматикой станков и дублируют основной функционал системы ЧПУ, но при этом они отражаются на себестоимости системы. В станочных системах наиболее востребованными являются дискретные

модули ввода/вывода, их количество может варьироваться от нескольких десятков, до нескольких сотен в зависимости от сложности объекта управления. Также используются различные типы модулей для приема аналоговых сигналов и специализированные модули подключения энкодеров. На сегодняшний момент на предприятиях используется широкая номенклатура станочного оборудования, специфика которого должна быть учтена при проектировании программных компонент систем логического управления.

Управление технологическими процессами. Одна из самых популярных областей применения систем логического управления - управление технологическими процессами в различных областях промышленности (металлургия, химическая промышленность, машиностроение и др.). В качестве одного из видов технологических процессов могут рассматриваться циклические (периодические) процессы, в которых действия выполняются через фиксированные периоды времени.

При управлении технологическими процессами традиционно применяются ПЛК, в современных условиях работы в рамках цифровых производств все большей популярностью для решения указанной задачи пользуются PAC контроллеры. Для управления простыми процессами в качестве вычислительной платформы могут применяться недорогие решения на базе одноплатных компьютеров.

Для контроля технологических процессов архитектура системы логического управления может быть традиционной или распределенной (если процесс разнесен на расстояния). Традиционные решения характеризуются большим количеством входов/выходов, относящихся к единому узлу автоматизации. Распределенные системы имеют не более нескольких десятков входов/выходов на одном узле автоматизации. В обоих случаях в качестве основного типа используются дискретные входы/выходы и, при необходимости, аналоговые.

Управление перемещениями. Модельные ряды современных контроллеров предлагают продвинутые ПЛК, в которых реализована поддержка управления движением (англ. Motion Control). Задача управления движением на сегодня становиться частью современных систем логического управления, её поддержка реализуется не только в ПЛК, но и в PAC и Soft PLC контроллерах, отличие заключатся в количестве одновременно интерполируемых осей. В качестве аппаратных входов/выходов используются традиционные дискретные модули, а при управлении частотными преобразователями по аналоговым каналам – модули аналоговых входов/выходов с требуемыми параметрами (токовые, потенциальные и т.д.).

Для реализации управления движением в ядре системы управления должен быть предусмотрен модуль работы с приводами, который поддерживает изменение скорости, ускорения, положения и других характеристик двигателей. При этом для ориентации на конкретный объект

управления реализуется программа логического управления, в которой должны быть предусмотрены регуляторы для каждой из осей управления (если их несколько). Реализованные регуляторы целесообразно объединять в специализированные библиотеки, которые могут быть подключены к требуемым проектам.

Задачи диагностики и мониторинга. Задачи диагностики и мониторинга – класс задач, в которых не осуществляется процесс управления. Системы логического управления реализуют диагностирование объекта управления посредством непрерывного опроса датчиков и мониторинг объекта, который предполагает наблюдение и сравнение полученных данных с эталонными или вычисленными по соответствующей модели, а также их регистрацию в специализированных средствах и запись протокола протекания технологического процесса.

Спектр объектов управления систем диагностики и мониторинга достаточно широк, они характеризуются количеством точек съема информации (количеством датчиков), которые в свою очередь определяют количество и тип модулей аппаратных входов/выходов. Для этих целей используют дискретные и аналоговые аппаратные входы/выходы.

Обеспечение безопасности. Для обеспечения безопасности и критического управления технологическими объектами используют автоматизированные системы специального назначения. В них применяются специализированные аппаратные решения, направленные на выявление потенциально опасных ситуаций и предотвращение их неконтролируемого развития. В случае невозможности предотвращения подобной ситуации, системы логического управления проектируется таким образом, чтобы перевести процесс в безопасное состояние в кротчайшие сроки. Системы общего назначения, не имеющие специального разрешения, не могут использоваться на объектах, где обеспечение безопасности относится к критичным процессам. К таким объектам относятся травмоопасные машины (станки, пресса, промышленные роботы и др.), в которых системы барьерной защиты (ограждение, датчики положения оператора и др.) должны иметь противоаварийную защиту.

В качестве модулей аппаратных входов/выходов в указанных системах должны применяться специализированные решения - Safety модули. Помимо этого, при обеспечении безопасности необходимо реализовать дублирование вычислений и резервирование критичных процессов и процессов обеспечения безопасности, реализовать алгоритмы верификации критичных вычислений.

Управление вспомогательными технологическими процессами, к которым можно отнести: термическую обработку (печи), сушку, мойку, управление транспортными узлами (тележки), упаковку и др. Помимо этого к вспомогательным процессам можно отнести процессы

тестирования и диагностики оборудования, для которых зачастую разрабатываются испытательные стенды, например, для контроля параметров электрических двигателей или испытания параметров авиационных деталей и др.

Проведенный анализ требований, предъявляемых к современным системам логического управления, позволил выявить взаимосвязи между типом технологического оборудования, задачами и функциями систем логического управления, влияющих на структуру системы управления и определяющих состав программно-аппаратных модулей. Результаты анализа приведены в таблице 1-5.

Таблица 1-5 Реализация систем логического управления для различных типов оборудования

		Задачи СЛУ	**Тип СЛУ**	**Платформа**	**Входов/выходов**	**Программные компоненты СЛУ**
Управление станками	Многокоорд. обработка	Реализация логической задачи ЧПУ	Soft PLC, ПЛК	Промышленный, персональный компьютеры (Soft PLC), спец. исполнение (ПЛК)	Дискретные входы/выходы, аналоговые (при необходимости), специализированные (sin-cos, интерфейс энкодера и др., при необходимости)	Реализация управления master-slave
	Гибридная и многофункц. обработка					Синхронизация при управлении энергиями обработки
	Гидроабразивная обработка					Управление параметрами гидроабразивной обработки
	Лазерная обработка					Модуль обработки сигналов лазера
Управление тех. процессами	Металлургия	Управление технологическим процессом и цикловой автоматикой	Soft PLC, ПЛК, РАС	Промышленный, персональный, одноплатный компьютеры (Soft PLC), спец. исполнение (ПЛК)	Дискретные входы/выходы, модули измерения температуры + аналоговые (при необходимости)	Специализированные библиотеки программ логического управления
	Химическая промышлен.					
	Нефтегазовая промышлен.				Дискретные входы/выходы + аналоговые (при необходимости)	
	Машиностроение					
Управление перемещениями		Управление движением (Motion			Дискретные входы/выходы + аналоговые (при управлении частотным преобразователем)	Модуль управления движением + библиотека регуляторов
Диагностика и мониторинг		Сбор и обработки данных			Дискретные входы/выходы + аналоговые (при необходимости)	Специализированные библиотеки программ логического управления
Обеспечение безопасности		Контроль безопасности	ПЛК, РАС	Специализированная	Модули безопасности (Safety)	Прямое аппаратное управление

1.4 Анализ средств программирования систем логического управления

Как правило, программы логического управления разрабатываются на языках стандарта МЭК 61131-3, к которым некоторые производители добавляют один из высокоуровневых языков, например, C. Наличие сформировавшихся языков написания программ логического

управления обеспечивает преемственность поколений инженеров и повторное использование ранее созданных программ в рамках единого инструментария разработки. Популярность существующих языков программирования систем логического управления представлена на рисунке 1.13 (согласно исследованиям журнала Control Engineering [51, 52] и агентства Reed Research). Представленная диаграмма отражает тот факт, что графические языки программирования (релейно-контактных схем и функциональных блоков) остаются наиболее востребованными в связи с тем, что они интуитивно понятны инженерам-разработчикам систем, т.к. являются прямой проекцией электрических схем цикловой автоматики. Электроавтоматика технологического оборудования состоит преимущественно из стандартных узлов, работа которых описана и программная реализация алгоритмов их управления реализована для систем, созданных ранее. Систематизация реализованных ранее программных модулей логического управления и объединение их в готовые библиотеки позволяет существенно сократить время на разработку программ логического управления для вновь создаваемых систем.

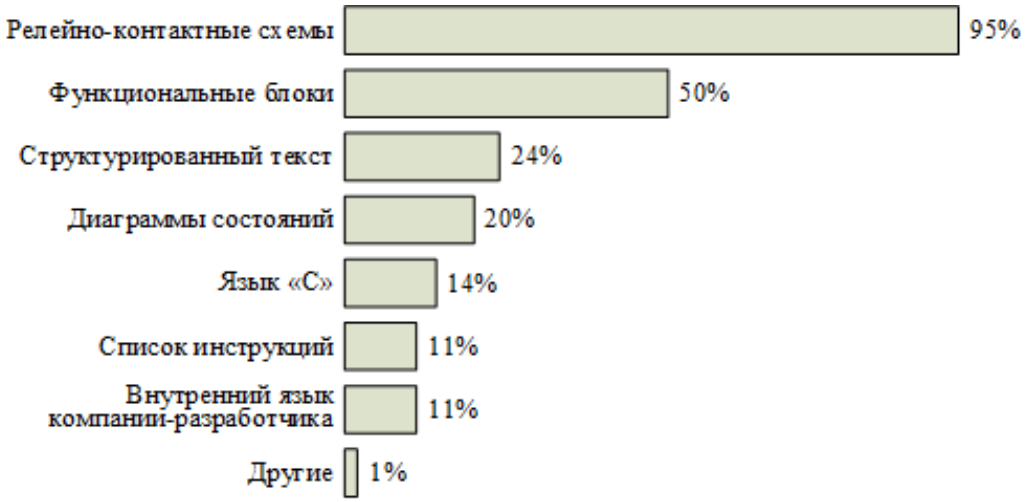


Рисунок 1.13 Используемые в ПЛК языки программирования

Однако повторное использование программного кода доступно только в рамках работы с системами управления одного производителя. В связи с этим возникает необходимость разработки архитектурных решений и математического обеспечения систем логического управления, которые позволят интегрировать в едином проекте программно-аппаратные решения различных производителей. На рынке автоматизации, среди существующих универсальных, не привязанных к конкретному производителю оборудования, средств программирования логических систем, по полноте реализации и масштабности применения выделяют ряд решений, среди которых:

- **CoDeSys** (фирма Smart Software Solutions). CoDeSys - акроним от Controller Development System. Распространенная среда программирования ПЛК в которой помимо 5 стандартных

языков программирования логических контроллеров, используется язык CFC (Continuous Function Chart) и поддерживается объектно-ориентированная парадигма. [82-83] Используется производителями оборудования для получения полноценной среды программирования стандарта МЭК 61131-3.

- **LabView** (Laboratory Virtual Instrumentation Engineering Workbench, фирма National Instruments). Система программирования и платформа выполнения программ, реализованная на основе графического языка программирования «G» фирмы National Instruments (США). LabVIEW применяется для сбора и обработки данных, управления технологическими объектами и процессами. Концепция LabVIEW близка к SCADA. [71, 85] В задачах автоматизации LabView применяется для построения интерфейса оператора на базе набора виртуальных приборов и для реализации графических схем подключения оборудования и др.
- **ISaGRAF** (фирма Rockwell Automation**)**. Программный продукт использует аппаратно независимый генератор исполняемого кода, что позволяет устранить ограничения на использование разнородной аппаратной платформы. Помимо этого, существует возможность трансляции проектов в программный код на языке С. ISaGRAF состоит из двух частей: непосредственно среды разработки (ISaGRAF Workbench) и адаптируемой под аппаратную платформу исполнительной среды (ISaGRAF Runtime), установка которой на аппаратную платформу позволяет получить полноценный контроллер. [86-88]
- **MULTIPROG wt** (фирма Klopper und Wiege Software). Исполнительское программного ядра базируется на ОСРВ собственного производства (ProConOS), среда программирования ориентирована на ОС Windows. Программный продукт является средой разработки программ логического управления с полным набором сервисов стандарта МЭК 61131-3. [89]
- **Open PCS** (фирма Infoteam Software). Это среда разработки программ логического управления стандарта МЭК 61131-3, которая может быть использована производителями оборудования для программирования контроллеров, для этого есть модуль поддержки ПЛК - SmartPLC. Программный комплекс ориентирован на использование языка программирования электравтмоатики IL. Позволяет пользоваться и другими языками, однако перед исполнением программа автоматически конвертируется в язык IL. Последняя версия программного комплекса построена на базе технологии ISaGRAF рассмотренной выше. [90]
- **iCon-L** (фирма ProSign). Принцип работы программного продукта во многом схож с принципами работы CoDeSys, но применяется для программирования средств автоматизации в значительно меньших объемах. Однако, может быть использован как средство автоматизации проектирования широкого круга задач, например, для управления детскими моделями (робототехника, авто моделирование и т.д.). Инструментарий базируется на графическом пред-

ставлении функциональных блоков и позволяет создавать программы на языке последовательных диаграмм (SFC). [91]. В таблице 1-6 представлен сравнительный анализ систем программирования.

Таблица 1-6. Анализ средств программирования систем логического управления

Характеристика / Продукт	ОС	Язык программирования	Работа с оборудованием различных производителей	Открытость	Лицензия
CoDeSys	Windows	IL, ST, LD, FBD, SFC	трансляцию в исполняемый код процессора	Ограничена	Платная
Lab View	Windows, Mac OS, Linux	Графический язык «G»	TCP/IP, UDP	Ограничена	Платная
ISaGRAF	Windows	IL, ST, LD, FBD, SFC	генерация исполняемого кода	Ограничена	Платная
Multiprog	ProConOS	FBD, LD и IL	TCP/IP, UDP/IP	Ограничена	Платная
OpenPCS	Windows	IL, ST, SFC	генерация исполняемого кода	Ограничена	Платная
iCon-L	Windows	FBD	Нет	Ограничена	Платная

Из проведенного анализа можно сделать вывод, что ряд недостатков не позволяет применять описанные программные среды в кроссплатформенных проектах. В первую очередь, это ориентация на конкретную операционную систему (чаще всего Windows), необходимость приобретения лицензии на коммерческое использование; закрытость системы, ориентированная на комплексное решение, но без возможности его модернизации и доработки. К тому же ни один из указанных программных пакетов не поставляется с «открытым» программным кодом, что делает затруднительным их использование в качестве базы при разработке программного продукта.

1.5 Научная проблема построения программных систем логического управления технологическим оборудованием

Бурное развитие вычислительной техники привело к повышению эффективности работы систем логического управления и к существенному усложнению программно-аппаратного обеспечения. В связи с этим перед разработчиками систем встают вопросы: использования существующих вычислительных платформ персональных и одноплатных компьютеров; надежности, структурированности программно-математического обеспечения; использования в системе

внешних коммерческих решений; повторного использования программ логического управления; структурирования и анализа больших объемов данных (big data), получаемых с систем управления о ходе технологического процесса.

Реальная ситуация в области систем логического управления на сегодняшний день складывается следующим образом: нет полного представления о принципах организации систем на базе разных вычислительной платформы компьютеров (например, одноплатных компьютеров); об архитектуре программного и аппаратного обеспечения систем, которое базируется на достижениях в области вычислительной техники, а не на опыте использования оборудования конкретных разработчиков. Существующие архитектурные модели не представляют системных решений для использования внешних коммерческих программных модулей и механизмов повторного использования кода программ логического управления. Модульный принцип организации существующих систем не систематизирован, что не позволяет реализовывать новые системы на базе компоновки и конфигурирования существующих модульных решений. Под распределенным управлением в отношении систем логического управления зачастую понимают распределенное функционирования аппаратных модулей ввода/вывода, но не программно-аппаратных вычислительных средств.

Современные системы управления разрабатываются исходя из узкоспециализированных запросов конечных пользователей, что приводит к усложнению систем, а значит к увеличению затрат времени, повышению требований к квалификации разработчиков и избыточности конечного функционала системы управления. Указанные проблемы не позволяют конечным пользователям производить адаптацию систем управления под свои нужды. В случае необходимости решения конкретной проблемы они вынуждены привлекать специалистов фирмы-изготовителя или системных интеграторов. Решение проблемы возможно за счет создания методологии построения систем управления, в которой каждый из шагов однозначно определяется формализованной моделью и подкреплён инструментарием разработки. Это позволит получить открытое решение, которое предполагает осуществление доработки системы под конкретную задачу без дополнительной избыточности.

Сложившиеся десятилетиями традиции сформировали принципы построения определенного типа систем (ПЛК, PAC, Safety, Motion Control). Однако, на сегодняшний момент отсутствует единый подход к проектированию и реализацию указанного типа систем в сложных не тривиальных проектах, что не позволяет создать хорошо структурированную систему и требует пересмотра методологических основ проектирования систем логического управления в целом.

Остро стоит вопрос о систематизации методов проектирования программ логического управления, которые решают широкий круг задач на базе известных математических аппаратов.

Систематизация методов также предполагает объединение решения схожих задач в единые параметризованные библиотеки программ логического управления, которые в дальнейшем могут быть использованы для реализации систем логического управления в определенной области.

Развитие систем в области аппаратного обеспечения состоит в использовании базы вычислительных платформ компьютеров различного типа и стандартных модулей аппаратных входов/выходов, поддерживающих один из высокоскоростных протоколов коммуникации. Развитие в области программно-математического обеспечения состоит в реализации клиент-серверного подхода, где ядро логического управления выступает в роли сервера, а также модульной организации системы управления.

Проведенный анализ систем управления показал, что существует актуальная научная проблема, имеющая важное значение для народного хозяйства, заключающаяся в создании теоретических основ построения современных систем логического управления широкого назначения, имеющих открытую модульную архитектуру, гибкое и конфигурируемое программно-аппаратное обеспечение, а также создание методического обеспечения разработки указанного типа систем, направленного на систематизацию подходов к их проектированию.

1.6 Выбор технологического объекта для реализации системы логического управления

Во введении было отмечено, что различные типы систем управления (ПЛК, PAC, контроллеры безопасности и контроллеры движения) развивались самостоятельно и имеют традиции, которые сформировали обособленные принципы построения каждой из них. Все большее расширение функциональных возможностей систем позволяет применять единое решение при проектировании и реализации логической задачи управления вне зависимсоти от типа, применяемого оборудования. Такой подход наиболее актуален для нетривиальных задач, к которым можно отнести проектирование электроавтоматики сложных станков. Для систем ЧПУ станками выделяют ряд задач, среди которых есть логическая, которая позволяет автоматизировать многочисленные вспомогательные операции технологического обеспечения. К числу этих вспомогательных операций относятся: управление автоматической сменой инструментов; управление переключениями в приводах подач, связанными с ограничениями рабочей зоны; управление зажимными приспособлениями, охлаждением, смазыванием, ограждениями и др. Все указанные функции выполняются системой логического управления, которую ещё называют системой цикловой электроавтоматики, которая обеспечивает: подготовку к работе, работу станка в заданных режимах, индикацию состояний электрооборудования во всех режимах, выход из аварийной ситуации, защиту электрооборудования и др. К числу функций логической

задачи управления не относятся функции управления согласованным движением приводов, и в работе они не рассматриваются.

Далее рассмотрены некоторые типы станочного оборудования, для которых актуально применение единого подхода при проектировании систем логического управления.

Станки, использующие нетрадиционные способы обработки, такие как гидроабразивная (например, станки компании Bystronic (Швейцария). Гидроабразивная обработка в качестве режущего инструмента использует струю воды или смесь воды и абразивного материала под высоким давлением. Специфика работы системы ЧПУ заключается в управлении струей, контроле параметров подачи абразивного песка и давления, реализации вспомогательных М-команд управления электроавтоматикой.

При гидроабразивной обработке необходимо согласованное решение геометрической и логической задач. Также важными являются функции коррекции контура при формообразовании с учетом физических особенностей процесса резания материала струей:

- Коррекция погрешности из-за ширины реза – ширина реза, определяемая размером струи, задается оператором в системных параметрах. Алгоритм коррекции аналогичен эквидистантной коррекции на размер инструмента при механической обработке в плоскости, либо коррекции на радиус инструмента в пространстве.
- Коррекция отклонения от контура – в процессе резания материала, с уменьшением мощности резания, происходит отклонение струи в обратном направлении. Алгоритм корректирует скорость подачи и давление струи в зависимости от типа обрабатываемого материала и толщины заготовки.
- Коррекция отклонения от формы – согласно закону сохранения энергии, струя теряет энергетические характеристики на выходе из материала и стремится описать конус. Алгоритм корректирует давление струи, количество подаваемого абразивного материала при резании в соответствии с толщиной и типом обрабатываемого материала.

Особенностью решения логической задачи управления для данного типа станков является управление технологическими узлами, участвующими в процессе обработки. Для гидроабразивной обработки – это изменение режимов резания за счет управления режущей головкой, при котором можно задавать размер и давление струи и количество абразивного материала. В рамках логической задачи управления гидроабразивной обработкой реализуются вспомогательные М-команды, такие как: открытие и закрытие заслонки подачи воздуха в режущую головку, регулирование подачи абразива, включение и отлучение высокого давления при обработке.

При таком виде обработки особую роль играют системы управления безопасностью, которые осуществляют контроль за высоконагруженными и критичными узлами (например, станция высокого давления, СВД) и обеспечивают контурную безопасность станка при работе.

Среди контролируемых параметров СВД выделяют: критическое значение давления в системе; критическое значение температуры в системе; загрязнённость фильтрационного элемента очистки воды; время переключения поршня мультипликатора; давление воды на входе мультипликатора (должно быть не более 4 кгс/см^2). При превышении критического значения любого из контролируемых параметров, система управления должна остановить СВД и выдать предупреждение оператору.

СВД объединяют в единую сеть с логическим контроллером системы ЧПУ с использованием промышленных протоколов для обмена данными и командами. При этом СВД работает в рамках решения единой логической задачи управления.

Гибридные станки, в которых совмещаются несколько видов обработки, например, аддитивные или лазерные и традиционные технологии. Среди компаний, производящих указанные виды оборудования, можно выделить следующие: Matsuura Machinery, DMG Mori, Hybrid Manufacturing Technologies и др.

Гибридные и многозадачные станки можно оснащать однотипными системами ЧПУ, поскольку, управление механической и лазерной обработкой (при последовательной обработке или в гибридных решениях с одновременной обработкой) отличается только в синхронизации задач, которую, зачастую, обеспечивает логический контроллер.

Гибридные станки обеспечивают более эффективный и продуктивный способ обработки с помощью одновременного и контролируемого процесса взаимодействия механических узлов и(или) источников энергии. Обработка материала с лазерным нагревом срезаемого слоя (laser assisted machining), как процесс гибридной обработки, был разработан в целях сокращения производственных затрат при обработке керамики. Лазер используется для изменения структуры материалы в зоне резания.

В промышленном производстве все активнее находят свое применение новые технологии обеспечения качественных характеристик изделий. В частности, широкое распространение получило лазерное структурирование металла, как один из видов упрочнения поверхности изделий, прошедших механическую обработку, придания им высоких механических характеристик и высокого класса шероховатости. В связи с этим популярным в настоящее время становится совмещение на одном оборудовании механической и лазерной обработки, что позволяет значительно сократить цикл изготовления изделия за счет уменьшения операции и количества переустановив заготовки. При этом механическая и лазерная обработка выполняется по единой управляющей программе на одном оборудовании в рамках одной операции, что позволяет использовать одни и те же технологические базы и системы координат.

Особенностью решения логической задачи на таком типе станков является реализация синхронизации при управлении энергиями обработки, а также организация управления механизмами защиты ответственных узлов (например, защита лазерной головки от стружки и СОЖ во время механической обработки).

Многофункциональные обрабатывающие центры, использующие большое количество вспомогательного технологического оборудования, синхронизация работы которого и управление реализуется посредством логического контроллера.

Быстрое развитие систем ЧПУ позволило существенно расширить технологические возможности промышленного оборудования, способствуя эффективности производства и повышению качества изделий. В настоящее время все большую силу набирает тенденция создания многозадачных станков, позволяющих реализовывать на одном оборудовании различные виды обработки: токарно-фрезерные, токарно-фрезерно-шлифовальные и др. Такого рода оборудование значительно сокращает цикл изготовления продукции, в частности, за счет упрощения маршрутной технологии, значительного сокращения подготовительно-заключительного времени при выполнении операций и способствует повышению точности изготовления изделий за счет уменьшения количества переустановок заготовок и реализации принципа постоянства баз. Кроме того, многофункциональное оборудование позволяет оптимизировать производственную структуру предприятия без потери технологических возможностей за счет сокращения номенклатуры оборудования и производственных площадей.

Среди многофункциональных обрабатывающих центров можно выделить активно продвигаемые на рынок токарно-фрезерные обрабатывающие центры, предназначенные для обработки сложных деталей с минимально возможным временем выпуска одной детали. Преимущество токарно-фрезерных обрабатывающих центров по сравнению с обычными токарными станками достигается за счет оснащения станков высокоскоростным фрезерным шпинделем. В результате потребитель получает возможность осуществлять технологический процесс обработки деталей, имеющих сложную геометрию. При этом обеспечивается не только высокая точность готовых деталей, но сокращается количество необходимого оборудования и временные затраты на выполнение вспомогательных переходов: переустанов заготовки, подготовка оснастки и инструментов станка.

Рассмотрим некоторые виды обрабатывающих центров мировых производителей. Токарно-фрезерный обрабатывающий центр NT3100 DCG (фирма Mori Seiki) имеет вертикальную станину, позволяющую обеспечить недоступные ранее величины ходов фрезерной головки по осям X, Y, что существенно расширяет возможности фрезерной обработки на данных машинах. В приводах реализована технология Drive at the Center of Gravity и компоновка подвижных элементов Box-in-Box, что обеспечивает высокую жесткость, точность станка, скорость и чистоту

обработки. Специально разработанная Mori Seiki восьмиугольная конструкция направляющих для фрезерной головки исключает влияние термических деформаций на точность ее позиционирования.

Многофункциональный токарно-фрезерный центр Super NTJX (фирма Nakamura-Tome, Япония) комплектуется 2-мя шпинделями, 1-ой револьверной головкой и 1-им фрезерным шпинделем с автоматической сменой инструмента. Возможность одновременной обработки деталей в шпинделе и противошпинделе инструментами револьверной головки и фрезерного шпинделя обеспечивает сокращение циклов обработки.

Анализируя токарно-фрезерные обрабатывающие центры наклонной компановки можно отметить, что обычно они оснащены: продольными (X1, X2) и поперечными (Z1, Z2) осями верхнего и нижнего суппортов, основным шпинделем и противошпинеделем (с возможность продольного перемещения - ось Z) для обеспечения параллельной обработки. При этом две детали, закрепленные в шпинделе и противошпинделе соответственно, одновременно обрабатываются инструментами, установленными в нижнем суппорте и фрезерном шпинделе.

Оба шпинделя, установленные на токарно-фрезерном центре являются интерполируемыми, что позволяет устанавливать в них не только фрезерные, но и токарные инструменты. Схема одновременной обработки двумя инструментами деталей типа тел вращения как с одной стороны, так и с обоих сторон увеличивает производительность оборудования и срок эксплуатации резцов за счет компенсации радиальных составляющих сил резания.

Кинематическая схема обрабатывающих центров наклонной компоновки требует от системы ЧПУ реализации двухканального управления. Первый канал управляет верхним суппортом (оси X1, Y1), продольной осью Z1 и шпиндельным узлом W. За вторым каналом управления закреплены: нижний суппорт (оси X2, Z2), продольная ось Z и противошпиндел W1.

Систематизируя требования, предъявляемые к системам ЧПУ при управлении сложными технологическими объектами можно выделить группу общих требований, необходимых при работе с любыми станками, таких как: открытость, модульность, функция коррекции траектории движения, удаленное управление и др., так и группу частных требований, актуальных для конкретного оборудования и определяемых особенностью функционирования технологического оборудования. К частным требованиям станков наклонной компоновки отнесены:

- многоканальность. Обеспечивает параллельное выполнение нескольких управляющих программ в одной системе ЧПУ. Первый канал управляет верхним суппортом (оси X1, Y1), продольной осью Z1 и шпиндельным узлом W. За вторым каналом управления закреплены: нижний суппорт (оси X2, Z2), продольная ось Z и противошпиндел W1.

- многошпиндельная обработка. Наличие привода главного движения, противошпиндель и фрезерного шпинделя на инструментальной головке позволяет обрабатывать изделия одновременно несколькими инструментами. Реализация многошпиндельной обработки в системе ЧПУ предполагает специализированный механизм выявления коллизий в управляющих программах при одновременной обработке двумя шпинделями и наличие специализированной М-функции для передачи детали между шпинедлем и противошпинделем.
- набор специализированных токарных, фрезерных, сверлильных и измерительных циклов. Станочные циклы реализованы в виде параметризированных G-функций (токарные - G281 - 289, сверления - G81 - 89, фрезерные – G181 – 189, измерительные – G581-585), допускающих расширение со стороны станкостроителей. Использование циклов упрощает написание управляющей программы применяя реализованные технологии для выточки, обработки последовательности отверстий, а также для операции резьбонарезная, измерения параметров инструмента или обрабатываемой заготовки.
- специализированные вспомогательные М-функции. Обрабатывающие центры содержат большое число технологического оборудования: револьверные головки верхнего и нижнего суппортов, зажимные патроны, система охлаждения инструмента и станка, станция охлаждения и гидростанция, транспортер стружки, защитное ограждение, система подачи воздуха, инструментальный магазин и система автоматической смены инструмента и др. Управление перечисленным набором технологического оборудования требует реализации ряда вспомогательных М-функций для: включения и выключения транспортера стружки), зажима и разжима патронов, открытия и закрытия защитного ограждения, управление перехватом детали противошпинделем и др.

1.7 Выводы

1. Анализ современных систем логического управления показывает, что за последние годы существенно изменилась их: архитектура, потребительские свойства, аппаратное и программно-математическое обеспечение.
2. Выявлены тенденции развития систем логического управления, такие как: распределенный принцип построения, программная реализация контроллера, поддержка единого интерфейса взаимодействия устройств на базе технологии OPC, поддержка управления движением, удаленная диагностика и настройка по Internet, применение в качестве системного ПО ОСРВ,

применение вычислительных платформ общего назначения, применение технологии обеспечения безопасности, применение интеллектуальных аппаратных устройств ввода/вывода, использование высокоскоростных протоколов связи на базе технологии Ethernet.

3. Фирмы-производители систем логического управления можно разделить на крупные, средние и мелкие. Особенностью работы крупных производителей является принцип комплектной поставки средств автоматизации, такой подход позволяет устанавливать высокие цены, в том числе на комплектующие изделия. Системы автоматизации производителей средней величины отличаются большей унификацией и допускают открытость для интеграции решений сторонних разработчиков или конечных пользователей. Небольшие производители характеризуются направленностью на решение определенного класса задач и не имеют ресурсов для развития общепромышленных сфер применения систем автоматизации.

4. Проведенный анализ средств программирования систем логического управления показал, что ряд недостатков не позволяет применять описанные программные среды в кроссплатформенных проектах, это ориентация на конкретную операционную систему (чаще всего Windows), необходимость приобретения лицензии на коммерческое использование, закрытость системы, ориентированная на комплексное решение, но без возможности его модернизации и доработки. Ни один из проанализированных программных пакетов не поставляется с «открытым» программным кодом, что делает затруднительным их использование в качестве базы при разработке программного продукта.

5. Проведенный анализ систем управления показал, что существует актуальная научная проблема, имеющая важное значение для народного хозяйства заключающаяся в создании теоретических основ построения современных систем логического управления широкого назначения, имеющих открытую модульную архитектуру, гибкое и конфигурируемое программно-аппаратное обеспечение, а также создание методического обеспечения разработки указанного типа систем, направленного на систематизацию подходов к их проектированию.

Глава 2 Разработка теоретических основ для описания моделей построения программных систем логического управления технологическим оборудованием

Концепция построения систем логического управления с момента разработки первых устройств на базе релейных схем автоматики и до недавнего времени оставалась неизменной. Работы авторов, внесшие наиболее значительный вклад в развитие указанного типа систем управления, были рассмотрены в первой главе. Однако на сегодняшний момент появилось ряд существенных факторов, которые требуют пересмотра указанной концепции:

- многократное увеличение вычислительных ресурсов ЭВМ (с конца 90-х годов прошлого века до сегодняшнего дня произошел рост вычислительных ресурсов более чем на порядок);
- появление новых типов ЭВМ, таких как мобильные устройства и одноплатные компьютеры, которые могут служить основой для построения систем управления;
- появление принципов организации контроллеров (Soft PLC, PAC системы);
- популяризация идей интеграции производственных ресурсов.

Наиболее полно вопросы развития новых принципов организации систем на базе логических контроллеров рассмотрены в рамках их применения в составе систем ЧПУ такими учеными, как Сосонкин В.Л. [2-11], Мартинов Г.М. [4, 6-9, 13-14], Петров И.В. [24-25], Yoshinori Tsujido, D.H. Johnson, J. Stenerson и др. Стоит отметить, что работы по системам ЧПУ рассматривают логические контроллеры лишь в качестве инструментария для управления электроавтоматикой станков, что сильно сужает круг рассматриваемых вопросов и не позволяет полностью раскрыть потенциал, заложенный в новых принципах организации систем логического управления. В частности, недостаточное внимание уделено следующим вопросам:

- Работа с большим количеством каналов ввода/вывода, что типично для нефтегазовой, энергетической и химической промышленности. В некоторых областях применения ПЛК количество каналов ввода/вывода может исчисляться тысячами и десятками тысяч (например, электроэнергетика), а в рамках систем ЧПУ зачастую применяются системы логического управления с количеством каналов ввода/вывода, не превышающим нескольких сотен.
- Согласованная работа систем логического управления удаленных друг от друга на большие расстояния (например, системы автоматики в нефтегазовой области) и реализация многоуровневой архитектуры систем управления (например, в рамках принципа «ведущий-ведомый»). В рамках систем ЧПУ согласованная работа систем логического управления обычно организована в пределах одного цеха или предприятия.

- Рассмотрение вопросов работы систем логического управления в областях, имеющих специфические особенности управления процессами в рамках систем ЧПУ не актуальна, поэтому применение новых принципов организации систем в указанных областях плохо освещено. Это касается, например, систем терморегулирования, систем управления климатическими параметрами, систем управления химическими процессами и др.

Указанные факты не позволяют напрямую применить принципы, применяемые для решения логической задачи ЧПУ на широкий круг областей, где используются системы управления на базе логических контроллеров.

Кроме того, производители систем управления, компании интеграторы и конечные пользователи сталкиваются с широким разнообразием программных, аппаратных средств, инструментального окружения систем логического управления и широким кругом областей применения такого рода систем. Это приводит к разнообразию применяемых стандартов и технологий и предполагает применение дополнительного периферийного оборудования, необходимого для конвертации протоколов коммуникации и соединения разнородных участков в рамках единой сети предприятия, что увеличивает себестоимость и усложняет процесс внедрения новых и модернизации существующих систем управления. Каждый производитель контроллеров и систем автоматизации продвигает решения, основанные на собственной линейке продуктов, без учета особенностей функционирования систем управления в целом. Международные стандарты (международной электротехнической комиссии - IEC, международной организации по стандартизации - ISO) определяют лишь границы, в рамках которых необходимо проектировать системы логического управления, но не сами базовые принципы. При этом даже крупные производители не обладают полной гаммой решений в области построения систем управления, однако можно выделить признанных мировых лидеров, среди которых компании: Siemens, Rockwell, Mitsubishi, Bosch-Rexroth, B&R и др. Поэтому при формулировании новых принципов организации систем логического управления необходимо учитывать проблемы разнообразия стандартов и инструментальных средств. В основе предлагаемой в работе концепции лежат наиболее перспективные из ранее сформулированных идей, которые развиты с учетом: современных достижений и тенденций в области построения систем логического управления и смежных с нею областях; требований, предъявляемых конечными пользователями.

Иерархичность, сложность и специфика применения систем управления предполагают наличие особенностей их реализации, среди которых можно выделить: повышенные требования к надежности и открытости, громоздкость и плохая обозримость программного кода, необходимость постоянной эволюции и поддержки системы на протяжении всего жизненного цикла. Все указанные особенности были учтены при проектировании архитектуры систем логического управления.

2.1 Систематизация требований, предъявляемых к системам логического управления, обусловленных потребностями рынка

Систематизируя требования, предъявляемые к системам логического управления при управлении сложными технологическими объектами, можно выделить группу общих требований, необходимых при работе с любым оборудованием и группу частных требований, актуальных для конкретного оборудования и определяемых особенностями его функционирования. К общим требованиям, предъявляемым к системам, отнесены:

- *Открытость.* Возможность интеграции в систему управления готовых программных или аппаратных решений системными интеграторами или конечными пользователями (например, станкостроителями). Это позволяет получить новый функционал систем логического управления на уровне внедрения (например, система технического зрения), без привлечения серьёзных ресурсов системных разработчиков.
- *Мультипротокольность* контроллера на уровне модулей ввода/вывода. На сегодняшний день используемое на предприятиях оборудование не имеет единых стандартов по применяемым промышленным протоколам связи. Это может приводить к наличию в рамах одного предприятия технологического оборудования, связь с которым организована с применением различных протоколов передачи данных. Поддержка стандартных промышленных протоколов обмена данными (SERCOS, EtherCAT, ModBus, и т.д.) позволяет подключать имеющиеся на рынке аппаратные модули ввода/вывода с применением стандартных физических каналов связи (RS-232, RS-485, Ethernet и т.д.) и разрабатывать системы логического управления на базе промышленных протоколов, доступных на конкретном технологическом оборудовании.
- *Наличие встроенного и автономного решений*. Возможность установки систем логического управления как на базе персонального компьютера, например, для встраиваемого в ЧПУ решения, так и на аппаратной базе однокристальных микропроцессоров для автономного решения. Это требование необходимо для обеспечения решений в различных ценовых диапазонах, без изменения аппаратной и программной платформы.
- *Кроссплатформенность реализации.* Система логического управления должна быть ориентирована на работу в рамках различных ОС РВ (Windows RTX, Windows CE, Linux и т.д.). Указанный подход позволяет осуществлять переносимость исполняемого кода на различные типы аппаратных платформ, поддерживающих работу операционной системы. Это обеспечивает независимость программного решения без привязки к конкретному типу микроконтрол-

лера, выбирать необходимо лишь аппаратную платформу (персональный компьютер, микро-ЭВМ и т.д.). При этом возможна быстрая переориентация на новое аппаратное решение при снятии с производства существующей платформы или при появлении новых.

- *Модульность,* со слабой связанностью модулей системы между собой. Функционально связанные части системы управления группируются в законченные узлы — модули. При этом каждый модуль в системе имеет определенный функционал и набор интерфейсов и может быть без дополнительных затрат соединён с другими модулями системы, что позволяет сократить время, необходимое на разработку новых систем, за счет их компоновки из готовых (ранее разработанных и отлаженных) модулей.
- *Конфигурируемость*. Позволяет изменять возможности системы управления, путём комбинирования модулей, выполняющих различные задачи. При этом комбинирование модулей осуществляется за счет изменения конфигурации системы управления, без дополнительных дорогостоящих системных разработок. Указанное требование позволяет сократить время, необходимое на разработку новых систем и избежать избыточности в готовых решениях.
- *Масштабируемость,* позволяет сохранять способность системе управления работать при увеличенной нагрузке, связанной с добавлением ресурсов. А также возможность расширения размера системы управления при увеличении размера объекта управления. Выполнение этого требования позволит при увеличении нагрузки производить лишь переналадку системы управления, не прибегая к глубокой модернизации.
- *Распределенная реализация.* Возможность работы программных и аппаратных модулей системы в рамках локальной или глобальной сети на удаленных вычислительных ресурсах (например, обработка данных с удаленных модулей входов/выходов или удаленная диагностика и отладка системы посредством сети Интернет). Распределенная реализация повышает надежность систем за счет децентрализации и сокращения линий связи. Что позволяет уменьшить требования, предъявляемые к вычислительным ресурсам, находящимся в непосредственном контакте с объектом управления. При этом связь с объектом управления должны иметь лишь пассивные аппаратные модули ввода/вывода информации.
- *Надежность.* Это возможность системы сохранять в установленном диапазоне значения всех параметров длительный период времени, а также способность реализовывать необходимый функционал для определенных режимов работы при использовании, техническом обслуживании, хранении и транспортировании. Без обеспечения надежности система управления не пригодна для эксплуатации.
- *Ремонтопригодность.* При возникновении неисправностей возможность восстановления рабочих характеристик системы за минимальное время в промышленных условиях. Без

обеспечения требований ремонтопригодности эксплуатация системы управления не рентабельна исходя из требований экономической эффективности.

- *Безопасность.* Соответствие требованиям по технике безопасности при работах с оборудованием и технике промышленной безопасности, что регламентируется стандартом ГОСТ Р МЭК 61131-6-2016 (Контроллеры программируемые. Часть 6. Безопасность функциональная) [92]. Требования к безопасности регулируют безопасность персонала и технологического производства, выполнение полных требований может сопровождаться внедрением дорогостоящих производственных мероприятий. Система, не удовлетворяющая требованиям безопасности, не допускается до эксплуатации.
- *Модифицируемость.* Возможность перенастройки системы для работы с технологическим оборудованием или технологическими процессами иного типа, чем предусмотрено техническим заданием. Требование особенно актуально для промышленных систем, когда номенклатура выпускаемой продукции может существенно расширяться. Это обязывает производителей систем управления предусматривать возможности перепрограммирования и переналадки систем управления силами конечных пользователей (инженеров на предприятиях).
- *Максимальная длительность жизненного цикла* без существенных потерь от морального устаревания компонентов, показатель обеспечивается периодическим обновлением программных и аппаратных модулей, а также ориентацией при проектировании на промышленные стандарты, которые длительное время остаются актуальными.
- *Сохранение инвестиций, вложенных в проект.* Основной функционал систем логического управления сосредоточен в программных модулях, тогда как аппаратное обеспечение служит лишь средством реализации функционала. В этом случае развитие и модернизация происходит постоянно и не требует серьезных капиталовложений, в отличие от решений, где основной функционал заложен в аппаратную составляющую. Помимо этого, указанный подход позволяет сократить конечную стоимость встроенной системы управления за счет замены дорогостоящей аппаратной составляющей на программное решение.
- *Сокращение времени пуско-наладочных работ.* Время установки и пуско-наладочных работ систем логического управления на технологическом оборудовании уменьшается за счет применения программных решений, требующих лишь инсталляции и настройки. Уменьшение времени пуско-наладочных работ и уменьшение времени на разработку систем позволяет сократить себестоимость системы в целом.

При проектировании современных систем логического управления необходимо обеспечить соблюдение перечисленных требований, что позволит реализовать продукт, отвечающий современным потребностям рынка.

2.2 Разработка модульной организации структуры системы логического управления

Любая сложная система управления, в том числе и система логического управления, не может быть реализована в виде единого программно-аппаратного комплекса, решающего одновременно весь широкий спектр задач. При проектировании архитектуры систем производится их разбиение на модули, которые представляются в виде законченного блока с реализацией определенного функционала; его внутреннее состояние изменяется в процессе работы; он взаимодействует с другими модулями системы и конфигурируется под определенные задачи, стоящие перед конкретной реализацией системы управления. Каждый модуль – это автономное решение в рамках системы, что в конечном итоге определяет общий уровень гибкости и конфигурируемости системы в целом.

Модульную организацию систем логического управления стоит рассматривать как на уровне приложений, так и на уровне аппаратуры и системного программного обеспечения. На аппаратном уровне модульный подход применяется довольно широко, типичным примером этого служит архитектура ПК, состоящая из отдельных конечных элементов, соединенных между собой на базе материнской платы. В рамках системы логического управления принцип модульной организации аппаратного обеспечения должен быть сохранен, и в зависимости от требований объекта управления должна быть подобрана конфигурация, состоящая из базового вычислительного модуля, плат расширения (например, для организации промышленной сети), модулей аппаратных входов/выходов, панелей оператора, пультов управления и т.д.

Каждая система управления выполняется для конкретного объекта управления и поэтому является уникальной. В этом случае модульность структуры системного программного обеспечения приобретает особую актуальность, в частности на уровне операционной системы. Это позволяет осуществлять поддержку функционала и аппаратного обеспечения конкретной системы, например, за счет установки компонентов расширения реального времени, установка драйверов поддержки специализированного аппаратного обеспечения и т.д.

На уровне прикладных решений система управления разбивается на модули по функциональным признакам, при этом каждый конкретный модуль может разрабатываться независимыми группами инженеров. Отсутствие системного подхода при проектировании вызывает естественные трудности при объединении модулей в единое целое, которые зачастую решаются за счет множественных межмодульных связей. В дальнейшем это приводит к невозможности развития и модернизации системы.

Новый подход к модульной структуре систем логического управления предложенный в работе позволит выделить отдельные модули, которые будут иметь: структурное единообразие,

общий механизм межмодульных связей; при этом система управления в целом примет структурированный вид.

Такой поход позволит:

- организовать систему управления как в рамках единой вычислительной платформы, так и на нескольких платформах, путем разделения группы модулей по единому функциональному признаку. Это актуально при разработке распределенных систем;
- комплектовать систему управления в зависимости от реализуемой задачи как программными, так и аппаратными модулями, имеющими определенный функционал, который приоритетен при управлении конкретным объектом. Например, возможность использования модулей управления движением, аппаратных модулей, поддерживающих аналоговые сигналы и др.;
- возможность применения программных и аппаратных компонент различных производителей, в том числе сочетая их между собой. Например, применение в рамках единой системы аппаратных модулей ввода/вывода нескольких производителей, что актуально при модернизации систем управления.

Вычислительные ресурсы систем управления могут физически располагаться на одной, двух или трех ЭВМ. При однокомпьютерной технологии все ресурсы установлены на единую платформу ЭВМ; при двукомпьютерной технологии подсистема программирования устанавливается на отдельную ЭВМ, а остальные ресурсы располагаются на единой платформе; при трехкомпьютерной технологии подсистема программирования и web сервер выделяются на отдельные ЭВМ, OPC сервер и подсистема логического управления располагаются на одной ЭВМ. Увеличение количества вычислительных ресурсов ведет за собой увеличение стоимости системы. Это целесообразно в случае построения сложной системы управления, при работе с которой предъявляются повышенные требования к количеству вычислительных ресурсов.

Указанные подходы могут быть реализованы в виде одной из структурных схем: однотерминальная схема (рисунок 2.1); схема на базе единой аппаратной платформы (рисунок 2.2); схема с web-сервером (рисунок 2.3).

Во всех решениях терминал системы логического управления содержит подсистему программирования, а ядро логического управления содержит подсистему логического управления. При реализации однотерминальной схемы, система логического управления содержит один терминал, к которому посредством физических линий связи (проводных или беспроводных каналов) подключены несколько ядер логического управления.

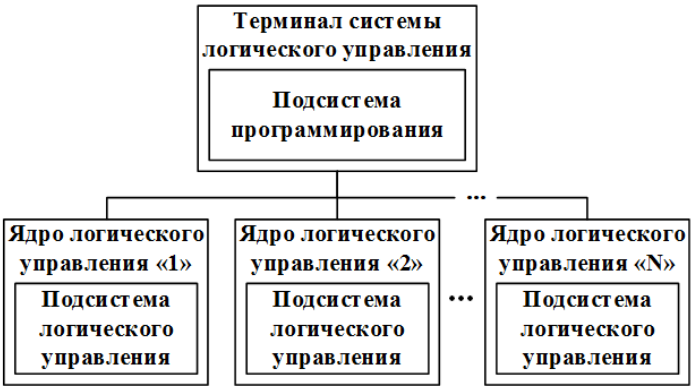


Рисунок 2.1 Схема системы логического управления с одним терминалом

Терминал в каждый конкретный момент времени может устанавливать логическое соединение только с одним ядром для осуществления загрузки или отладки программы логического управления. Частным случаем однотерминальной схемы является схема с одним терминалом и одним ядром логического управления. Применение однотерминальной системы возможно в случае установки терминала на мобильной аппаратной платформе (например, ноутбук), которая будет подключаться к конкретной системе управления в момент загрузки и отладки программы логического управления или посредством доступа терминала к системе внутри единой информационной сети цеха или предприятия.

Рисунок 2.2 Схема системы логического управления на базе единой аппаратной платформы

Реализация схемы на базе единой аппаратной платформы (однокомпьютерный вариант) предполагает, что терминал и ядро логического управления установлены на общую аппаратную платформу (например, персональный компьютер) и имеют постоянное логическое соединение в процессе загрузки и отладки программы логического управления. В качестве механизма для реализации коммуникационного канала при схеме с единой аппаратной платформой может использоваться разделяемая память между терминалом и ядром или программный интерфейс на базе сокетов (английское socket). Схема на базе единого терминала позволяет снизить стоимость системы за счет исключения второго компьютера, однако возникает проблема объединения терминальной задачи и задачи управления в рамках единой операционной системы.

Схема с web-сервером (2.3) имеет не менее трех аппаратных платформ, на которых располагаются: терминал, ядро системы управления и web-сервер. На Web-сервере установлена

база данных, которая накапливает информацию о состоянии объекта управления и может передавать её по сети Internet web-клиентам, расположенным в любой точке земного шара. Web-сервер может обслуживать несколько ядер логического управления, количество которых определяется мощностью аппаратного обеспечения, на котором установлен web-сервер.

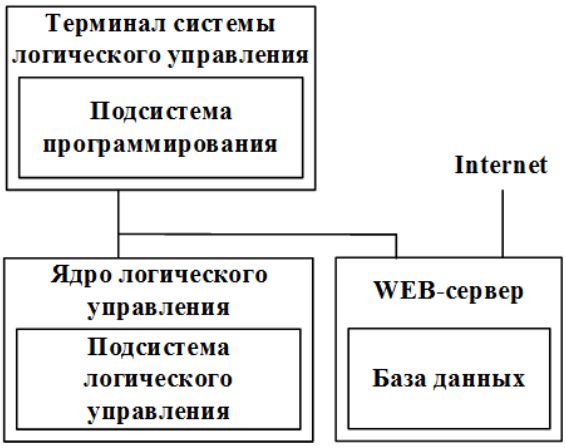


Рисунок 2.3 Схема системы логического управления с web-сервером

Предложенный подход к проектированию и реализации систем логического управления предполагает новый способ организации систем, в котором важную роль играет прикладная компонента, которая определяет характеристики системы с точки зрения конечного пользователя. Для реализации нового подхода к проектированию систем логического управления во второй главе работы поставлена цель – разработка конфигурируемой, модульной, распределенной архитектуры системы, каждый модуль которой должен обладать законченным функционалом и быть интегрирован в единое пространство системы управления. Для этого необходимо:

- сформировать представление о функционале систем логического управления с разработкой функциональной модели (раздел 2.3);
- определить иерархическую структуру системы логического управления (раздел 2.4);
- выявить основные потоки данных, пересылаемых между модулями (раздел 2.5);
- определить архитектурные решения в области построения систем логического управления (раздел 2.6);
- разработать механизмы преобразования программ логического управления на каждом из этапов реализации программ (раздел 2.7);
- разработать механизмы распределенного функционирования ситсемы логического управления (раздел 2.8).

2.3 Разработка последовательной схемы трансформации моделей системы логического управления

Системы логического управления являются системами со сложной иерархической структурой, достигнуть существенного упрощения в понимании принципов, лежащих в основе работы системы можно за счет выбора модельно-ориентированного подхода к разработке системы в целом и за счет разбиения структуры системы на ряд модулей, определив функционал модулей и предложив варианты межмодульных связей. По аналогии с модульным принципом построения систем управления (в частности, систем ЧПУ) в работе был применен подход, учитывающий возможности системы и последние достижения в области промышленной автоматизации, что позволило определить формальные модели, которые необходимы при разработке и объединить их в логическую схему последовательного их создания и трансформации. На рисунке 2.4 представлена схема последовательного создания и трансформации моделей систем логического управления технологическим оборудованием, которая позволяет наиболее полно описать и формализовать этапы разработки. В результате моделирования получим ряд основных модулей системы, определим их функционал и взаимосвязь между собой.

Различают два этапа моделирования – проектной и технологической подготовки. К проектной подготовке относятся функциональное моделирование, моделирование по типу виртуальной машины и потоковое моделирование. После этапа проектной подготовки необходимо определить среду реализации системы управления и средства производства.

На *первом этапе* в рамках построения функциональной модели проводится анализ требуемых потребительских свойств систем логического управления, исходя из которых выделяются основные функции и компоненты их реализующие.

На *втором этапе* создается модель по типу виртуальной машины, которая позволяет выстроить многоуровневую структуру системы логического управления с выделением межуровневых зависимостей между платформой и прикладной компонентой и обеспечивается портируемость посредством ввода кроссплатформенной библиотеки.

На *третьем этапе* создается потоковая модель данных (англ. Data Flow Diagrams – DFD), которая позволяет систематизировать основные потоки данных в системе логического управления технологическим оборудованием.

На этапе технологической подготовки разрабатывается архитектурная модель, модель подготовки и исполнения программы логического управления и распределенная модель. Далее реализуется процесс изготовления системы логического управления, в результате которого получается готовая система, отвечающая техническому заданию.

Рисунок 2.4 Схема последовательной трансформации моделей систем логического управления

На *четвертом* этапе строится архитектурная модель системы логического управления, которая определяет структуру системы и выделяет взаимные связи между объектами.

На *пятом* этапе строится модель подготовки и исполнения программы логического управления, которая определяет привязку компонент системы управления к фазам процесса управления.

На *шестом* этапе строится распределенная модель системы логического управления, которая позволяет определить схему её функционирования в условиях распределения ресурсов.

2.4 Разработка функциональной модели системы логического управления технологическим оборудованием в нотации IDEF 0

Системы логического управления решают широкий круг производственно-технических задач, среди которых можно выделить следующие: автоматизация технологического оборудования общепромышленного назначения, управление технологическими процессами, решение

логической задачи ЧПУ, управления робототехническими комплексами, управление движением и др. Проектирование, разработка и анализ работы постоянно усложняющихся систем широкого назначения, к которым относятся системы логического управления, требуют применения специализированных средств описания и анализа. В качестве инструментария первоначального исследования функционала и структуры систем управления предлагается использовать методологию IDEF0. IDEF – англоязычная аббревиатура от **I**CAM **Def**inition (Integrated Computer Aided Manufacturing – интегрированное автоматизированное производство). Выбранная методология позволяет произвести моделирование функционала системы управления с представлением модели в графической нотации. Ориентированность IDEF0 на соподчинённость объектов позволяет рассматривать логические отношения между функциями системы, без учета их последовательности во времени.

Для построения функциональной модели работы системы логического управления необходимо систематизировать и описать полный набор функций, реализуемых системой (таблица 2-1). Для этого необходимо выделить:

- входные и специализированные данные получаемые функцией,
- результат работы функции в виде выходных данных,
- модуль системы управления, который будет реализовывать функцию.

В результате проведенного анализа функционала систем логического управления была разработана функциональная модель системы в нотации IDEF0 (рисунок 2.5), в которой она представлена как набор функций, связанных между собой связями.

Таблица 2-1 - Систематизация функций, реализуемых системой логического управления

Функция	**Входные данные**	**Выходы**	**Спец. данные**	**Модуль СУ реализующий функцию**
Разработка программы логического управления	- Начальные условия	- Программа логического управления; - Конфигурация аппаратных входов/выходов;	- Техническое задание; - Принципиальная электрическая схема.	Среда разработки программ логического управления
Разработка пользовательской подпрограммы	- Описание пользовательского объекта	- Пользовательская подпрограмма	-	Среда разработки программ логического управления
Разработка конфигурации аппаратных входов/выходов	- Таблица привязки входов/выходов	- Конфигурация аппаратных входов/выходов	-	Модуль конфигурации аппаратных входов/выходов

Продолжение таблицы 2-1.

Функция	Входные данные	Выходы	Спец. данные	Модуль СУ реализующий функцию
Отладка программы логического управления	- Программа логического управления	- Ошибка	- Методика тестирования	Среда разработки программ логического управления
Выполнение программы логического управления	- Программа логического управления; - Конфигурация аппаратных входов/выходов; - Данные с аппаратных входов/выходов; - Ошибка	- Ошибка в работе программы; - Данные о работе системы; - Данные на аппаратные входы; - Программа логического управления.	-	Модуль реализации цикла логического управления
Сохранение программы в файловой системе	- Программа логического управления	- Программа логического управления	-	Файловая система
Визуализация работы системы	- Данные о работе системы	- Ошибка	-	Система диспетчерского управления
Обмен с системой управления верхнего уровня	- Данные о работе системы	- Ошибка	-	Система управления верхнего уровня
Удаленная диагностика	- Данные о работе системы	- Ошибка	-	Система удаленной диагностики и настройки
Обмен с аппаратными входами/выходами	- Данные на аппаратные входы	- Данные с аппаратных выходов; - Ошибка	-	Аппаратные входы/выходы

Каждая функция представляет собой «чёрный ящик» с указанием входов, выходов, специализированных данных и модуля системы, отвечающего за реализацию функции. Стрелки входов приходят в левую кромку активности функции, стрелки с указанием специализированных данных — в верхнюю кромку, стрелка с указанием модуля системы — в нижнюю кромку, стрелки выхода — в правую кромку. Цветом выделены следующие группы функций: фиолетовый – функции по разработке программ логического управления; зеленый – функции исполнительной системы (ядра) логического управления; желтым – функции внешних систем.

Функциональные моделирование позволило: выделить основные функции системы логического управления; привязать функции к компонентам системы, их реализующим.

В результате моделирования выделены следующие особенности систем логического управления:

- в работе систем выделены две этапа: разработка и исполнение программ логического управления;
- для разработки программы логического управления необходимо иметь техническое задание и принципиальную электрическую схему технологического оборудования, а также определиться с начальными условиями работы системы;
- разработанная программа логического управления должна содержать пользовательские подпрограммы и конфигурацию аппаратных входов/выходов;

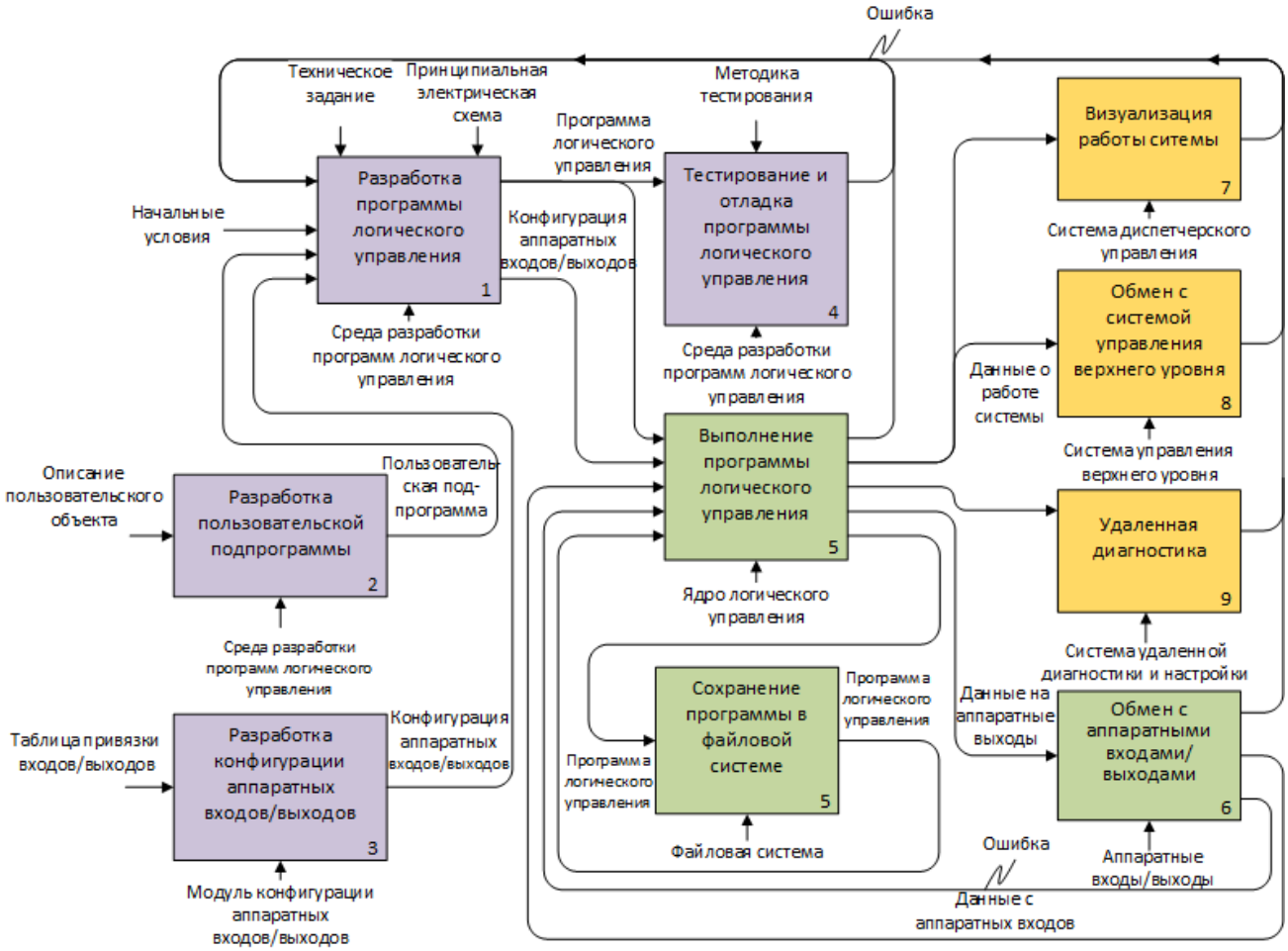


Рисунок 2.5 Функциональная модель системы логического управления IDEF 0

- конфигурация аппаратных входов/выходов создается на базе таблицы привязки входов/выходов технологического оборудования ко входам/выходам системы управления;
- проверка работоспособности системы производится по методике тестирования с применением специализированных стендов и тестовых программ;
- выполнение программы логического управления производиться в модуле реализации цикла логического управления;

- визуализация работы системы осуществляется в системе диспетчерского управления и сбора данных (SCADA - Supervisory Control And Data Acquisition);
- в качестве системы управления верхнего уровня может выступать система ЧПУ;
- для удаленной диагностики и настройки используется специализированное приложение, не входящее в основной пакет системы управления;
- все основные модули системы замкнуты обратной связью по ошибке.

2.5 Разработка модели системы логического управления технологическим оборудованием по типу виртуальной машины

Исходя из данных, полученных при функциональном моделировании, систему логического управления можно представить в виде двух частей: разработка, отладка и сопровождение программ логического управления (за это отвечает терминал) и реализация процесса управления (за это отвечает ядро системы). Разделение функционала на две отдельные независимые части позволяет устанавливать программное обеспечение терминала и ядра системы управления как на общую аппаратную платформу, так и на две независимые аппаратные платформы. Терминал может работать на операционной системе машинного времени, например, на ОС Windows. Ядро логического управления должно работать в рамках ОСРВ, например, на Linux RT. Схема работы указанных архитектурных решений описана в разделе 2.2.

При разработке систем автоматизированного управления, в том числе и систем логического управления используются следующие элементы: аппаратное обеспечение, системное программное обеспечение (операционная система, драйвера и др.), пользовательское программное обеспечение, прикладные компоненты. Объединить все используемые элементы с указанием связей между ними можно в виде вертикально расположенной многоуровневой модели, представляющей собой модель по типу виртуальной машины (рисунок 2.6).

Аппаратный уровень – это нижний уровень вертикальной модели систем логического управления. При двухкомпьютерном исполнении терминал устанавливается на аппаратной платформе персонального компьютера, ориентированном на нужды оператора, а ядро на компьютере промышленного исполнения или одноплатном компьютере, которые ориентированы на процесс управления. Компьютер ядра системы управления должен иметь: не менее нескольких аппаратных выходов Ethernet, возможность подключения специализированных плат для реализации высокоскоростных протоколов коммуникации (например, SERCOS) и приемлемую производительность для осуществления процесса управления в режиме реального времени. К

аппаратному уровню ядра системы управления также относятся модули аппаратных входов/выходов. Производитель системы может разрабатывать входы/выходы самостоятельно или использовать имеющиеся на рынке. При этом собственная разработка увеличивает себестоимость системы.

Рисунок 2.6 Модель системы логического управления по типу виртуальной машины

Выше аппаратного уровня размещается системный уровень, содержащий операционную систему и системное программное обеспечение. В качестве операционной системы терминала может выступать ОС Windows (версии XP и выше). В терминале используется стандартное системное ПО операционной системы, т.к. отсутствует специализированное аппаратное обеспечение. Ядро системы управления работает в рамках ОСРВ. В современных системах управления большое внимание уделяется возможности работы ядра в нескольких операционных системах (кроссплатформенность), поэтому в качестве возможных ОСРВ предлагается использовать Windows с расширением RTX (Real Time eXtension) или Linux RT. В подсистеме ядра системы управления широко используется специализированное аппаратное обеспечение, что требует разработки программных драйверов.

Доступ к службам операционной системы и к управлению устройствами предоставляется на уровне интерфейсов прикладного программирования (в англоязычной литературе API – Application Program Interface). Слой API индивидуален для каждой операционной системы и

чаще всего реализуется в виде динамических библиотек (Dynamic Link Library – DLL). Уровень интерфейсов прикладного программирования разделяет системный уровень и прикладные решения. Функциональность этого уровня может быть обновлена или дополнена без изменения верхних уровней, при условии сохранения интерфейсов.

Выше уровня интерфейсов прикладного программирования расположен уровень программной платформы, являющийся основой для исполнения приложений системы управления. В терминале в качестве программной платформы может применяться «.NET Framework» от компании Microsoft. В качестве базы платформа использует мультиязыковую среду исполнения Common Language Runtime (CLR). В качестве языков программирования могут применяться: C#, Visual Basic .NET, JScript .NET, C++/CLI, F#, J#, при этом функциональные возможности CLR доступны в любых языках программирования.

Возможность кроссплатформенной работы определяется на уровне программной платформы. Обеспечение совместимости достигается посредством классов-оболочек (wrappers) инкапсулирующих платформо-зависимые API вызовы (рисунок 2.7). Базовые классы, реализующие логику работы ядра системы управления, используют предоставляемые уровнем оболочек функции, что обеспечивает им платформонезависимую реализацию.

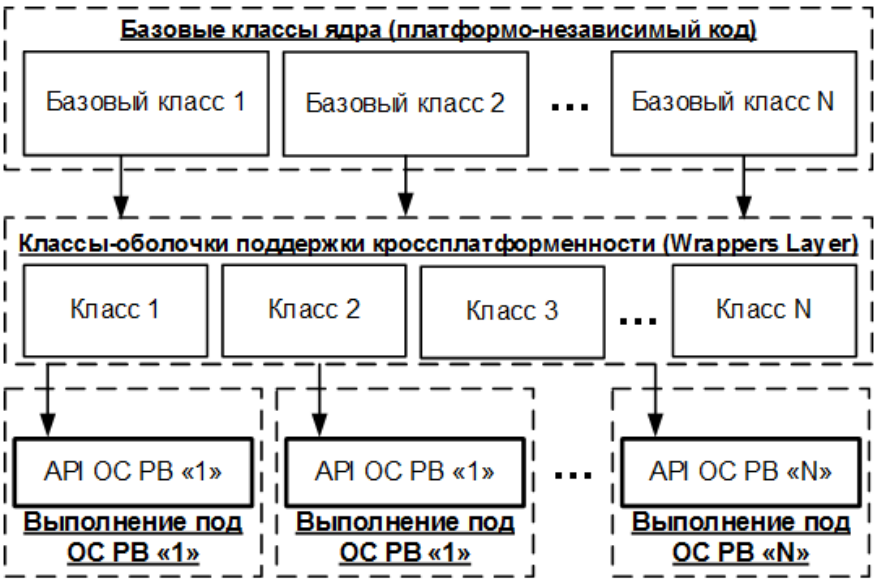


Рисунок 2.7 Структурная схема уровня программной платформы

Прикладной уровень содержит проблемно-ориентированные модули, среди которых можно выделить модули: реализации цикла логического управления, работы с разделяемой памятью, управления аппаратными входами/выходами, конфигурации и настройки параметров, верификации программ, а также пользовательские приложения (при наличии). Уровни виртуальной модели – это независимые уровни абстракции, которые можно рассматривать без привязки к другим уровням.

В этой связи можно выделить задачи, решаемые на каждом из уровней:

- аппаратный уровень – выбора компонентов, определение их физической совместимости;
- системный уровень – конфигурация операционной системы, разработка и установка драйверов специализированного оборудования (для ядра системы);
- уровень интерфейсов прикладного программирования – контроль доступа к системному уровню, определение поддерживаемых сервисных функций;
- уровень программной платформы – выбор программной платформы, поддержка кроссплатформенности;
- прикладной уровень – решение конкретных задач с использованием готовых решений и нижних уровней.

Многоуровневая модель по типу виртуальной машины позволяет:

- получить целостное представление о системе логического управления, без привязки к способу программного исполнения компонент прикладной области;
- проектировать системы управления с учетом следующих особенностей: кроссплатформенность, снижение времени на разработку, повышенное внимание к прикладным задачам, независимость приложения от объекта управления.

2.6 Разработка потоковой модели системы логического управления технологическим оборудованием

При проектировании информационных систем в качестве одного из инструментов структурного анализа целесообразно использовать потоковую модель данных, которую также называют диаграммой потоков данных. При проектировании систему управления рассматривают как набор связных между собой процессов, взаимодействие которых осуществляется посредством передачи данных. Данные могут приниматься извне от внешних объектов. Система управления содержит процессы, преобразующие информацию, в результате преобразования порождаются новые потоки данных. Потоки данных поступают на вход других процессов и передаются внешним объектам. Потоковая модель является многоуровневой иерархической моделью. Каждый процесс подвергается декомпозиции, т.е. разделению на отдельные подпроцессы, отношения между которыми показываются отдельной диаграммой в той же нотации. Потоковая модель — удобное средство для отображения разрабатываемой системы совместно с внешней средой. Это отображение описывается на исходной диаграмме верхнего уровня в иерархии потоковой модели (рисунок 2.8).

Назначение верхнего уровня — ограничение рамок системы и определение границы между разрабатываемой системой и внешней средой. На диаграмме верхнего уровня система логического управления представлена глобальным процессом «Управлять объектом» (круг), а внешние объекты и адресаты данных (прямоугольники) составляют окружение системы.

Рисунок 2.8 Исходная диаграмма потоковой модели системы логического управления

Стрелки с именами указывают на потоки данных – это позволяет моделировать передачу информации. При этом потоки данных могут быть одно- или двунаправленными. Процесс «Управлять объектом» преобразует информацию, полученную с входных потоков и порождает новые потоки данных поступающие на выход. К внешним объектам, по отношению к системе логического управления относятся: система диспетчерского управления (SCADA), система удаленной диагностики и настройки, система управления верхнего уровня (например, ЧПУ), файловая система и аппаратные входы/выходы.

Диаграмма, представленная на рисунке 2.9, отображает декомпозицию процесса «Управлять объектом» на два других, которые соответствуют терминалу (процесс «Разработать программу логического управления») и ядру системы (процесс «Управлять электроавтоматикой»). В результате моделирования потоков взаимодействия с внешней средой порождаются потоки между двумя подсистемами.

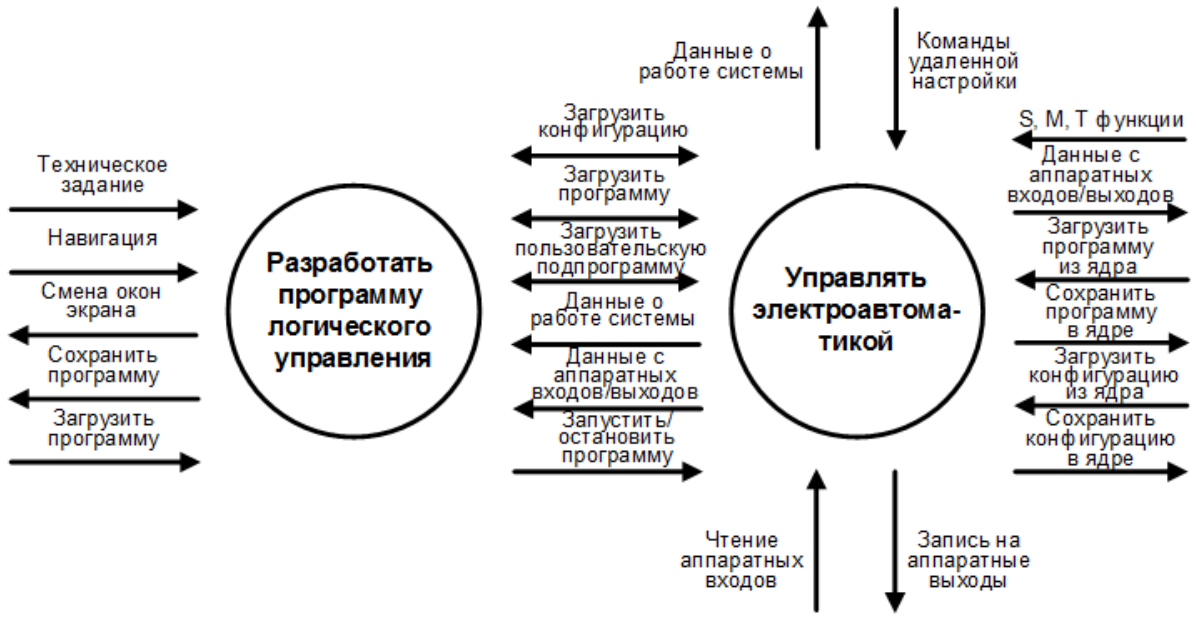


Рисунок 2.9 Декомпозиция процесса «Управлять объектом»

Диаграмма на рисунке 2.10 представляет декомпозицию процесса «Разработать программу логического управления».

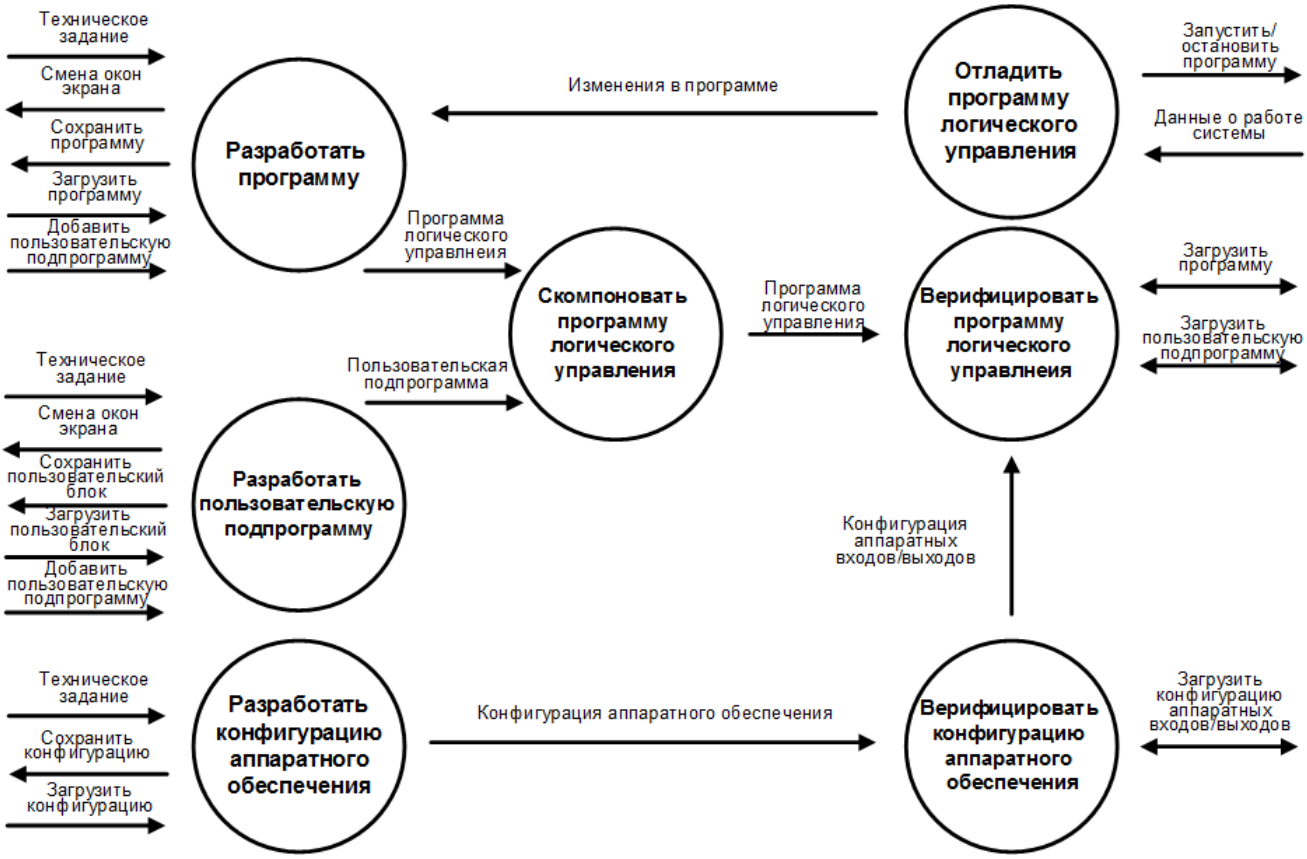


Рисунок 2.10 Декомпозиция процесса «Разработать программу логического управления»

Процесс разработки программы логического управления инициализируется данными с внешних источников. В разработке программы логического управления задействованы

следующие подпроцессы: разработать программу, разработать пользовательскую подпрограмму, скомпоновать программу логического управления, верифицировать программу логического управления, отладить программу логического управления, разработать конфигурацию аппаратного обеспечения, верифицировать конфигурацию аппаратного обеспечения.

В ходе разработки программы логического управления инициализируются новые потоки данных, которые передают информацию в ядро системы управления.

Декомпозиция процесса «Управлять электроавтоматикой» представлена на рисунке 2.11. Поток данных с терминала запускает процессы «Верифицировать программу логического управления» и «Сопоставить конфигурацию с реальными устройствами». Эти процессы осуществляют подготовку и преобразование данных для запуска процессов «Выполнить цикл работы логического контроллера» и «Распределить разделяемую память», отвечающих за исполнение программ логического управления. Процесс «Записать/считать данные из разделяемой памяти» отвечает непосредственно за управление и выдает поток данных на оборудование.

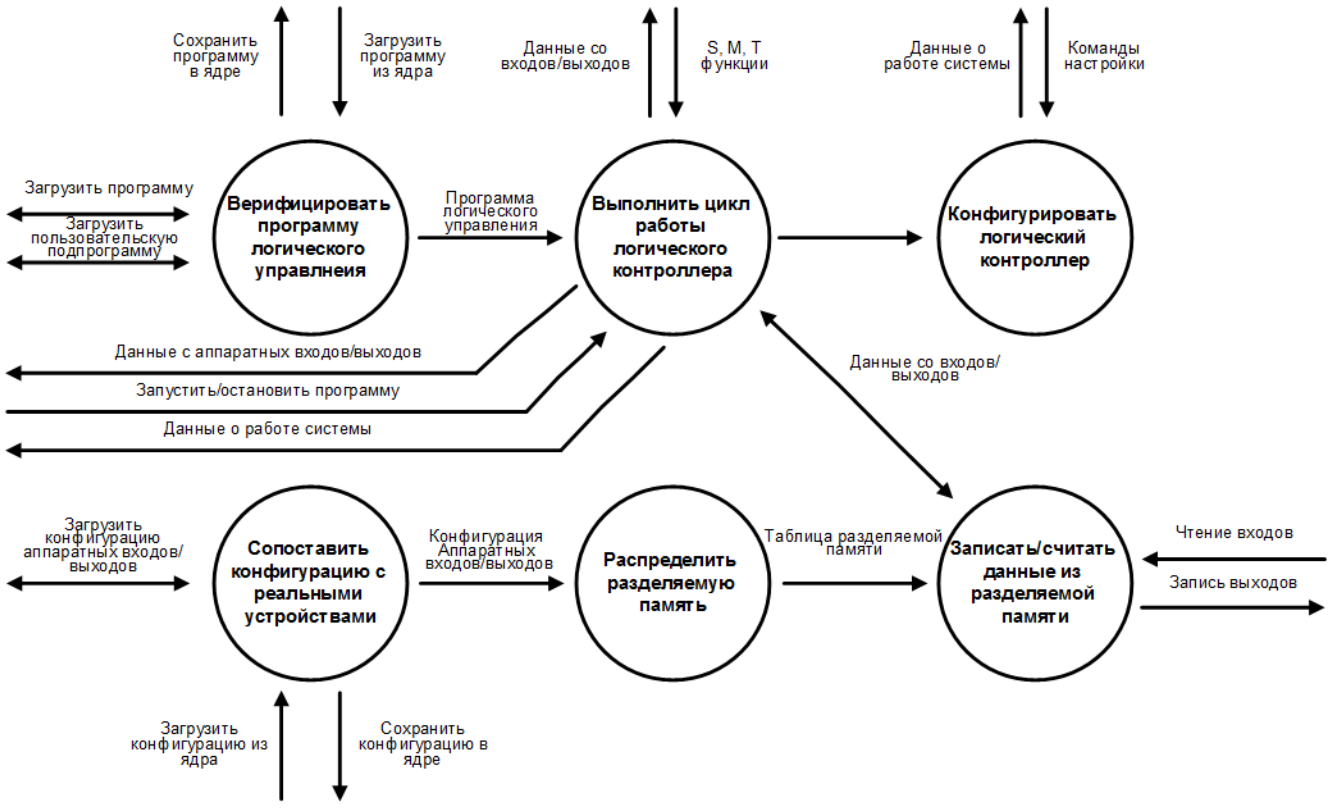


Рисунок 2.11 Декомпозиция процесса «Управлять электроавтоматикой»

Каждый из процессов, представленных на рисунках 2.10 – 2.11, может быть подвергнут более глубокой декомпозиции, которая не отражена в работе.

Потоковая модель позволяет смоделировать процесс трансформации входных потоков данных в выходные потоки. Декомпозиция начальной модели системы логического управления

позволяет смоделировать процессы и потоки данных. Однако декомпозицию начальной модели стоит производить с высоким уровнем абстракции, что позволяет получить модель без привязки к программной реализации. Структура, семантика и типы потоков данных полученные при моделировании используются в дальнейшем при построении архитектурной модели.

2.7 Разработка архитектурной модели системы логического управления технологическим оборудованием

С точки зрения системного анализа концептуальное моделирование позволяет выявить количество и структурное расположение модулей, объектов, факторов и явлений, полный набор которых позволит реализовать поставленные перед системой цели и задачи. Одним из вариантов концептуальной модели систем управления является архитектурная модель.

Описание архитектуры программных систем регламентируется стандартами, среди которых можно выделить: IEEE 1016-1998 Recommended Practice for Software Design Descriptions (методические рекомендации для описания программных решений), IEEE 1471-2000 Recommended Practice for Architectural Description of Software-Intensive Systems (методические рекомендации для описания архитектуры программных систем) [93-94]. Согласно указанным стандартам, архитектура это набор основных принципов организации системы, реализованных в виде набора ее компонентов, межкомпонентных связей и связей с окружением системы, а также принципов проектирования и развития системы. Архитектура системы логического управления, представленная на рисунке 2.12, состоит из двух подсистем: программирования и логического управления.

Подсистема программирования организована на основе модульного подхода и решает задачу проектирования и реализации программы логического управления. Диалог с пользователем организован на базе графического интерфейса пользователя (англ. GUI - Graphical user interface), за реализацию которого отвечает «Модуль отображения интерфейса пользователя».

Задача разработки программы логического управления разделена на две составляющие:

- *Реализация алгоритма работы системы управления.* За это отвечают модули, объединенные под единым понятием «Среда разработки программ логического управления». Среда разработки позволяет осуществлять: визуальное отображение, отладку и верификацию программ логического управления, а также добавлять в проект пользовательские подпрограммы, оформленные в виде отдельных библиотек функций.
- *Конфигурирование аппаратных входов/выходов,* которое определяет «модуль конфигурирования аппаратных входов/выходов».

Вне подсистемы программирования хранятся: программа логического управления в формате «*.fbv», библиотеки пользовательских подпрограмм в формате «*.fbl», конфигурация аппаратных входов выходов в формате «*.xml» и описание набора графических компонент в формате «*.xml».

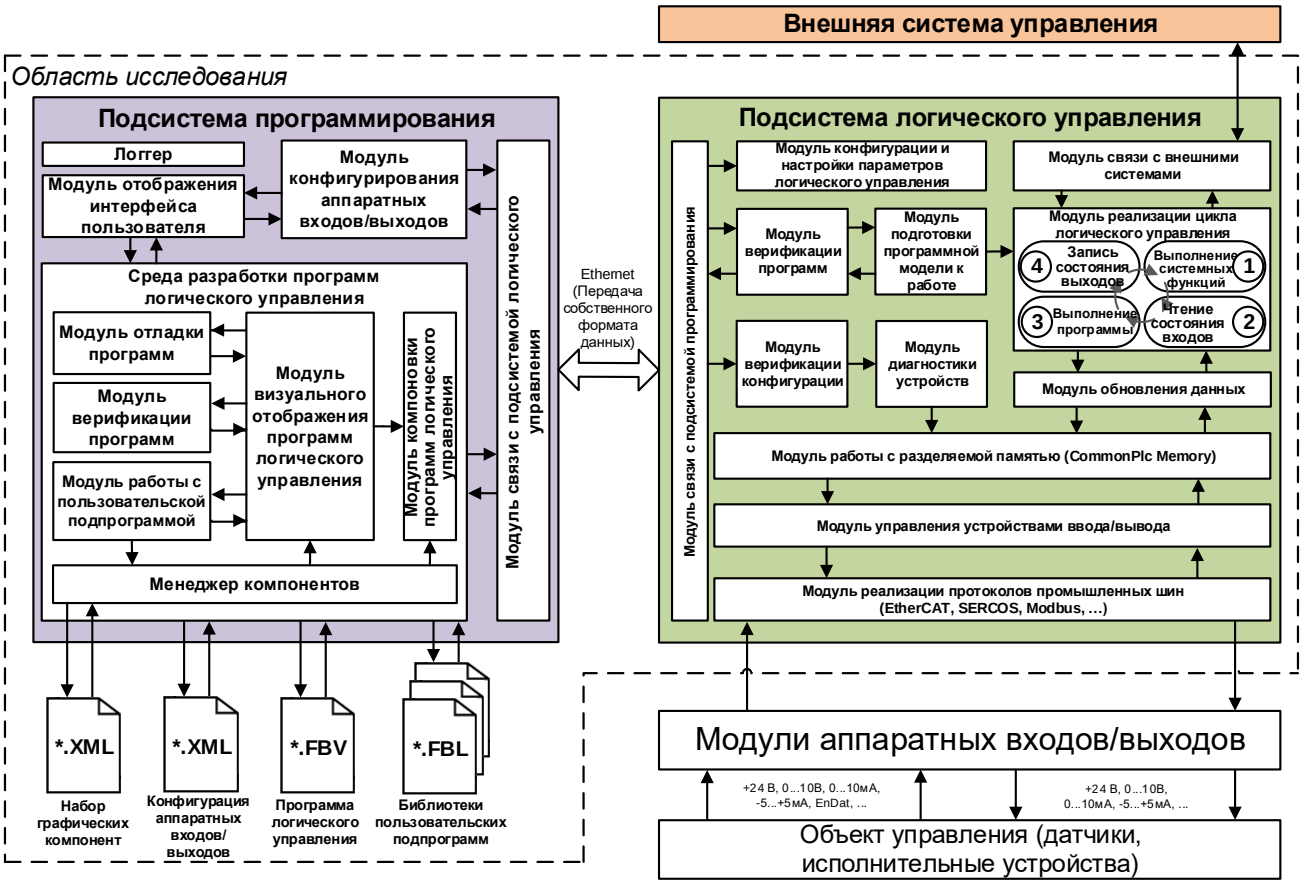


Рисунок 2.12 Архитектурная модель системы логического управления

Подсистема программирования связана с подсистемой логического управления посредством коммуникационного канала. При расположении подсистем на различных аппаратных платформах реализация коммуникационного канала осуществляется на базе стандарта Ethernet.

Центральным модулем подсистемы логического управления является «модуль реализации цикла логического управления», в рамках которого запускается классический цикл работы контроллера. До запуска в цикле управления программа проходит предварительную верификацию и подготовку программной модели к работе. При записи на аппаратные выходы данные последовательно передаются между модулями: «работы с разделяемой памятью», «управления устройствами ввода/вывода» и «реализации протоколов промышленных шин». При чтении с аппаратных входов данные проходят модули в обратном порядке.

Головные устройства внешних модулей аппаратных входов/выходов связаны с подсистемой логического управления посредством одного из высокоскоростных протоколов связи

(SERCOS, EtherCAT и др.). Головные устройства дополнительно комплектуются слотами, которые реализуют различные виды аппаратных входов и выходов, например, дискретных входы или выходы (сигналы уровня +24 В), аналоговые входы или выходы (сигналы уровней 0...10 В, 0... ±10 В, 1...5 В и 4...20 мА, 0...20 мА), входы подключения термосопротивлений, входы подключения термопар и др. Модули аппаратных входов/выходов соединены физическими линиями связи (проводами) непосредственно с датчиками и исполнительными устройствами объекта управления.

Построение архитектурной модели системы управления – это один из ключевых моментов проектирования, который позволяет:

- определить модульную структуру системы;
- произвести моделирование взаимоотношений модулей друг с другом и внешней средой;
- определить возможные пути масштабирования и эволюции системы;

Результаты моделирования архитектуры системы логического управления могут быть использованы в следующих целях:

- *Анализ альтернативных проектов системы.* При построении архитектурной модели системы управления важно помнить, что создаваемая система может применяться для смежных задач управления. При этом хорошо структурированная и продуманная архитектурная модель позволяет проанализировать возможности реорганизации системы управления для получения нового функционала.
- *Проектирование и разработка отдельных модулей системы*. Архитектурная модель содержит сведения о декомпозиции системы управления на уровне модулей. Это позволяет каждый модуль рассматривать как независимый элемент и вести его разработку с учетом его внешних связей.
- *Планирование перепроектирования системы*, изменение в ее организационной структуре. При внесении изменений в структуру системы управления в процессе эксплуатации, архитектурная модель позволяет оценить возможность проведения модернизации и служит базой для оценки экономической эффективности модернизации.
- *Определение критериев приемо-сдаточных испытаний системы при ее запуске в эксплуатацию* с разработкой проектной документации по ее использованию и сопровождению, а также учебные и маркетинговые материалы.
- *Планирование бюджета и использования других ресурсов в проекте при разработке, сопровождении и эксплуатации системы.*

- *Проведение анализа и оценка качества работы системы.* В качестве основы для планирования проведения анализа качества работы системы может применяться как на этапе проектирования, так и в процессе эксплуатации системы.

2.8 Разработка модели подготовки и исполнения программы логического управления

При проектировании программно-математического обеспечения систем логического управления необходимо помнить о том, что реализуемая система является гибкой и перепрограммируемой. Поэтому важную роль при реализации конечного проекта системы управления конкретным технологическим оборудованием играет программа логического управления. Механизм преобразования программы логического управления технологическим оборудованием в процессе работы системы основан на модели трансформации, представленной на рисунке 2.13. Разработка программы логического управления разделяется на два независимых процесса: разработка программы на языке FBD и разработка конфигурации аппаратных входов/выходов. Каждый из этих процессов проходит фазы: подготовки, исполнения и управления.

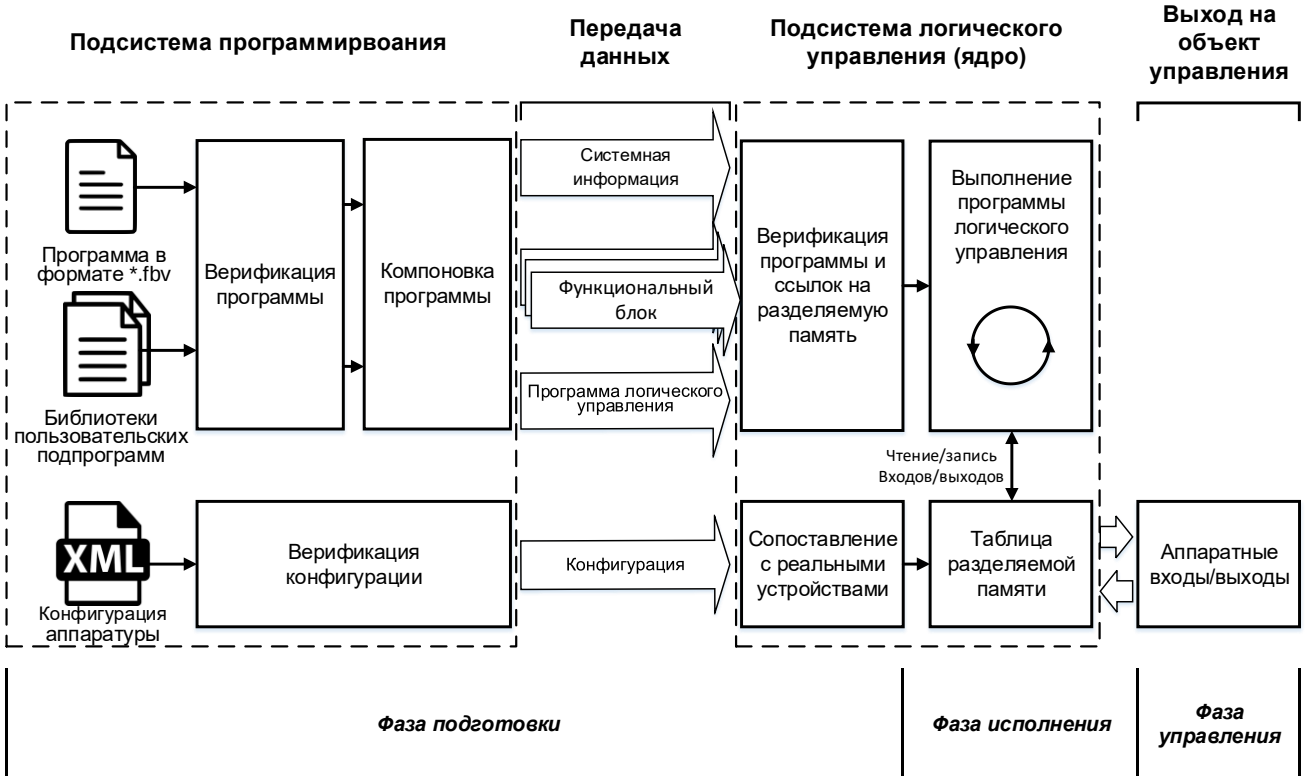


Рисунок 2.13 Модель подготовки и исполнения программы логического управления

Подготовка программы логического управления. Программа логического управления состоит из основной программы, которая хранится в отдельном файле файловой системы с расширением «*.fbv», и множества библиотек пользовательских подпрограмм, хранящихся в

файлах с расширением «*.fbl». В подсистеме программирования предварительно проводится верификация программы и пользовательских подпрограмм на: наличие ошибок, отсутствие связей, согласование типов данных и наличие других нерегулярных ситуаций. При успешном прохождении процедуры верификации происходит компоновка программы, в процессе которой программа логического управления объединяется с пользовательскими подпрограммами. В результате компоновки получается единая программа логического управления.

На следующем шаге производится передача подготовленной программы в подсистему логического управления, которая осуществляется поэтапно:

- Первый этап - передача системной информации. Этап необходим для определения размеров резервируемых машинных ресурсов под передаваемую программу в подсистеме логического управления. На этом этапе передаются данные о размерах объединенной программы логического управления, о размерах каждого из блоков пользовательских подпрограмм и другая системная информация.
- Второй этап – передача отдельных модулей пользовательских подпрограмм. На этом этапе каждая из пользовательских подпрограмм передается отдельно. В процессе передачи сначала пересылается справочная информация, включая размер занимаемой памяти, далее пересылается сама подпрограмма.
- Третий этап – передача объединенной программы логического управления.

В ходе работы системы программа логического управления может быть передана в обратном направлении, от подсистемы логического управления к подсистеме программирования. В этом случае для обеспечения возможности сохранения не только программы логического управления, но и пользовательских подпрограмм, необходимо иметь пользовательские подпрограммы в первоначальном виде (до компоновки), поэтому в механизме передачи данных в ядро логического управления предусмотрен второй этап.

При получении объединенной программы логического управления подсистема логического управления осуществляет верификацию ссылок на разделяемую память, которые прописываются в функциональных блоках входов и выходов программы. При отсутствии ошибок производится запуск программы в цикле работы логического контроллера.

Подготовка конфигурации аппаратного обеспечения. Конфигурация аппаратных входов/выходов хранится в файловой системе подсистемы программирования в формате «*.xml» и содержит сведения о модулях аппаратных входов/выходов, подключенных к системе управления. В конфигурации помимо количества модулей определяются: порядок их подключения, их расположение в разделяемой памяти и размер занимаемой ими памяти.

Перед началом работы системы логического управления конфигурация проходит процесс верификации на наличие ошибок. При успешном прохождении верификации конфигурация передается в ядро логического управления. В ядре логического управления конфигурация сопоставляется с реальными устройствами, подключенными к системе управления. При отсутствии ошибок, формируется таблица разделяемой памяти, которая содержит группы ячеек памяти, закрепленных за отдельными устройствами аппаратного ввода/вывода.

Работающая программа логического управления осуществляет чтение данных о состоянии входов из разделяемой памяти и запись обновленных данных о состоянии выходов в разделяемую память. Разделяемая память с периодичностью в один такт работы ядра логического управления синхронизируется с аппаратными входами /выходами, что позволяет осуществить запись на аппаратные выходы и произвести чтение с аппаратных входов. Таким образом осуществляется непосредственно фаза управления.

2.9 Разработка распределённой модели системы логического управления технологическим оборудованием

При проектировании и реализации современных систем управления наметилась тенденция разделяемого использования внешних вычислительных ресурсов, находящихся как внутри локальной вычислительной сети предприятия, так и в глобальной сети Интернет. Вместе с тем для согласованной работы отдельных модулей системы между собой предполагается построение хорошо структурированной распределенной модели работы системы управления.

Зачастую системы логического управления рассматриваются как независимые системы для решения отдельных задач контроля и управления, при этом встраивание разработанных систем в общую информационную структуру предприятия происходит после решения локальных задач управления объектами. Однако, для эффективного управления сложными системами, такими как современные крупные предприятия, необходимо применение иерархического подхода на начальных этапах проектирования, что позволяет согласовать цели и задачи управления между уровнями иерархии и определить инструментарий и механизмы взаимодействия между уровнями систем управления на начальных этапах проектирования. В случае систем логического управления увеличение количества датчиков, рост территории охвата системы, увеличение количества и видов решаемых задач, а также значительное усложнение алгоритмов работы вызывает потребность оптимального использования вычислительных ресурсов. Этого воз-

можно достичь, используя распределенную схему работы системы. При этом возникает необходимость выделения задач, которые можно решать на отдельных программно-аппаратных модулях распределенной системы управления.

На сегодняшний день задача распределенного функционирования систем управления на базе ПЛК сводится к задаче разделения ресурсов на нижнем уровне, которые взаимодействует с объектом управления. Но в современных условиях цифрового производства целесообразно использовать вычислительные мощности верхних уровней управления и удаленные вычислительные мощности.

При построении систем логического управления часто возникает необходимость подключения стандартного технологического оборудования, для этого необходимо иметь унифицированный интерфейс связи. В частности, большинство производителей аппаратного обеспечения систем управления на сегодняшний день поддерживают технологию OPC. Указанная технология особенно актуальна при внедрении оборудования сторонних производителей.

Все большую популярность получают web-технологии, которые позволяют перенести задачу распределенного управления с уровня локальной вычислительной сети предприятия на уровень глобальной сети Интернет.

Распределенная модель системы логического управления позволяет выявить:

- связи с объектами управления посредством аппаратных модулей ввода/вывода, которые берут на себя задачи сбора и предварительной обработки данных;
- необходимое стандартное оборудование сторонних производителей;
- необходимое специализированное оборудование, разработанное для конкретной системы управления;
- способ интеграции с системами управления верхнего уровня;
- взаимодействие с системами, находящимися на значительном удалении от объекта управления с применением web технологий.

Для построения распределенной модели системы логического управления были систематизированы функции, выполняемые на удаленных вычислительных ресурсах. (Таблица 2-2)

Таблица 2-2 Задачи организации распределенной работы системы логического управления

Задача	**Решение**	**Пример**
Распределенное функционирование аппаратных устройств нижнего уровня.	- Поддержка высокоскоростных протоколов связи; - Работа с портами ввода/вывода; - OPC технология.	- Подключение специализированного оборудования; - Подключение стандартного оборудования; - Подключение аппаратных входов/выходов.

Продолжение таблицы 2-2.

Задача	Решение	Пример
Распределенный многотерминальный доступ.	Применение web технологий.	Реализация терминала на базе мобильных устройств.
Распределенное выполнение задач во внешних системах управления.	- Применение web технологий; - Работа с вычислительными ресурсами в рамках локальных сетей.	- Организация работы с системой управления верхнего уровня; - Организация работы со SCADA системами; - Организация работы с системой удаленной диагностики и настройки.

На основе проведенного анализа предложена распределенная модель системы логического управления (рисунок 2.14).

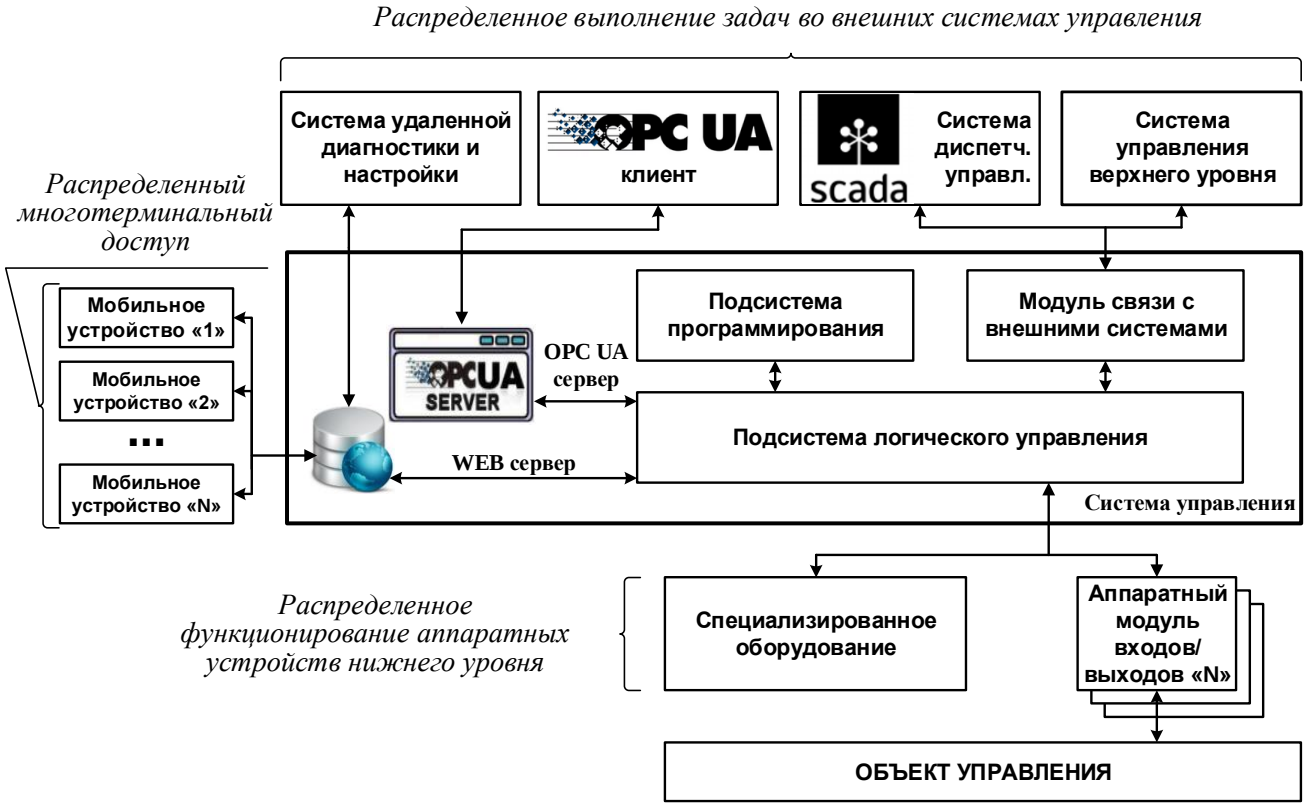


Рисунок 2.14 Распределенная модель системы логического управления

Вычислительные ресурсы системы логического управления располагаются в области, выделенной жирной линией, к ним относятся:

- подсистема программирования, позволяющая проектировать, разрабатывать и отлаживать программы логического управления;
- подсистема логического управления, отвечающая непосредственно за выполнение алгоритмов управления и связь с аппаратными устройствами;
- OPC сервер, позволяющий подключать стандартное оборудование, поддерживающее технологию OPC (например, дополнительный контроллер);

- Web сервер, позволяющий подключать внешние ресурсы, в том числе по глобальной сети Internet.

На нижнем уровне модели представлено распределенное функционирование аппаратных устройств. Специализированное оборудование, разработанное для конкретной системы управления, может подключаться непосредственно к подсистеме управления по стандартному порту или протоколу связи (например, специализированные панель управления или автоматизированные приспособления). Стандартное оборудование, выбранное при проектировании системы управления может подключаться посредством технологии OPC UA. Аппаратные модули входов/выходов подключаются к подсистеме логического управления на базе одного из высокоскоростных протоколов связи (например, SERCOS, EtherCAT, Modbus TCP или др.). При этом к одной системе логического управления могут быть подключены различные типы аппаратных входов/выходов, работающих на разных протоколах связи.

Посредством web-сервера можно реализовать распределенный многотерминальный доступ к системе управления, в том числе используя мобильные устройства. Это особенно актуально, если система разнесена на большие расстояния. В этом случае для доступа к функционалу системы из удаленных точек, не имеющих собственного терминала управления, можно использовать переносные или мобильные устройства с установленными на них мобильными терминалами управления.

Функция распределенного выполнения задач на внешних системах управления позволяет подключать к работе вычислительные ресурсы систем управления верхнего уровня или удаленные вычислительные ресурсы. При этом системы управления верхнего уровня, находящиеся в рамках локальной вычислительной сети предприятия, подключаются непосредственно к подсистеме логического управления, а удаленные вычислительные ресурсы посредством web сервера. На удаленных вычислительных ресурсах может работать, например, система диагностики и настройки. При этом специалисты могут диагностировать объект и систему управления дистанционно, в том числе и во время её работы.

Построение распределенной модели системы логического управления позволило:

- Выделить вычислительные ресурсы системы логического управления, к которым относятся подсистема программирования, подсистема логического управления, OPC и web-сервер.
- Установить, что вычислительные ресурсы могут находится как на единой аппаратной платформе ЭВМ, так и быть разнесены.
- Выделить три уровня задач в организации распределенной работы системы управления: распределенное функционирование аппаратных устройств нижнего уровня, распределенный многотерминальный доступ, распределенное выполнение задач во внешних системах управления.

- Установить механизмы взаимодействия вычислительных ресурсов СЛУ и распределенных вычислительных ресурсов.

2.10 Выводы

1. Существующие концепции построения систем логического управления не учитывают факторы, возникшие с развитием информационных технологий за последние годы, среди которых: многократное увеличение вычислительных ресурсов ЭВМ, появление новых типов ЭВМ, таких как мобильные устройства и одноплатные компьютеры, появление новых форм организации контроллеров и популяризация идей интеграции производственных ресурсов.
2. Предложена новая концепция конфигурируемой, модульной, распределенной архитектуры системы логического управления, ориентированная на максимальное использование аппартных возможностей современных вычислительных систем, отличающаяся свойством кроссплатформенности, каждый модуль которой имеет законченный функционал и может быть интегрирован в единое пространство системы управления.
3. Определение набора функций системы логического управления позволяет закрепить конкретную функцию или набор функций за модулем или набором модулей системы.
4. Предложенная вертикальная структура системы логического управления по типу виртуальной машины с выделением уровней абстракции позволила обеспечить независимость программных решений от аппаратного обеспечения и прикладных задач.
5. Систематизация потоков данных системы логического управления произведена с высоким уровнем абстракции, что позволило получить потоковую модель без привязки к программной реализации.
6. Предложенная архитектура системы логического управления выполнена в виде двух взаимосвязанных подсистем, что позволяет разделить задачи программирования и исполнения алгоритмов логического управления.
7. Предложенная распределенная модель системы логического управления позволяет задействовать не только ресурсы аппаратного уровня, приближенные непосредственно к объекту управления, но вычислительные мощности верхних уровней управления и удаленные вычислительные мощности, за счет применения стандартных средств локальных вычислительных сетей и web-сервера.
8. Предложенный подход к проектированию модульной системы логического управления позволяет получить открытую, масштабируемую систему с возможностью поддержки мультипротокольности и кроссплатформенности.

Глава 3 Создание формального аппарата построения подсистемы программирования и исполнительного ядра системы логического управления технологическим оборудованием

Многообразие аппаратных решений, используемых при построении систем логического управления, нуждается в хорошо структурированном программно-математическом обеспечении, которое позволит наиболее полно использовать вычислительные ресурсы аппаратной платформы и предоставит конечному пользователю инструментарий для реализации алгоритмов работы объекта управления. В состав программно-математического обеспечения системы логического управления входит подсистема программирования и исполнительное ядро.

Исполнительное ядро функционирует на аппаратной платформе в режиме реального времени. Создание и отладка программ логического управления реализуются на прикладном уровне через автономную подсистему программирования. К подсистеме программирования предъявляются следующие требования: независимость от применяемого аппаратного обеспечения, наличие механизма открытости для конечных пользователей, универсальность и простота работы, использование одного или нескольких языков программирования стандарта МЭК 61131-3, возможность отладки программы логического управления без подключения к реальному технологическому оборудованию.

При проектировании и реализации технически сложных, распределенных программно-аппаратных систем требуется выполнить анализ и согласование применяемых промышленных стандартов и нормативных документов различного уровня, с выделением рекомендаций и требований по их использованию. Выбранные в ходе анализа стандарты должны быть адаптированы по отношению к возможности их применения для конкретных проектов, модулей или процессов проектируемой системы управления. Для решения указанных задач используют инструментарий, получивший название профиля открытости программной или информационной системы. [95]

3.1. Разработка профиля открытости системы логического управления технологическим оборудованием

Профиль открытости - это выбор ряда базовых стандартов с ориентацией на их гармонизированные подмножества, предназначенные для проектирования конкретных модулей, функций или подмножества функций программной системы в рамках конкретной функциональной среды. Функциональная модель, процесс разработки которой представлен в главе 2.3, является

основой для формирования и дальнейшего использования профиля открытости. При проектировании целостной системы управления и выборе программных, аппаратных средств и средств коммуникации различных производителей, которые будут встраиваться в проектируемую систему, необходимо учитывать их соответствие профилю открытости, т.е. отследить их исполнение в рамках выбранных гармонизированных стандартов. В этом случае все составные части системы управления будут функционировать в единой среде, а также будут обеспечены следующие свойства системы: масштабируемость, функциональная расширяемость и переносимость приложений.

Профиль открытости системы управления состоит из набора стандартов, которые должны быть согласованы между собой и охватывать взаимодействие программных и аппаратных составляющих системы. Помимо этого, профиль открытости должен определять спецификацию протоколов и интерфейсов взаимодействия, которые составляют структуру системы управления. При построении профиля открытости необходимо использовать многоуровневую модель системы управления по типу виртуальной машины, на основе которой необходимо определить правила взаимодействия между уровнями. На каждом из уровней выделяются типы функциональных компонент в соответствии с моделью открытых систем, которые взаимодействуют между собой. Среди основных типов функциональных компонент выделяются следующие: внешние и внутренние интерфейсы системы (в том числе интерфейсы взаимодействия с пользователями), языки и стандарты программирования, управление задачами, управление данными, физические протоколы коммуникации, безопасность систем и данных и др.

Ограничения, задаваемые базовыми стандартами и нормативными документами профиля, и проведенная на этапе проектирования профиля гармонизация, должны обеспечить соответствующее качество, совместимость и корректную работу компонентов в рамках единой системы управления. При проектировании системы управления выбранные стандарты применяются как нормативная база.

В качестве целей разработки и применения профилей открытости можно выделить следующие:

- повышение связности и снижение затрат на разработку систем управления;
- обеспечение переносимости прикладных компонент систем управления;
- проектирование возможностей для расширяемости функционала и масштабируемости отдельных компонент и систем управления в целом;
- проектирование возможностей функциональной интеграции в систему управления задач, которые ранее решались обособленно, с повышением эффективности решения этих задач;
- повышение качества исполнения и надежности отдельных модулей систем управления.

Выбор стандартов и нормативных документов при формировании профиля открытости осуществляется в зависимости от приоритета среди перечисленных целей. Методологией построения профилей открытости систем управления может служить 1-ая и 2-ая части ГОСТ Р ИСО/МЭК ТО 10000-1-99, определяющие основы и таксономию профилей открытости информационных систем [96].

Международными комиссиями по стандартизации и сертификации допускается использование в качестве документов, входящих в состав профиля открытости, только стандартов международного, регионального и национального уровня и не допускается применение стандартов де-факто и технических документов отдельных фирм. Указанный подход не предполагает возможностей унификации и параметризированного использования функций и модулей современных систем управления. В этой связи большинство разработчиков систем управления при построении профилей применяют помимо базовых международных и национальных стандартов, также открытые широко используемые спецификации стандартов де-факто и рекомендательные документы международных консорциумов.

3.1.1 Определение стандартов и программных технологий, применяемых при проектировании систем логического управления

Для построения профиля открытости системы логического управления были проанализированы международные и региональные стандарты, а также стандартные технологии, применяемые при разработке программного обеспечения (таблица 3-1).

Таблица 3-1 Стандарты используемые в системах логического управления

Наименование стандарта	Область применения
ГОСТ IEC 61131-1-2016. Контроллеры программируемые. Общая информация [97].	Стандарт определяет и идентифицирует характеристики, на основе которых производится выбор ПЛК и периферийного оборудования, а также определяется область применения выбранных экземпляров. Используется на аппаратном уровне.
ГОСТ IEC 61131-2-2012. Контроллеры программируемые. Требования к оборудованию и испытания [98].	Стандарт устанавливает требования к оборудованию, тестовым испытаниям ПЛК и периферийных устройств. Используется на аппаратном уровне.
ГОСТ IEC 61131-3-2016. Программируемые контроллеры. Языки и средства программирования [99].	«Стандарт определяет базовые наборы элементов программирования, применимых тестов и средств, с помощью которых изготовители могут расширять или адаптировать данные базовые наборы в собственном внедрении ПЛК для каждого из наиболее широко используемых языков программирования, основных сфер применения, синтаксических и семантических правил.» Используется на прикладном уровне.

Продолжение таблицы 3-1.

Наименование стандарта	Область применения
ГОСТ IEC 61131-4-2004. Программируемые контроллеры. Руководства для пользователя [100].	Стандарт определяет требования к руководству по применению, которое поставляется в комплекте совместно с ПЛК. Используется на прикладном уровне.
ГОСТ IEC 61131-5-2000. Программируемые контроллеры. Спецификация сообщений [101].	Стандарт определяет характеристики обмена данными между ПЛК и другими электронными системами. Используется на аппаратном уровне.
ГОСТ IEC 61131-6-2015. Программируемые контроллеры. Промышленные сети [92].	Стандарт определяет требования для ПЛК и связанных с ними периферийных устройств, которые предназначены для использования в качестве логической подсистемы, связанной с безопасностью электрической/ электронной/ программируемой электронной. Используется на аппаратном уровне.
ГОСТ IEC 61131-7-2000. Программируемые контроллеры. Программирование с нечеткой логикой [102].	Стандарт определяет языки программирования для нечеткого контроля. Используется на прикладном уровне.
ГОСТ IEC 61131-8-2003. Программируемые контроллеры. Руководящие принципы применения и реализации языков ПЛК [103].	Стандарт определяет руководства по использованию и внедрению (имплементации) языков программирования. Используется на прикладном уровне.
ГОСТ Р 51840-2001 Программируемые контроллеры. Общие положения и функциональные характеристики. [104]	Стандарт определяет минимальные требования к функциональным характеристикам, условиям обслуживания, параметрам конструкции, общей безопасности и испытаниям ПЛК и связанных с ними периферийных устройств. Используется на аппаратном уровне.
ГОСТ 30607-98 Контроллеры программируемые станочные. Общие технические требования. [105]	Стандарт определяет общие требования к ПЛК, предназначенных для управления машинами и механизмами станочного типа, имеющими электромеханический или электроуправляемый привод для сообщения движения узлам машины. Используется на аппаратном уровне.
ГОСТ 26.013-81. Средства измерения и автоматизации. Сигналы электрические с дискретным изменением параметров входные и выходные. [106]	Стандарт распространяется на средства измерения и автоматизации и устанавливает параметры электрических входных и выходных сигналов тока и напряжения с дискретно изменяющимися амплитудой, длительностью, фазой и частотой, предназначенных для информационной связи. Используется на аппаратном уровне.
XML (eXtensible Markup Language) [107]	Расширяемый язык разметки — общий свод синтаксических правил языка разметки, который рекомендован всемирным консорциумом W3C (World Wide Web Consortium). XML — формат данных текстового вида, предназначенный для хранения данных в структурированной форме. Данные размеченные с использованием этого языка могут служить основой для организации обмена информацией между приложениями, а также для проектирования на его основе специализированных языков разметки. Используется на прикладном уровне.

Продолжение таблицы 3-1.

Наименование стандарта	Область применения
API (Application Programming Interface)	Интерфейс прикладного программирования — набор готовых программных интерфейсов, которые применяются при программной реализации пользовательских приложений и гарантируют верное взаимодействие между прикладной и системной компонентами программного обеспечения. Используется на прикладном уровне.
MFC (Microsoft Foundation Classes)	Библиотека программных классов на языке C++, разработчиком которой выступила корпорация Microsoft. Библиотека MFC служит для облегчения разработки пользовательских приложений для операционной системы Microsoft Windows. Используется на прикладном уровне.
SQL (Structured Query Language) [108]	Язык структурированных запросов — универсальный язык формирования запросов к банкам и базам данных. Применяется для работы со структурированными данными в реляционных базах данных. Используется на прикладном уровне.
COM (Component Object Model) [109]	Стандарт компании Microsoft. Позволяет создавать программное обеспечение на основе соединенных между собой распределённых компонентов, каждый из которых может быть применен в нескольких программных приложениях одновременно. Используется на прикладном уровне.
DCOM (Distributed COM)	Технологический стандарт, позволяющий COM-компонентам взаимодействовать друг с другом по сети. Используется на прикладном уровне.
OPC	Семейство стандартов программных технологий, которые позволяют посредством единого интерфейса осуществлять управления объектами автоматизации и технологическими процессами. Используется на прикладном уровне.
.Net Framework	Программная технология компании Microsoft, предназначенная для создания пользовательских и веб приложений. Позволяет совместить различные модули и службы системы, написанные на разных языках программирования. Используется на уровне программной платформы.
API операционной системы	Под интерфейсами прикладного программирования операционных систем понимают набора программных средств взаимодействия пользовательских приложений с функциями операционной системы. Используется на уровне интерфейсов прикладного программирования.
POSIX (Portable Operating System Interface) [110]	Переносимый интерфейс операционных систем — группа нормативных правил и стандартов, которые описывают интерфейсы между взаимодействия операционных систем и прикладных приложений. Стандарт позволяет обеспечить совместимость операционных систем подобных UNIX на уровне исходного кода. Возможно применение стандарта и для других типов операционных систем. Нормативные документы POSIX были разработаны комитетом 1003 IEEE Международной организации по стандартизации (ISO) и одобрены Международной электротехнической комиссией (IEC). Используется на системном уровне.

Продолжение таблицы 3-1.

Наименование стандарта	Область применения
ISO 9001:2008 Системы менеджмента качества. Требования. [111]	Стандарт описывает комплекс требований к организации, для обеспечения гарантии соответствующего качества выпускаемой продукции.
EIA232 (RS232) [112]	Стандарт связи между оборудованием, выступающим в качестве терминального клиента и связным с ним по физическим линиям связи оборудованием. При этом обменом осуществляется по последовательному двоичному коду. Используется на аппаратном уровне.
EIA485 (RS485) [113]	Стандарт связи между оборудованием на основе двухпроводного полудуплексного многоточечного последовательного канала связи. Используется на аппаратном уровне.
TCP/IP	Семейство протоколов, образующих глобальную транспортную среду для связи ЭВМ в рамках локальных и глобальных сетей. Используется на аппаратном уровне.
Fieldbus	Коммуникационная технология построения единой информационной сети, включающая физический канал соединения устройств (например, RS485) и программный протокол их взаимодействия. Используется на аппаратном уровне.
Ethernet (IEEE 802.3)	Стандарт определяет на канальном уровне модели OSI физические линии соединения и электрические сигналы в них, а также формат передаваемых кадров и протоколы управления доступом к среде. Используется на аппаратном уровне.
EtherCAT	Стандарт промышленной сети, относимый к семейству Industrial Ethernet и технологиям, используемым для распределенного управления в режиме реального времени. EtherCAT разработан компанией Beckhoff. Целью разработки протокола было использование технологии Ethernet для автоматизации приложений, которые требуют частого обновления с низкими затратами вычислительных ресурсов. Дейтаграммы EtherCAT пропускаются внутри стандартного фрейма Ethernet. Используется на аппаратном уровне.
SERCOS (SErial Real-time COmmunication System)	Интерфейс взаимодействия между контроллером и преобразователями в системах автоматизированного управления движением и СЛУ. Механизм TDMA (Time Division Multiplex Access) позволяет обеспечить работу в режиме реального времени. Последняя версия интерфейса получившая наименование SERCOS III базируется на стандарте физического уровня Ethernet. Используется на аппаратном уровне.
CAN (Controller Area Network)	Стандарт промышленной сети, ориентированный на объединение в единую сеть различных исполнительных устройств и датчиков. Режим передачи: последовательный, широковещательный, пакетный. Используется на аппаратном уровне.

3.1.2 Представление о системе логического управления технологическим оборудованием, как об открытой системе

Понятие открытости на сегодняшний момент не имеет четкого определения, поэтому применительно к системам логического управления определим его следующим образом: модульную систему можно считать открытой, если её архитектура допускает замену программных или аппаратных модулей на аналогичные сторонние модули, а для интеграции с другими системами (в том числе с пользовательскими) предусмотрен специализированный механизм. При этом необходимо понимать, что в любой системе управления могут быть подвергнуты замене лишь ограниченный ряд определенных модулей. Открытость целесообразно рассматривать применительно к конкретным уровням иерархии системы логического управления или ее составных частей (таблица 3-2).

Таблица 3-2 Профиль открытости системы логического управления технологическим оборудованием

Уровень открытости	Применяемые нормативные документы
Прикладной уровень	ГОСТ IEC 61131-3, ГОСТ IEC 61131-4, ГОСТ IEC 61131-7, ГОСТ IEC 61131-8, XML, API, MFC, SQL, COM, DCOM, OPC
Уровень программной платформы	.Net Framework, Специализированные классы поддержки кроссплатформенности
Уровень интерфейсов прикладного программирования	Win API, API ОС РВ
Системный уровень	POSIX
Аппаратный уровень	SERCOS, EtherCAT, CAN, Ethernet, Modbus,Fieldbus, TCP/IP, RS232, RS484, ГОСТ 26.013, ISO 9001:2008, ГОСТ IEC 61131-1, ГОСТ IEC 61131-2, ГОСТ IEC 61131-5, ГОСТ IEC 61131-6 , ГОСТ Р 51840-2001, ГОСТ 30607-98

Исходя из построенного профиля можно сделать выводы о том, что реализуемая система будет легко интегрируемой с большинством средств автоматизации за счет стандартизованных интерфейсов взаимодействия. Помимо этого, важной целью построения открытой системы управления является цель переноса открытости на уровень конечного пользователя, которая определяется следующими факторами: возможность программной реализации алгоритмов управления, возможность выбора аппаратной платформы в зависимости от типа решаемых за-

дач, возможность выбора операционной системы, возможность настройки системы с применением файлов конфигурации и добавления пользовательских сервисов. На основе профиля открытости были определены уровни открытости системы логического управления и выделены варианты компоновки (рисунок 3.1).

Выделение уровней открытости позволяет на этапе проектирования заложить в систему свойство многовариантности компоновки как программных, так и аппаратных компонент. Открытость на программном уровне позволяет конфигурировать систему исходя из условий работы и добавлять приложения сторонних производителей (в том числе и приложения конечных пользователей). Открытость на уровне операционной системы позволяет реализовать кроссплатформенное решение, которое может работать в рамках поддерживаемых операционных систем. Открытость на уровне ввода/вывода данных позволяет принимать, обрабатывать, анализировать и передавать на объект управления различные типы физических сигналов. Открытость на аппаратном уровне позволяет использовать в качестве базовой вычислительной платформы и аппаратных модулей ввода/вывода широкий круг устройств, удовлетворяющих предъявляемым требованиям по их подключению к системе управления. Открытость на уровне среды программирования позволяет: использовать в качестве языка реализации программных алгоритмов один из стандартных языков программирования электроавтматики и иметь реализованные библиотеки программ в качестве подключаемых к проектам ресурсов.

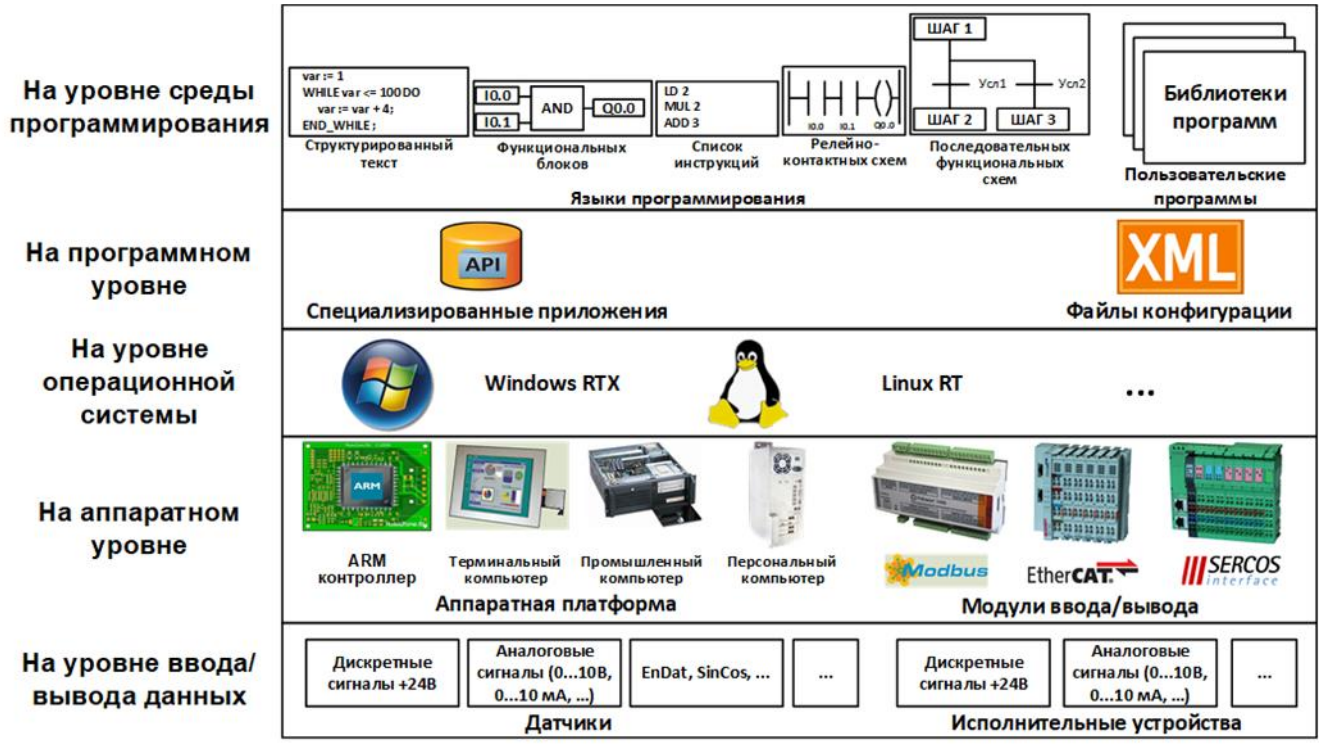


Рисунок 3.1 Многовариантность компоновки системы логического управления

3.2. Реализация подсистемы программирования для систем логического управления технологическим оборудованием

Основной функцией подсистемы программирования системы логического управления является организация интерфейса оператора, в состав которого входят среда проектирования и реализации программ логического управления, выполненная по требованиям стандарта МЭК 61131-3 и утилита конфигурации оборудования.

В большинстве систем логического управления в качестве устройства для программирования контроллеров используются ноутбук или персональный компьютер с установленным специальным программным обеспечением, выполняющим трансляцию языка в исполняемый код процессора, который загружается в память ПЛК [114-115]. Для поддержки нового контроллера применяемый на сегодня подход предполагает дополнение программного кода среды программирования модулем, осуществляющим взаимосвязь с конкретным применяемым микропроцессором. Т.е. для каждого впервые подключаемого контроллера необходимо разрабатывать дополнительный программный код для среды программирования. В предлагаемой в диссертационной работе системе логического управления привязка осуществляется к протоколу взаимодействия между исполнительным устройством и аппаратными входами/выходами. Это позволяет отказаться от программной реализации трансляторов под конкретную вычислительную платформу, при этом необходимо реализовать ряд стандартных высокоскоростных протоколов связи (как, SERCOS и EtherCAT) и на их основе осуществлять связь между аппаратурой системы.

Задачу организации среды проектирования и реализации программ логического управления можно решить двумя способами: использовать готовый программный продукт известного производителя или создать решение, ориентированное на собственные потребности. Из проведенного в разделе 1.4 анализа можно сделать вывод, что ряд недостатков не позволяют применять существующие программные среды в кроссплатформенных проектах [116-117]. В первую очередь это ориентация на конкретную операционную систему (чаще всего Windows), необходимость приобретения лицензии на коммерческое использование; закрытость системы, ориентированная на комплексное решение, но без возможности его модернизации и доработки. К тому же ни один из указанных программных пакетов не поставляется с «открытым» программным кодом, что делает затруднительным их использование в качестве базы для реализации системы логического управления.

Для ориентации на кроссплатформенные решения необходима разработка среды проектирования и реализации логических программ стандарта МЭК 61131-3, удовлетворяющей сле-

дующим требованиям: аппаратная и платформенная независимость, отсутствие платной лицензии, открытость механизма для конечного пользователя (в частности для станкостроителей), простота работы, стандартизированный язык программирования, наличие режима эмуляции, возможность расширения функционала и модернизации за счёт открытости кода (т.к. продукт является собственной разработкой), а также возможность использования в учебных целях [115 - 117].

3.2.1 Формирование требований к построению подсистемы программирования стандарта МЭК 61131-3 для систем логического управления

Среда проектирования, реализации, визуализации и отладки программ логического управления должна являться простым и универсальным инструментарием проектирования и разработки программного обеспечения для систем логического управления, адаптированным под работу с аппаратной частью различных производителей.

При разработке инструментария за прототип была взята среда CoDeSys, как один из наиболее известных универсальных и хорошо себя зарекомендовавших инструментов программирования контроллеров и промышленных компьютеров. Поддержка большого количества языков программирования на первом этапе разработки инструментария является экономически нецелесообразной. В качестве базового языка разработки программ логического управления был выбран язык Functional Block Diagram (FBD). Выбор был обусловлен тем, что большая часть проектов по реализации систем логического управления предполагает первоначальную разработку принципиальных электрических схем подключения узлов электроавтоматики. Программа на языке FBD внешне напоминает функциональную схему логического устройства – совокупность элементов (блоков), входы и выходы которых соединены линиями связи. Связи, соединяющие выходы одних блоков с входами других, являются линиями передачи переменных и служат для передачи данных между функциональными блоками. FBD обеспечивает простой переход от принципиальной электрической схемы к программе на языке FBD.

Выбранный язык входит в состав международного стандарта МЭК 61131-3 и используется во многих прикладных средах программирования контроллеров, по нему имеется большое количество литературы и подготовленных специалистов. В качестве основного преимущества FBD можно отметить наглядность, поскольку язык является полностью графическим. Программы, написанные на графических языках, при реализации сложных проектов становятся довольно громоздкими, однако этот недостаток решается за счет вложенности и возможности

формирования пользовательских функциональных блоков. Это позволяет разделить реализацию программы логического управления на несколько уровней и отображать только редактируемые в настоящий момент уровни. Также выбранный язык предлагает возможности для отладки путем визуального отображения текущих значений всех входов, выходов и компонентов программы логического управления, находящейся в работе.

Каждый блок программы, представляющий собой некоторую функцию, может содержать произвольное количество входов и выходов. Значения переменных задаются с помощью входных блоков или констант, выходные блоки должны быть логически связаны с аппаратными выходами, либо с глобальными переменными программы. Графический редактор программ логического управления должен иметь широкие возможности для создания и редактирования программ. Диаграмма прецедентов среды разработки программ логического управления представлена на рисунке 3.2.

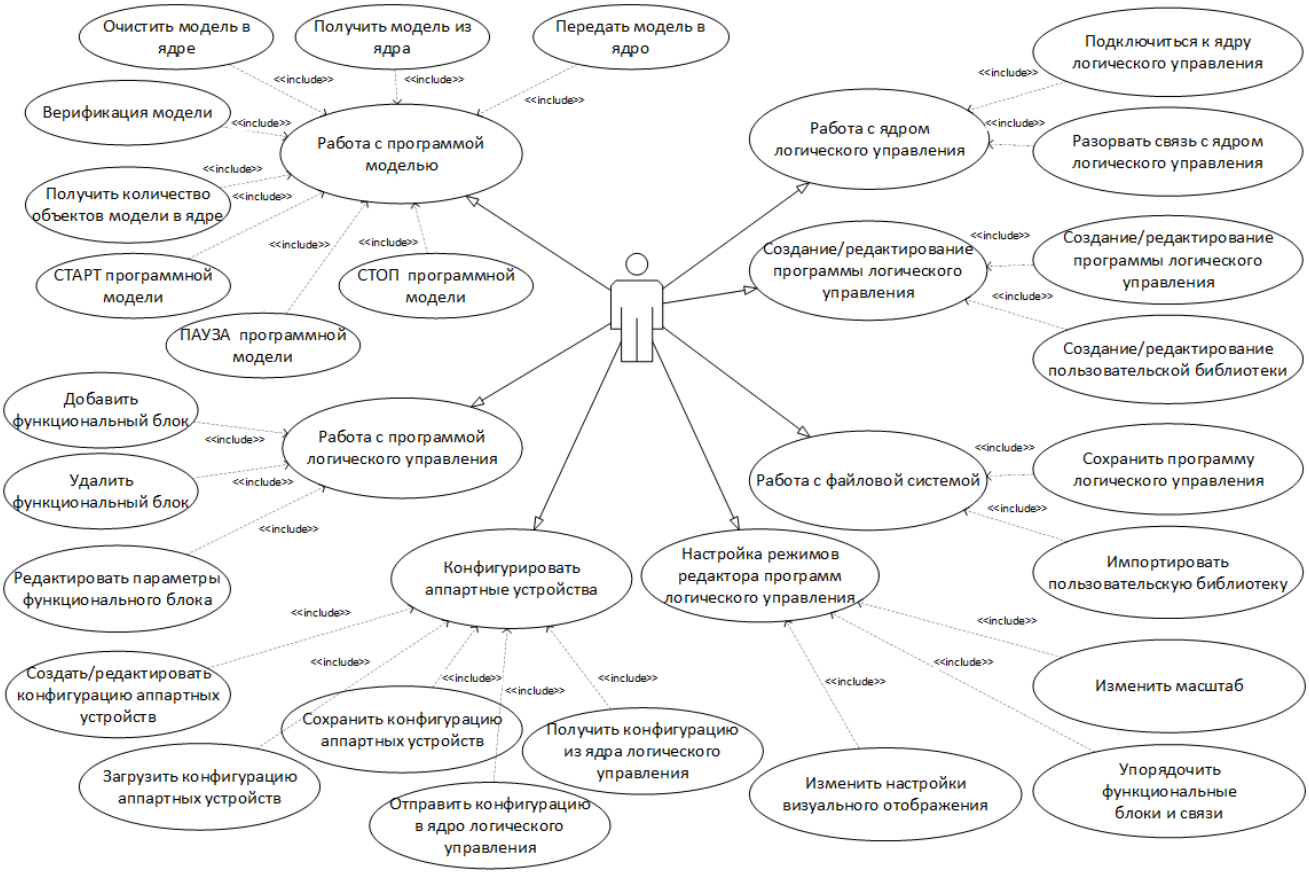


Рисунок 3.2 Построение диаграммы прецедентов инструментария разработки и отладки программ логического управления

Проблема построения специализированной среды разработки программ логического управления, интегрированной в состав терминальной части системы ЧПУ, была рассмотрена в диссертационной работе «Расширение функциональных возможностей специализированных

систем ЧПУ посредством организации многоцелевого канала взаимодействия их основных компонентов» П.А. Никишечкиным. [133] В указанной работе не уделено внимание вопросам работы среды разработки программ логического управления как отдельного приложения.

Разработка инструментария программирования и отладки программ логического управления велась с использованием технологий .NET и XML. При разработке сохранялось следование основам объектно-ориентированного проектирования, таким, как компонентная архитектура, модульность, а также использование концепции MVC (Model View Controller) – разбиения программного продукта на отдельные модули, которые отвечают за хранение, визуализацию и управление данными. Применение указанных принципов позволило добиться высокой гибкости ПО, а также независимости модулей хранения и визуализации данных. «Интерфейс среды разработки, представленный на рисунке 3.3, позволяет производить редактирование как визуальной части (создание/удаление/перемещение функциональных блоков, масштабирование, регулировка приоритета видимости, копирование объектов и др.), так и настраивать параметры функциональных блоков с панели настроек». [115-117]

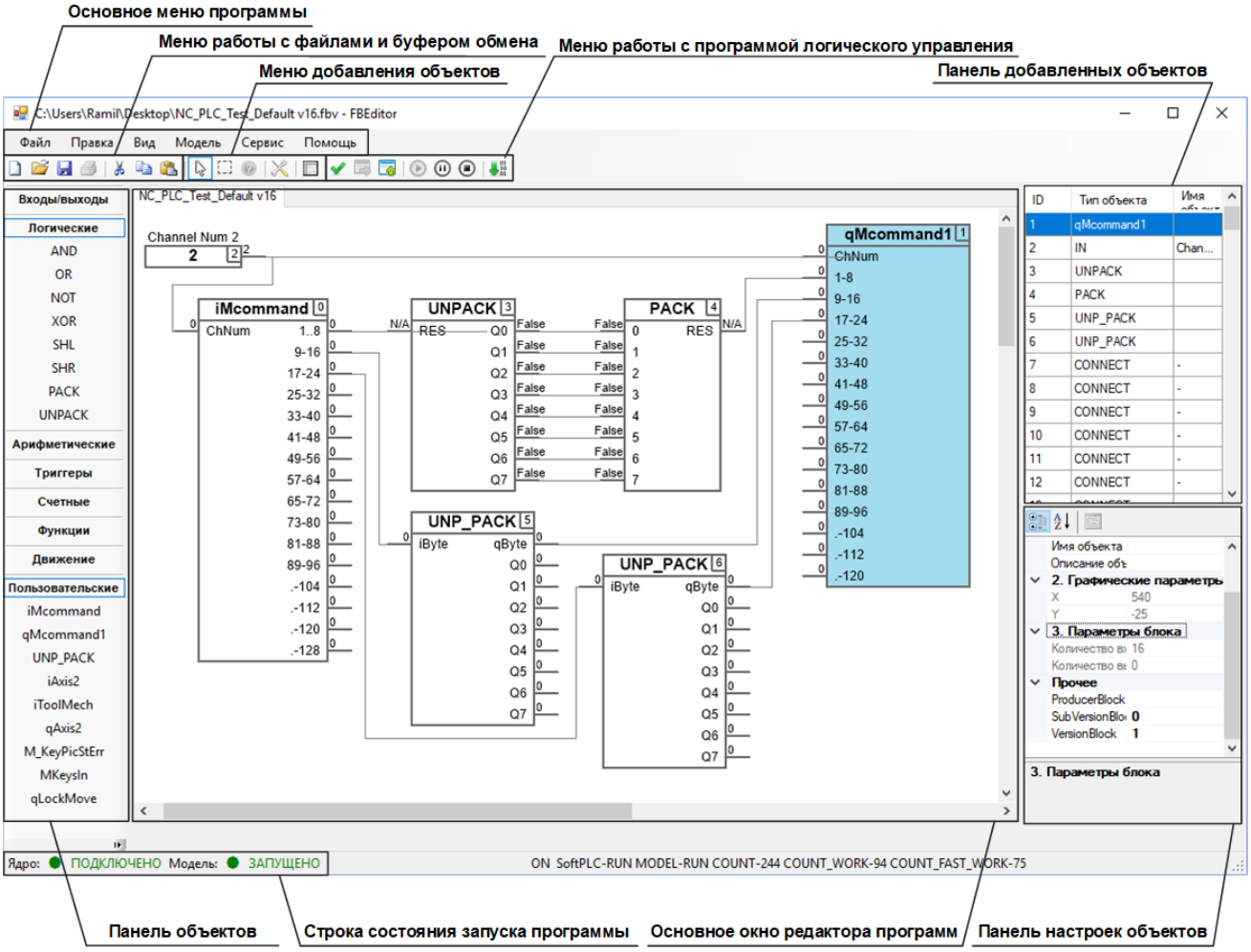


Рисунок 3.3 Интерфейс редактора программ логического управления

Одним из возможных режимов работы редактора является режим отладки, который можно активизировать при запущенной программе логического управления. В режиме отладки возможно визуально отследить нарушения в логике работы программ.

Интерфейс среды разработки программ логического управления состоит из следующих компонентов: основное меню программы, меню работы с файлами и буфером обмена, меню добавления объектов, меню работы с программой логического управления (содержит функции по отправке/получению из ядра программы логического управления, а также ее запуску и останову), панель объектов (содержит визуальные компоненты, соответствующие различным функциональным блокам, которыми можно оперировать при создании программы логического управления), панель добавленных объектов (содержит список актуальных объектов рабочей программы), строка состояния запуска программы (отображает текущее состояние подключения к ядру системы, а также статус запуска программы), основное окно редактора программы, панель настроек объектов.

С точки зрения архитектуры программного обеспечения весь функционал разработанного инструментария делится на отдельные модули. Схема взаимодействия основных модулей представлена на рисунке 3.4.

Модули выполнены независимо друг от друга и взаимодействуют между собой через специально разработанные программные интерфейсы, что позволяет следовать основным парадигмам компонентного подхода. Базовым модулем является визуализатор модели программ логического управления, на вход которого подаются команды на добавление, редактирование функциональных блоков и настройку модели. Разработанный инструментарий содержит базовый набор элементов на языке FBD, с помощью которого можно решать задачи автоматизации производственных процессов.

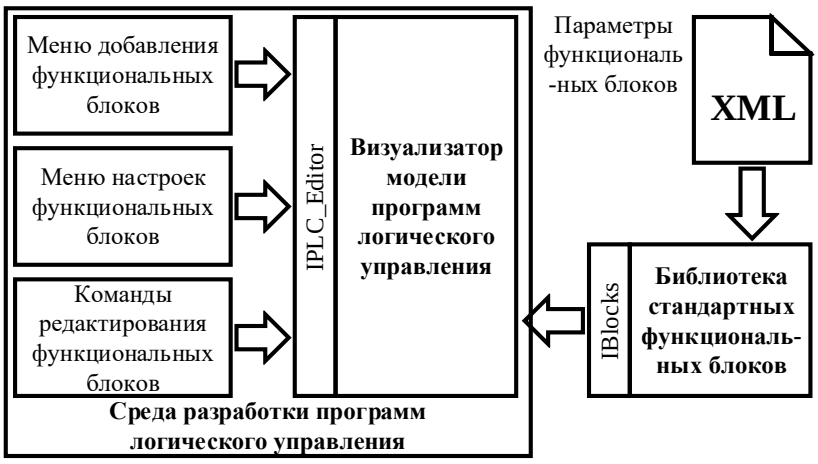


Рисунок 3.4. Основные модули редактора программ логического управления

Набор функциональных блоков хранится в библиотеке стандартных функциональных блоков, содержащей в себе информацию обо всех объектах (логические элементы, математические функции, таймеры, триггеры, счетчики, и др.), которыми можно оперировать при создании программы. Программный интерфейс библиотеки содержит методы, позволяющие получать объекты функциональных блоков по их названию или индексу. Каждый объект функционального блока состоит из графической и модельной частей. Графическая часть содержит поля и методы визуализации функционального блока, а модельная часть хранит в себе информацию, необходимую для реализации логики работы блока. Описание функциональных блоков содержится в XML-файле (FBBlocksParam.xml) [115-117].

На рисунке 3.5 представлена XSD схема, которая отображает структуру данных, содержащихся в файле «FBBlocksParam.xml». Каждый функциональный блок имеет набор обязательных (тип блока) и не обязательных (отображение типа блока, максимальная высота, максимальная ширина, отображение приоритета блока) атрибутов. Схема содержит характеристики входных и выходных контактов: обязательные (тип) и опциональные (имя, отображение имени, инверсность). Представленный набор характеристик позволяет среде программирования осуществлять визуальную отрисовку функционального блока в соответствии с его параметрами. Предложенный подход позволяет обеспечить изменение отображаемых характеристик блоков, не прибегая к перекомпиляции самой среды программирования.

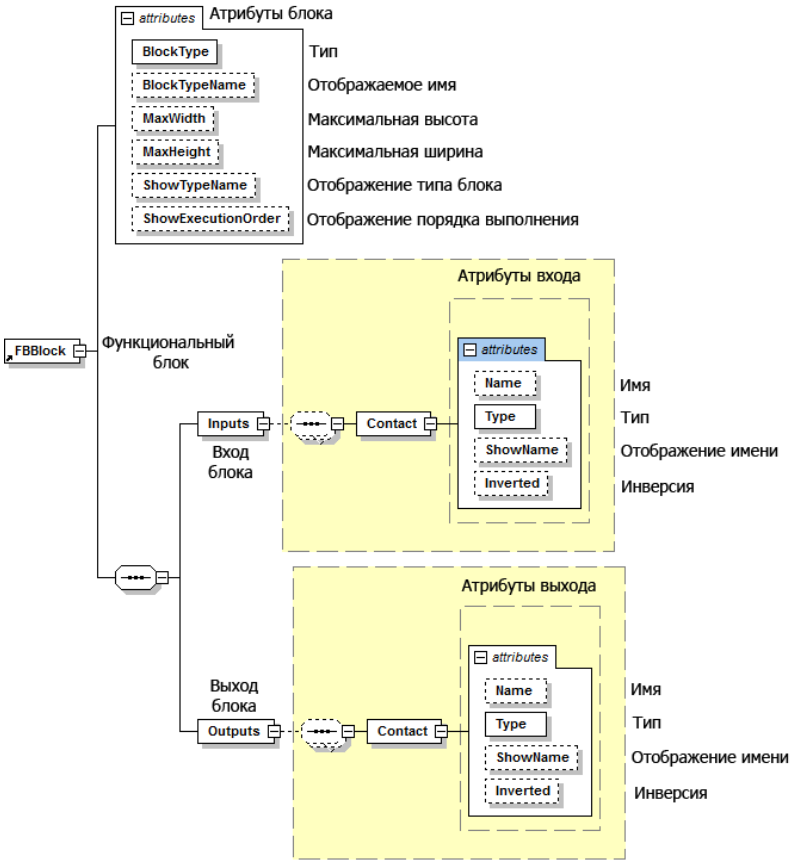


Рисунок 3.5. XSD-схема описания обобщенного функционального блока

«Работа каждого из функциональных блоков определяется набором программных параметров, настройку которых можно осуществить двумя способами: установить требуемые параметры в панели настроек блока или открыть специализированное окно настроек» [115-117]. Параметры работы с аппаратными входами/выходами задаются в блоках входов/выходов.

Реализация среды разработки программ логического управления позволяет функционировать в режиме подключения к аппаратным входам/выходам и в автономном режиме. Автономный режим позволяет вести разработку и отладку программ логического управления без использования дорогостоящего оборудования на базе персонального компьютера в режиме эмуляции, что снижает риск возникновения аварийных ситуаций на реальном объекте управления.

3.2.2 Особенности проектирования модуля конфигурирования аппаратных входов/выходов для систем логического управления

Непосредственное управление исполнительными устройствами осуществляется через модули аппаратных входов/выходом. В работе аппаратным модулем входов/выходов называется функционально законченное устройство, состоящее из головного модуля (модуля коммуникации) и одного или нескольких слотов ввода/вывода. Головной модуль обеспечивает взаимодействие с «мастером» сети (интерфейсной платой в ядре системы управления верхнего уровня) на базе одного из поддерживаемых открытых протоколов. Слоты ввода/вывода предназначены для приема входных и формирования выходных дискретных и аналоговых сигналов. Модули аппаратных входов/выходов являются пассивными устройствами, не производят самостоятельные вычисления, но поддерживают работу с одним или несколькими промышленными протоколами связи, могут соединяться между собой, образовывая сетевую структуру, в том числе многоранговую.

Отображение переменных ПЛК-проекта на аппаратные входы/выходы задается с помощью модуля конфигурации оборудования, где переменные сопоставляются каналам входов/выходов. Полная конфигурация аппаратных входов/выходов, как правило, имеет разветвленную многоуровневую структуру. Все настройки производятся во встроенном в подсистему программирования конфигураторе, без необходимости применения внешних инструментов. Модуль конфигурации оборудования позволяет: сформировать иерархию устройств в дереве проекта; настроить параметры устройств; привязать переменные к каналам ввода/вывода (рисунок 3.6).

Для привязки аппаратных модулей входов/выходов к программе логического управления на основе технического задания необходимо определить: структуру системы управления, топологию сети, протоколы коммуникации, а также производителя, модель и количество аппаратных модулей входов/выходов.

Конфигурирование и выбор параметров модулей входов/выходов для их интеграции в состав системы логического управления осуществляется с использованием механизма разделяемой памяти. Распределение адресного пространства области разделяемой памяти в ядре системы и расчет параметров конфигурирования осуществляется в индивидуальном порядке для каждой уникальной комбинации модулей. Каждый слот, в зависимости от своего типа, в области разделяемой памяти занимает определенный объем и позицию. На основе выбранных параметров состояние входов проецируется из внутренней памяти устройства в область разделяемой памяти системы управления, а в обратном направлении передаются управляющие воздействия на выходные канала периферийных модулей.

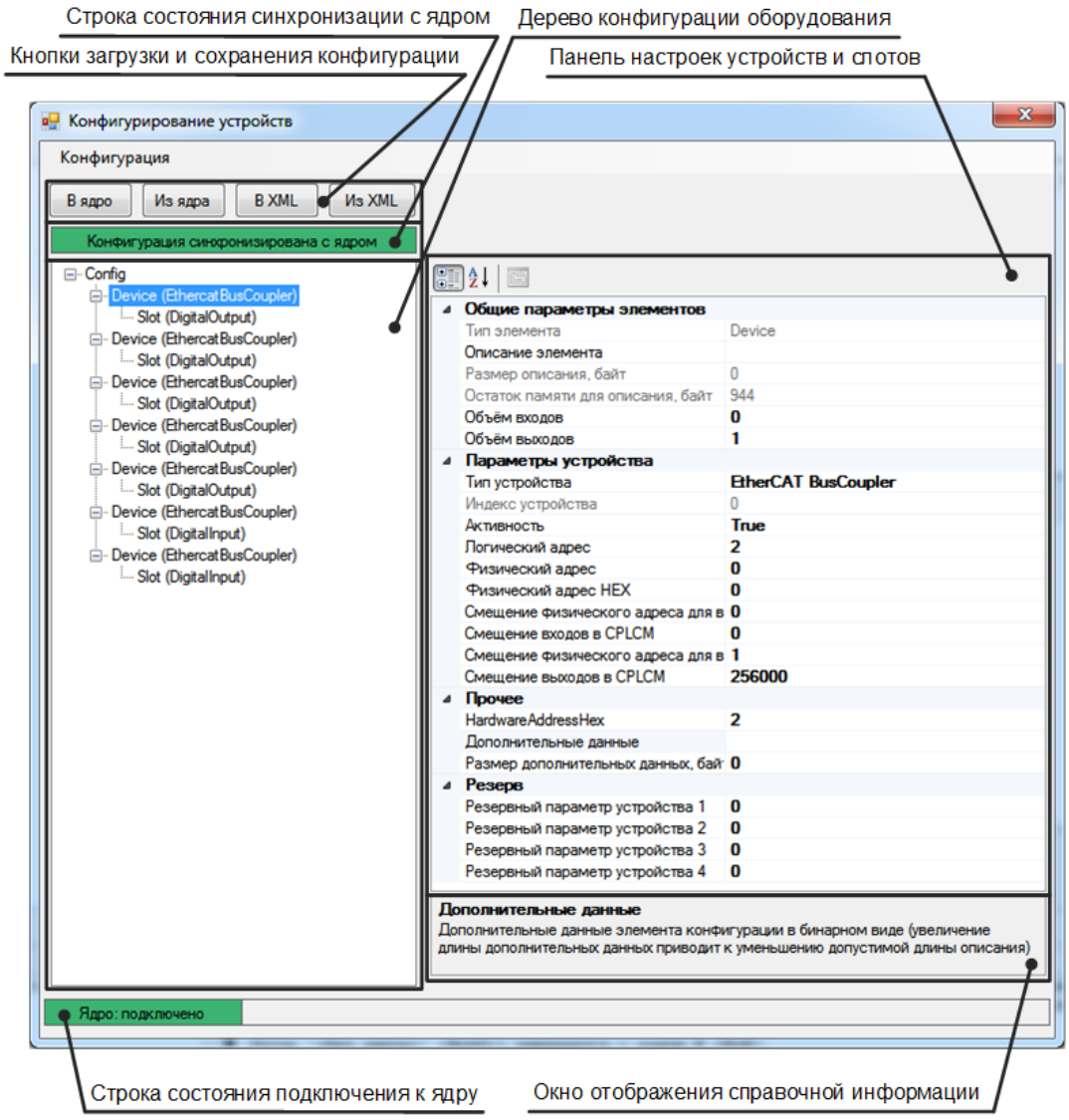


Рисунок 3.6 Модуль конфигурации программы логического управления

Сохранение параметров конфигурации аппаратных входов/выходов может осуществляться в файловой системе подсистемы программирования в файл с расширением «*.bin» или формата XML. Структура файла конфигурации, содержащая атрибуты аппаратных устройств, представлена на XSD схеме (рисунок 3.7). Каждый из узлов «Device» и «Slot» дополнительно должен содержать атрибуты, которые позволяют однозначно идентифицировать подключаемые аппаратные модули и слоты. Среди атрибутов можно выделить: протокол коммуникации, тип/назначение устройства, общий размер передаваемых данных, адрес в адресном пространстве разделяемой памяти.

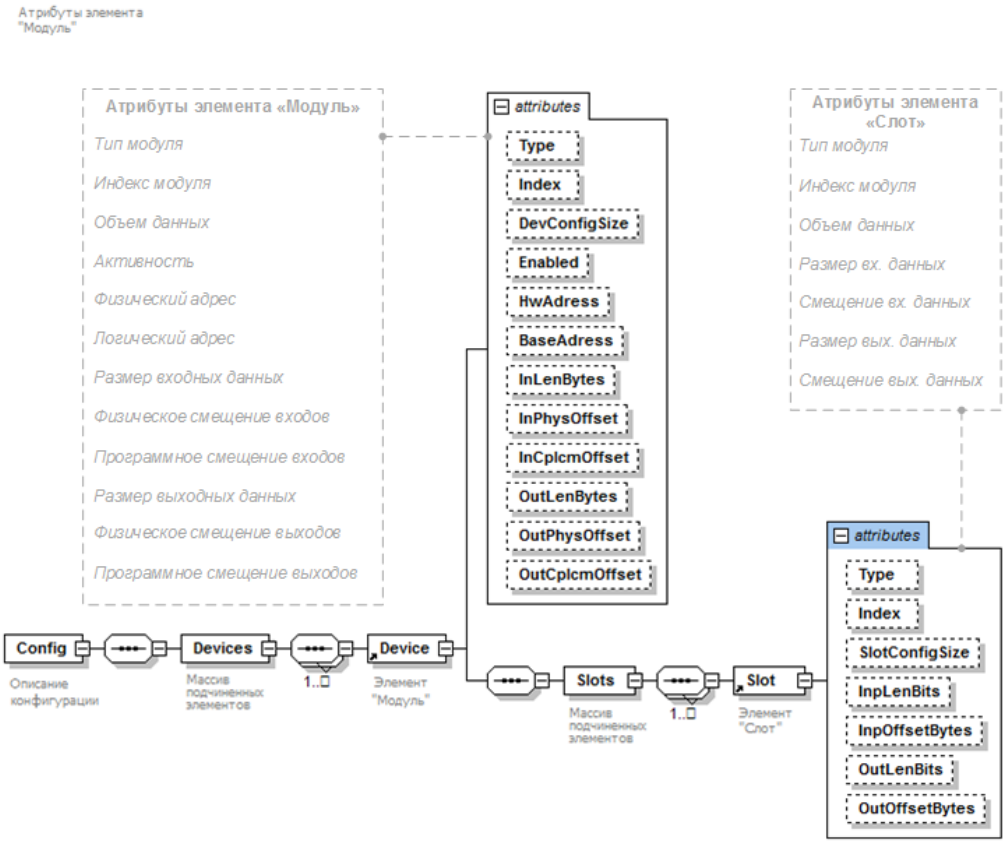


Рисунок 3.7 XSD схема файла конфигурации аппаратных входов/выходов

Готовая конфигурация упаковывается в специальную структуру и передается в исполнительное ядро системы логического управления. Инициализация адресного пространства ядра логического управления для работы с аппаратными входами/выходами производится на основе загруженной конфигурации в момент первоначального запуска ядра логического управления.

3.3. Реализация исполнительного ядра системы логического управления технологическим оборудованием как системы реального времени

Управление аппаратной частью системы логического управления производится через программное обеспечение, находящееся в исполнительном ядре, функционирующем в режиме реального времени [124-125]. Основными задачами ядра являются:

- последовательный опрос входных сигналов с аппаратных входов;
- выполнение заложенных в память алгоритмов управления;
- подготовка и формирование управляющих сигналов на аппаратные выходы;
- сохранение результатов обработки;
- передача результатов в подсистему программирования для отладки;
- контроль системы на отсутствие ошибок.

Ядро является базовым модулем системы управления и определяет ее функциональные возможности [126-127]. При реализации ядра использовался кроссплатформенный подход, что позволило адаптировать работу системы логического управления под различные ОСРВ (MS Windows RTX, ОС Windows Embedded CE, Linux) [128].

В состав исполнительного ядра системы входят следующие модули: конфигурации библиотек логических элементов, конфигурации устройств связи с подсистемой программирования, конфигурации системы, работы с разделяемой памятью (рисунок 3.8) [129-130].

Модуль работы с разделяемой памятью позволяет осуществить синхронизацию состояния аппаратных входов/выходов с состоянием ячеек разделяемой памяти ядра системы. Синхронизация данных между разделяемой памятью и аппаратными входами/выходами должна производится один раз за цикл работы ядра: для входов - до выполнения алгоритмов управления, для выходов – после.

Модуль конфигурации аппаратных входов/выходов позволяет инициализировать адресное пространство разделяемой памяти в соответствии с конфигурацией аппаратных входов/выходов, заданной пользователем и загруженной из подсистемы программирования. При этом каждый аппаратный вход/выход получает уникальный адрес отображения в разделяемой памяти, с которым работает (считывает/записывает значения) на протяжении активности системы логического управления. Посредством модуля конфигурации системы управления пользователь выбирает режимы работы ядра логического управления, среди которых можно выделить: время цикла работы ядра, время цикла синхронизации данных между разделяемой памятью и аппаратными входами/выходам, выбор протокола связи с аппаратными входами/выходами, режим верификации данных и др.

Рисунок 3.8 Модули системного ядра логического управления

Модуль конфигурации функциональных блоков необходим для определения следующих параметров подсистемы программирования: режимы хранения данных, относящихся к функциональным блокам; режимы отображения функциональных блоков в среде программирования; режимы передачи функциональных блоков между средой программирования и ядром логического управления.

Для обеспечения связи подсистемы логического управления с подсистемой программирования используется канал связи на базе протокола TCP/IP с применением на физическом уровне витой пары. Взаимодействие и обмен данными между подсистемами осуществляется с использованием специализированного канала взаимодействия.

До запуска программы логического управления в ядре необходимо провести её верификацию на предмет корректности полученных из подсистемы программирования пакетов данных и программ логического управления, выявленные при этом ошибки отправляются в подсистему программирования для отображения на экране оператора.

Модуль исполнения цикла логического управления организует работу ядра логического управления в соответствии со стандартным циклом логического управления.

Каждый из представленных модулей работает независимо от остальных, что позволяет оперативно обрабатывать предоставленную ему информацию. В тоже время, благодаря жестко выполняемому циклу работы обеспечивается строгое соблюдение актуальности данных.

3.3.1 Реализация машины состояний ядра системы логического управления

Функционал ядра системы логического управления реализован на базе автоматной парадигмы программирования, при которой система имеет конечное число состояний, переход между которыми осуществляется при выполнении ряда условий [131]. Модуль машины состояний ядра представляет собой конечный автомат с семью состояниями и переходами между ними (рисунок 3.9).

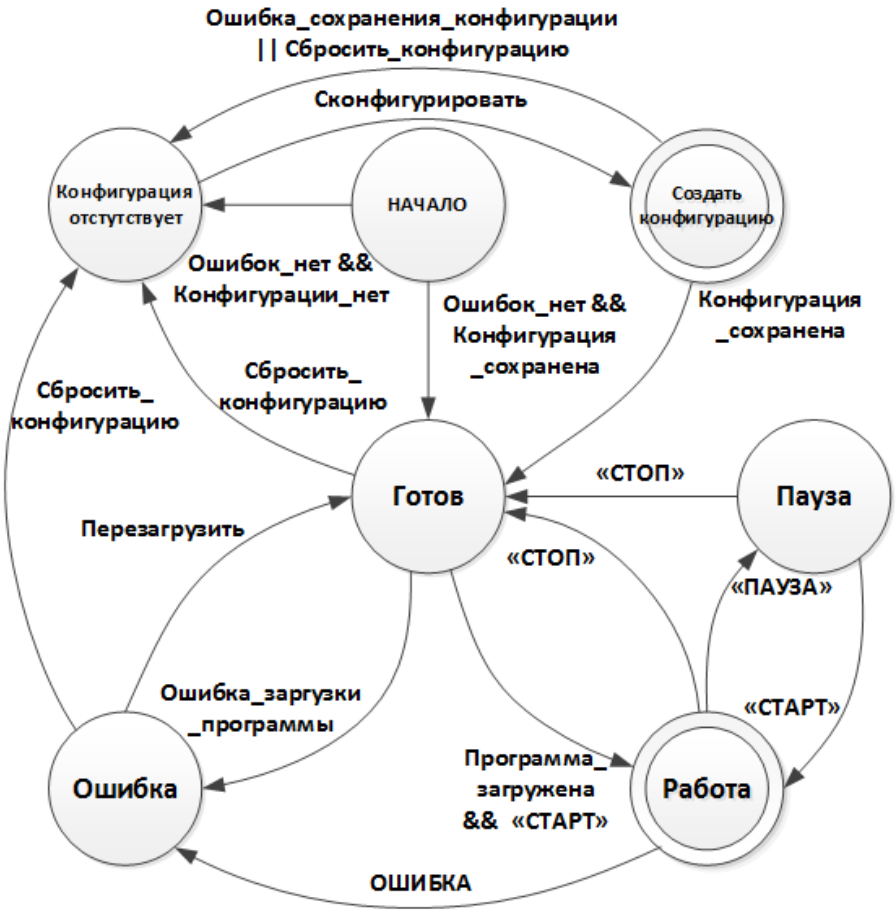


Рисунок 3.9 Машина состояний ядра системы логического управления

В момент первоначального включения ядро логического управления находится в состоянии «НАЧАЛО», при отсутствии системных ошибок осуществляется переход в состояние «Конфигурация отсутствует». Дальнейшие действия можно выполнять, только сконфигурировав ядро, т.е. перейдя в сложное состояние «Создать конфигурацию». При выключении ядра файл конфигурации сохраняется, последующее включение приведет к загрузке сохраненной конфигурации. После конфигурирования ядро переходит в режим готовности, загрузка программы логического управления и нажатие на кнопку «СТАРТ» переводит систему в режим работы. Из каждого из состояний при наличии ошибки предусмотрен переход в состояние

«Ошибка», выход из состояния «Ошибка» осуществляется посредством сброса системы. Состояние «Работа» является сложным состоянием, в котором происходит запуск цикла работы ядра с программой логического управления.

Ядро системы логического управления периодически (с периодом от 10 до 100 мс) повторяет жестко определенную последовательность действий (рисунок 3.10), включающую в себя следующие 4 фазы:

- Системного анализа. Производится тестирование контроллера на наличие ошибок и нерегулярных ситуаций.
- Чтения входов. Производится чтение актуальных данных, хранимых в разделяемой памяти контроллера, которые являются отображением состояния аппаратных входов системы.
- Выполнение программы логического управления. Производится выполнение алгоритма логического управления, программно заданного пользователем.
- Запись выходов. Производится запись актуальных результатов выполнения программы логического управления в разделяемую память контроллера. Область разделяемой памяти, в которой хранятся результаты выполнения программы логического управления, периодически синхронизируется с аппаратными выходами контроллера.

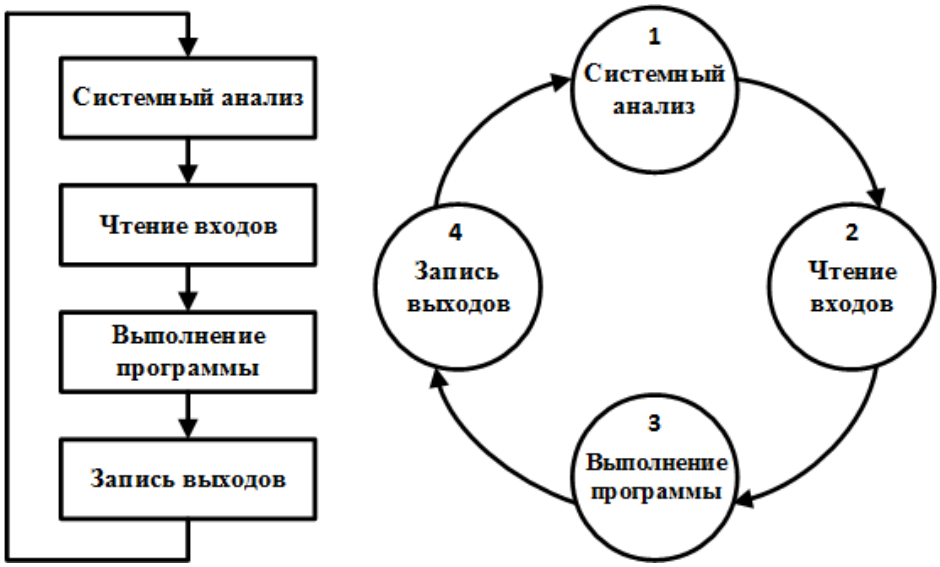


Рисунок 3.10 Цикл работы и машина состояний логического управления

Указанная последовательность действий называется циклом работы ПЛК и была сформирована в представленном виде с появлением первых логических контроллеров. Реализация машины состояний на языке C++ осуществлена в виде перечисления (enum). Каждая из позиций перечисления соответствует одному из состояний модели машины состояний (таблица 3-3).

Таблица 3-3 Соответствие состояний модели и машины состояний ядра

Отдельные состояния модели машины состояний	Отдельные состояния программной реализации машины состояний	Описание состояния
Начало	`SOFT_PLC_UNK_C = 0,`	Начальное состояние после загрузки ядра
Конфигурация отсутствует	`SOFT_PLC_NO_CFG_C = 1,`	Конфигурация в ядре не сохранена
Создать конфигурацию	`SOFT_PLC_CREATE_CFG_C = 2,`	Состояние создания конфигурации
Готов	`SOFT_PLC_READY_C = 3,`	Состояние готовности
Работа	`SOFT_PLC_RUN_C = 4,`	Программа логического управления запущена
Пауза	`SOFT_PLC_PAUSE_C = 5,`	Программа логического управления временно приостановлена
Ошибка	`SOFT_PLC_ERROR_C = 13`	Возникла ошибка при работе ядра

Состояния SOFT_PLC_UNK_C = 0, SOFT_PLC_NO_CFG_C = 1, SOFT_PLC_ERROR_C = 13 не позволяют перейти к работе, для перехода в рабочее состояние необходимо устранить ошибки или загрузить конфигурацию.

3.3.2 Особенности реализации исполнительного ядра системы логического управления

В разделе 2.8 была рассмотрена модель трансформации программы логического управления. В ней подробно описаны манипуляции, проводимые с программой на уровне подсистемы программирования и подсистемы логического управления, однако не рассмотрены механизмы, которые позволяют производить указанные манипуляции. В этой связи рассмотрим подробно этап формирования программной модели на уровне подсистемы логического управления и принципы, лежащие в основе выполнения цикла работы программы логического управления. Для выполнения алгоритмов в подсистему логического управления передается конфигурация и программа логического управления (рисунок 3.11).

Передачу программы инициирует пользователь через подсистему программирования, которая отправляет по каналу взаимодействия команду на формирование списка элементов, при этом ядро переходит в состояние `SOFT_PLC_CREATE_CFG_C = 2.` Программа логического управления в виде модели, включающей в себя функциональные блоки и связи между ними, помещенные в список типа `CComList <CNcSoftPlcBaseObject>,` который совместно со

служебной информацией передается по каналу взаимодействия от подсистемы программирования к подсистеме логического управления (рисунок 3.11). При этом в ядре активируется функция `ReadAndAddObjectFromBuffer(pInData, inSize)`, которая осуществляет: разбор пакетов данных, полученных от среды программирования, и формирование объектов ядра логического управления. Для передачи программной модели от подсистемы программирования к исполнительному ядру системы логического управления необходимо выполнение определённой последовательности действий, которая отображена на диаграмме последовательности (англ. sequence diagram), представленной на рисунке 3.12.

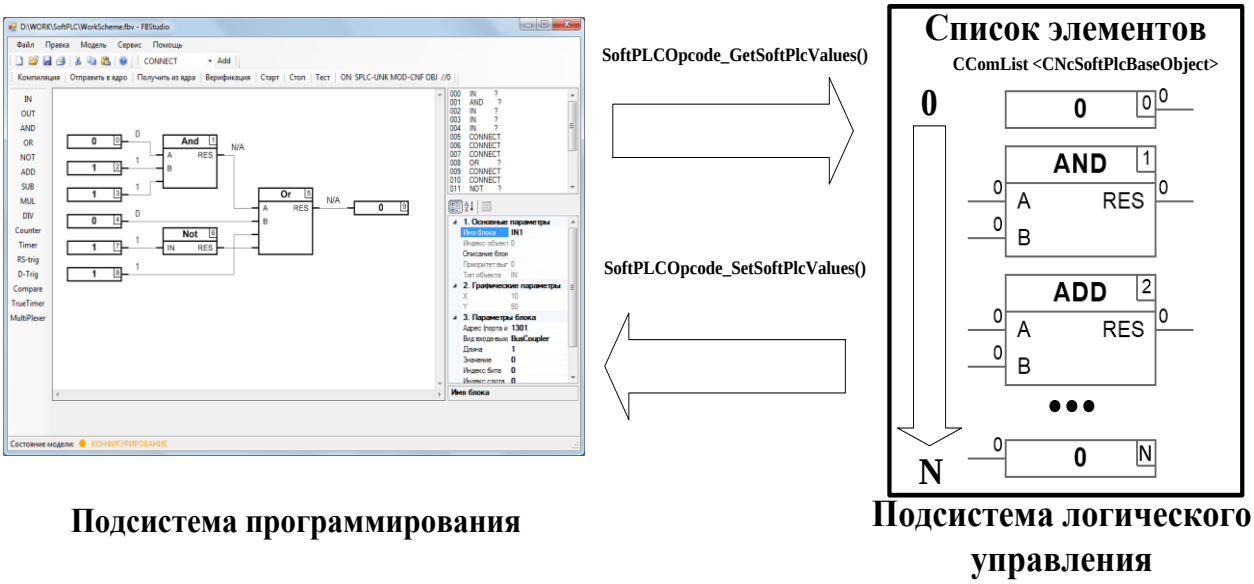


Рисунок 3.11 Механизм передачи программы логического управления от среды программирования к ядру логического управления

В процессе передачи программной модели в ядро логического управления участвуют: среда разработки программ логического управления, клиент коммуникации с ядром, модуль приема/передачи в исполнительном ядре, модуль канала взаимодействия и модуль реализации цикла логического управления. До передачи программной модели среда разработки программ логического управления формирует запрос состояния программной модели, находящейся в ядре (`GetModelState()`), по результатам которого принимается решение о возможности передачи данных. Если передача программной модели возможна, то первоначально передается её идентификатор (`SetModelId()`).

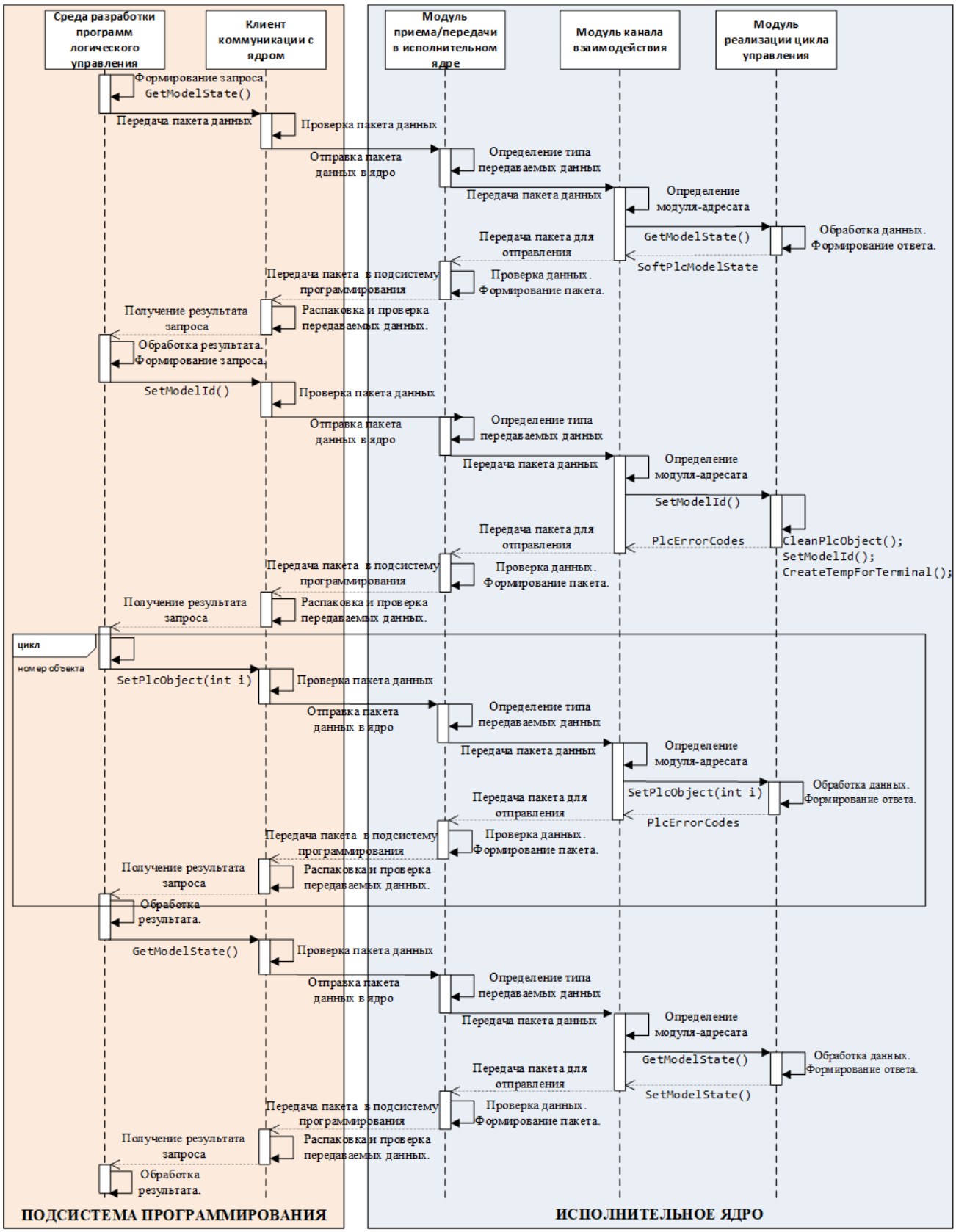


Рисунок 3.12 Диаграмма последовательности передачи программной модели в ядро

При получении идентификатора модуль реализации цикла логического управления последовательно производит: очистку области памяти, выделенной под программную модель

(CleanPlcObject()), установку идентификатора программной модели в исполнительном ядре (SetModelId()) и выделение временной области памяти под формирование программной модели в исполнительном ядре (CreateTempForTerminal()). После успешности установки идентификатора программной модели в исполнительное ядро, в цикле по одному пересылаются объекты программной модели (SendPlcObject()), количество итераций цикла равно количеству объектов программной модели. При получении данных о каждом из объектов в модуле реализации цикла логического управления вызывается функция, формирующая в ядре объект программной модели (CreatePlcObject()).

После формирования программной модели в исполнительном ядре и до её запуска в работу производятся подготовительные операции, такие как:

- верификация, с проверкой на наличие ошибок в модели;
- строго однозначное определение взаимосвязей между элементами типа Connect (программная связь) и FunctionalBlock (функциональный блок). Эта операция позволяет создать канал передачи сигналов от одних функциональных блоков к другим;
- анализ функциональных блоков со множественной вложенностью на наличие синтаксических ошибок.

После формирования списка элементов в объекте класса `class NC_API CSoftPlcObjectList : public CComList<CNcSoftPlcBaseObject*>` и их верификации ядро логического управления переходит в состояние SOFT_PLC_READY_C и ожидает поступления команды запуска модели в работу.

Команда запуска модели в работу поступает из подсистемы программирования после активации пользователем команды «ПУСК». Это приводит к переходу ядра логического управления в состояние SOFT_PLC_RUN_C с активацией рабочего режима, в котором происходит запуск цикла, описанного на рисунке 3.10.

Для выполнения алгоритма логического управления необходимо выполнить алгоритмы каждого из функциональных блоков. Для этого классы, на основе которых формируются объекты функциональных блоков, наследуются от единого базового класса CNcSoftPlcBaseObject и переопределяют его виртуальную функцию `virtual void OnTick(uint64_t timer_counter, uint32_t period_in_mks)`, которая осуществляет расчет алгоритма заложенного в функциональный блок на основе данных, поступивших на вход блока, и привязку результата выполнения алгоритма к выходу блока.

Локальные функции OnTick() каждого из функциональных блоков вызываются последовательно из глобальной функции OnTick() на этапе «выполнение программы» цикла работы

ядра логического управления (рисунок 3.13). Порядок вызова локальных функций `OnTick()` определяется пользователем на этапе формирования общей функциональной модели.

Во время выполнения программы логического управления актуальные значения состояний входов, выходов и переменных, относящихся к функциональным блокам, должны быть доступны пользователю. Для этого необходимо один раз в такт работы ядра логического управления осуществлять передачу актуальных данных.

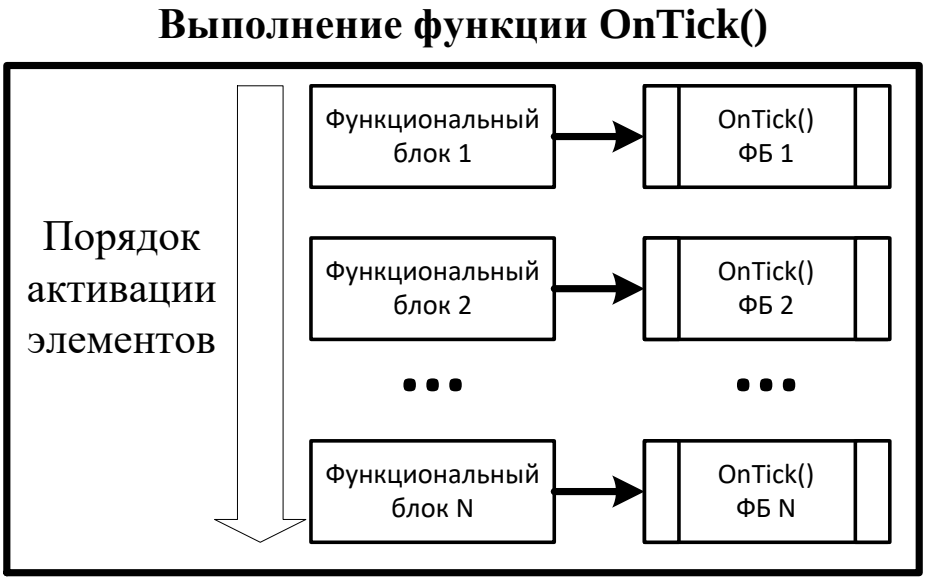


Рисунок 3.13 Порядок выполнения функции OnTick()

Обновление данных по всем функциональным блокам приведет к существенному увеличению нагрузки на канал обмена данными между подсистемой программирования и ядром логического управления. Для устранения этого при нажатии кнопки «ПУСК» происходит формирование списка обновляемых функциональных блоков, механизм которого представлен на рисунке 3.14. К числу обновляемых функциональных блоков относятся лишь те, которые находятся в активной области интерфейса оператора, которые формируются в подсистеме программирования вызовом функции `CreateSoftPlcListUpdate()`. Сформированный массив объектов имеет название `m_SoftPlcObjUpdList` и передаётся в исполнительное ядро. Если на этапе выполнения программы логического управления произошла критическая ошибка, то система переходит в состояние `SOFT_PLC_ERROR_C` и находится в нем до тех пор, пока пользователь не устранит возникшую ошибку и не переведет ядро логического управления в состояние `SOFT_PLC_READY_C`, используя команды подсистемы программирования.

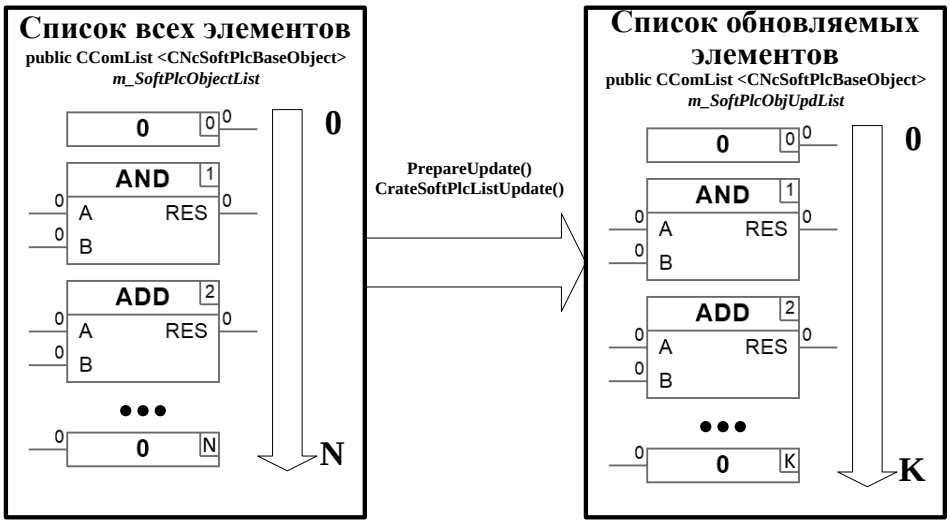


Рисунок 3.14 Механизм формирования списка обновляемых элементов

В некоторых случаях необходимо передать программную модель, хранящуюся в исполнительном ядре, в подсистему программирования (например, при подключении к ядру нового клиента), для этого предназначен механизм, представленный на диаграмме последовательности (рисунок 3.15).

До получения программной модели среда разработки программ логического управления формирует запрос о состоянии программной модели, находящейся в ядре (`GetModelState()`), по результатам которого принимается решение о возможности передачи данных. Если передача программной модели возможна, то первоначально запрашивается количество объектов программной модели (`GetCountsOfPlcObjects()`). После получения количества объектов формируется цикл, в каждой итерации которого запрашивается отдельный объект программной модели (`GetPlcObject()`). После получения всех объектов программной модели подсистема программирования последовательно запрашивает из исполнительного ядра идентификатор поделии (`GetModelId()`) и состояние модели (`GetModelState()`).

3.3.3 Организация структуры разделяемой памяти

Адресное пространство для синхронизации данных между аппаратными входами/выходами и программой логического управления реализовано на основе механизма разделяемой памяти (англ. shared memory). Это связано с тем, что время прямого обращения к физической памяти аппаратных устройств (зависящее от скорости поддерживаемого протокола коммуникации) много больше программного запроса на обмен данными с внутренним модулем ядра логического управления.

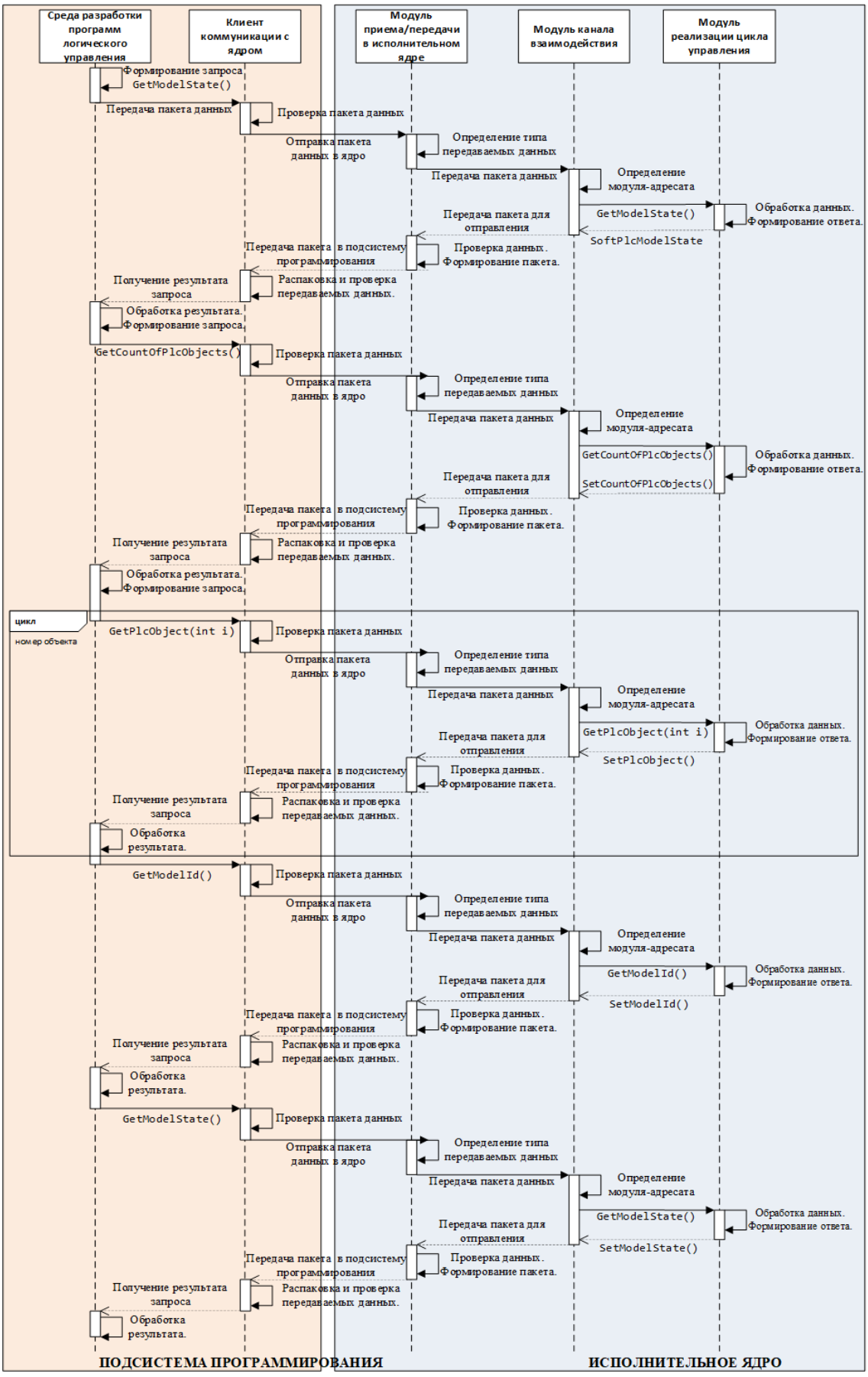


Рисунок 3.15 Диаграмма последовательности передачи модели из ядра

Подход к формированию адресного пространства был предложен в диссертационной работе «Разработка способа аппаратно-независимого управления электроавтоматикой для сокращения времени выпуска различных компоновок токарно-фрезерных станков с ЧПУ» Кулиевым А.У. [132] В указанной работе разделение адресного пространства было проведено между аппаратными входами/выходами помодульно. Однако практический опыт внедрения показал, что при обращении к разделяемой памяти удобно работать отдельно с массивом входов и отдельно - с массивом выходов. В связи с этим для системы логического управления предлагается осуществлять выделение отдельного подмножества адресного пространства для массива входных и выходных данных, что продемонстрировано на рисунке 3.16.

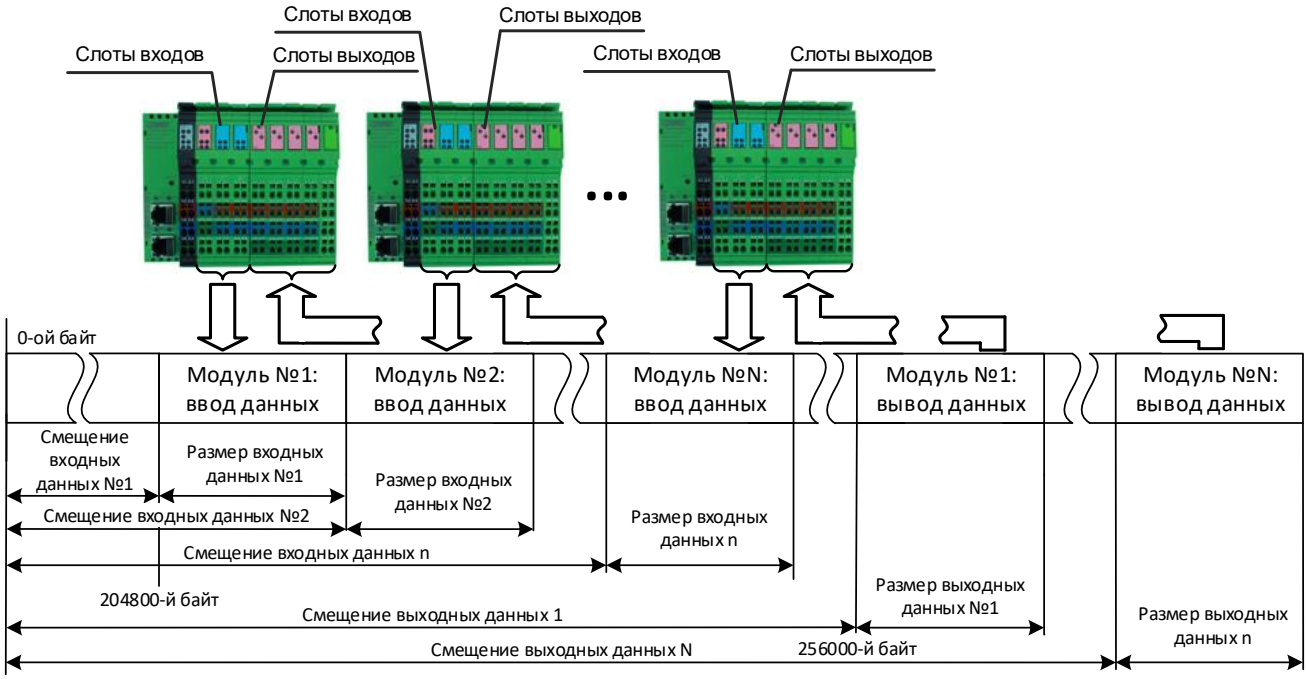


Рисунок 3.16 Адресное пространство ядра логического управления

Размер разделяемой памяти устанавливается пользователем в настройках системы. Общий объем разделяемой памяти по функциональному назначению разделен на 6 областей (рисунок 3.17):

- область 1 - для организации взаимодействия между ядром системы логического управления и системами управления верхнего уровня (например, ЧПУ);
- область 2 - «внутренняя» памятью ядра, которая хранит текущие данные (константы, результаты промежуточных вычислений и т.д.);
- область 3 - для организации обмена данными с аппаратными входами/выходами. Делится на подобласть входов и подобласть выходов;
- область 4 - хранит значения системных переменных;

- область 5 – интерфейс организации взаимодействия с системами управления движением (приводами);
- область 6 - резервная область для внедрения перспективных решений.

Рисунок 3.17 Циклы обновления данных в разделяемой памяти

Для работы с аппаратными входами/выходами в адресном пространстве ядра логического управления организована область 3 с выделением в ней подобластей работы с входными данными (с 204 800 до 255 999 байта) и выходными данными (с 256 000 до 307 199 байта). На каждом из участков третьей области разделяемой памяти, в соответствии с переданной в ядро конфигурацией, последовательно записываются данные с аппаратных модулей, начиная с младшего бита модуля с наименьшим номером физического адреса до старшего бита модуля с наибольшим номером физического адреса.

Для конфигурирования системы логического управления каждый из головных модулей ввода/вывода и слотов должен иметь свой адрес в рамках адресного пространства. Формулы для расчета параметров адресации каждого из модулей и слотов входов/выходов были предложены А.У. Кулиевым [132]. В них были внесены коррективы в соответствии с предложенными изменениями в механизме адресации входов/выходов в разделяемой памяти, далее приведен расчет адресации.

Уникальными идентификационными признаками слотов в системе логического управления выступают следующие величины:

- **S_S** - размера пакета данных слота. Это физическая характеристика слота, которая определяется по его паспортным данным.
- **O_S** - адрес смещения начального байта данных слота относительно начального байта головного модуля ввода/вывода. Для j-ого слота i-ого головного модуля равна сумме размеров пакетов данных предыдущих слотов i-го модуля по формуле

$$(O_S)_{ij} = \sum_{k=1}^{j-1} (S_S)_{ik} \tag{3.1}$$

Величину смещения данных для слотов, следующих за расчетным, определяется как сумма смещения данных и размер пакета данных расчетного слота:

$$(O_S)_{i(j+1)} = (O_S)_{ij} + (S_S)_{ij} \tag{3.2}$$

К параметрам, идентифицирующим головной модуль ввода/вывода в рамках конфигурации системы, относятся следующие:

- **Off_I** – начало отсчета байт входных данных модуля ввода/вывода относительно нулевого байта разделяемой памяти, для первого головного модуля значение 204800;
- **Off_Q** – начало отсчета байт выходных данных модуля ввода/вывода относительно нулевого байта разделяемой памяти для первого головного модуля значение 256000;
- **S_D** – общий размер пакета данных головного модуля ввода/вывода, определяется как сумма пакетов данных каждого из слотов подключенных к модулю и задается формулой:

$$(S_D)_i = \sum_{j=1}^{n_i} (S_S)_{ij} \tag{3.3}$$

- **O_D** – смещение начального байта данных головного модуля ввода/вывода относительно начального байта разделяемой памяти, определяется по формуле:

$$(O_D)_i = Off + \sum_{p=1}^{i-1} (S_D)_p \tag{3.4}$$

- **N** – порядковый номер байта данных j-ого слота i-ого модуля в разделяемой памяти, определяется по формуле:

$$N_{ij} = (O_D)_i + (O_S)_{ij} \tag{3.5}$$

Время выполнения цикла синхронизации данных между аппаратными входами/выходами и разделяемой памятью существенно меньше времени выполнения цикла ядра логического управления (рисунок 3.17). В связи с этим во время выполнения цикла работы ядра логического управления укладывается несколько циклов обновления данных, что позволяет поддерживать актуальные данные о состоянии аппаратных входов/выходов в разделяемой памяти системы логического управления. Разделяемая память является энергозависимой.

Время цикла обновления данных зависит от количества аппаратных модулей ввода/вывода, которые выбираются в зависимости от типа и сложности технологического оборудования. Для определения времени цикла обновления данных необходимо определить функцию T = *f* (V) зависимости периода (T) от объема передаваемых от аппаратных модулей данных (V).

Время цикла обновления данных от аппаратных модулей не должно превышать период одного цикла работы ядра логического управления. Прямое измерение периода одного цикла обновления данных не является достоверным, т.к. функции чтения/записи данных в библиотеке «SOEM» используемой в коммуникационном драйвере являются асинхронными. Это означает, что полученный в текущий момент времени пакет входных данных является ответом от объекта управления на пакет выходных данных, отправленный в предыдущий момент времени. Самостоятельная генерация события изменения состояния объекта управления с последующим отслеживанием времени его появления в разделяемой памяти позволит определить время реакции системы логического управления.

Экспериментальные исследования по определению времени реакции системы логического управления были проведены с изменением количества подключаемых аппаратных модулей ввода/вывода (в качестве которых использовались модули Beckhoff) на основе методики, предложенной в диссертационной работе А.У. Кулиева. [132] По полученным в ходе эксперимента данным был построен график зависимости периода обмена данными от объема передаваемых данных, представленный на рисунке 3.18.

Зависимость, полученная экспериментально, близка к линейному закону, который выражается формулой T = aV + b. Коэффициенты для функции, описывающей полученную прямую, вычисляются путем аппроксимации по методу наименьших квадратов:

$$b = \frac{\sum_{n=1}^{N} T_n \sum_{n=1}^{N} (V_n)^2 - \sum_{n-1}^{N} V_n \sum_{n=1}^{N} (T_n V_n)}{N \sum_{n=1}^{N} (V_n)^2 - (\sum_{n=1}^{N} V_n)^2} \tag{3.6}$$

$$a = \frac{N \sum_{n=1}^{N} (T_n V_n) - \sum_{n=1}^{N} T_n \sum_{n=1}^{N} V_n}{N \sum_{n=1}^{N} (V_n)^2 - (\sum_{n=1}^{N} V_n)^2} \tag{3.7}$$

Погрешность метода наименьших квадратов определяется по формуле:

$$\sigma = \frac{1}{N} \sum_{n=1}^{N} (T_n - (aV_n + b))^2 \tag{3.8}$$

Полученные и обработанные экспериментальные данные (таблица 3-4) позволили произвести расчет коэффициентов.

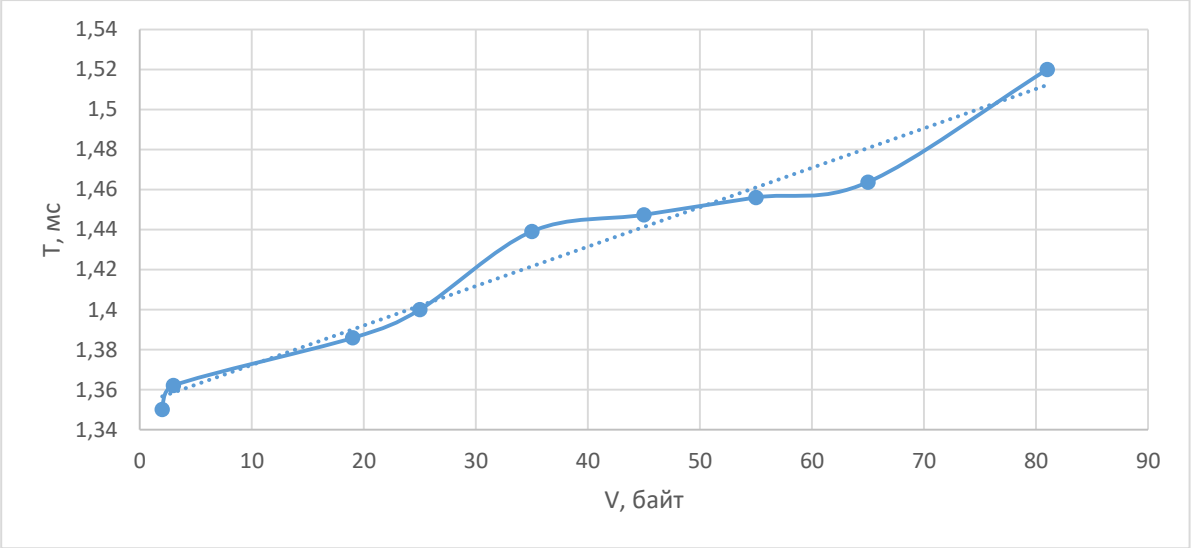


Рисунок 3.18 Зависимость периода цикла обмена данными от объема передаваемых данных

Таблица 3-4 Обработанные экспериментальные данные

n	**Vn**	**Tn**	**$(Vn)^2$**	**(Vn * Tn)**
1	2	1,3501011	4	2,700202
2	3	1,3621001	9	4,0863
3	19	1,3859003	361	26,33211
4	25	1,4000231	729	35,00058
5	35	1,4391001	1369	50,3685
6	45	1,4473841	2209	65,13228
7	55	1,4560454	3249	80,0825
8	65	1,4637351	4489	95,14278
9	81	1,5199871	6889	123,119
Σ	330	12,82438	18060	481,9642073

$$b = \frac{12{,}82438 * 18060 - 330 * 481{,}9642073}{9 * 18060 - 330^2} \approx 1{,}352724 \quad (3.9)$$

$$a = \frac{9 * 481{,}9642073 - 12{,}82438 * 330}{9 * 18060 - 330^2} \approx 0{,}001969 \quad (3.10)$$

Погрешность вычислений определяется по формуле (3.8) и составляет σ = 0,000087. В результате проведенных экспериментов получена зависимость периода цикла обновления данных от объема передаваемых данных:

$$T = (0{,}001969\,V + 1{,}352724) \pm 0{,}000087 \quad (3.11)$$

3.3.4 Программная реализация ядра системы логического управления

Программная реализация модулей ядра логического управления осуществлялась на языке программирования C++, в среде MS Visual Studio. Базовыми модулями ядра логического управления являются: модуль конфигурации библиотек функциональных блоков, отвечающих за набор и характеристики пользовательских функциональных блоков, используемых в проекте; модуль конфигурации аппаратных устройств, отвечающий за формирование в ядре логического управления структуры аппаратных устройств ввода/вывода; модуль связи с подсистемой программирования, отвечающий за реализацию канала взаимодействия подсистемы программирования и ядра логического управления; модуль конфигурации системы управления электроавтоматикой, отвечающий за состав и функционал системы логического управления в целом; модуль работы с разделяемой памятью, отвечающий за формирование и обмен данными с разделяемой памятью. Проект реализации ядра логического управления является проектом повышенной сложности, поэтому при декомпозиции задач было принято решение воспользоваться уровнями абстракции, которые позволяют скрыть детали реализации функциональных возможностей и сосредоточиться на концептуальных особенностях реализации модулей ядра. Диаграмма классов исполняемых модулей ядра логического управления в нотации UML представлена на рисунке 3.19.

Диаграмма отражает состав и взаимоотношения классов системы, их атрибуты и методы. Все наименования классов, переменных, констант и других идентификаторов представлены в венгерской нотации (соглашение о наименовании идентификаторов в программном коде). Наименование имени класса начинается с заглавной буквы "C" (например, `CNcAbstractSoftPlc`). Линии с различным набором стрелок изображают отношения между классами.

Основным классом ядра логического управления является класс `CSoftPlc`, который реализует функции по исполнению основного цикла работы системы логического управления и вспомогательные операции, среди которых можно выделить следующие: `OnTick()` – функция запускает расчет алгоритмов логического управления; `StartPLC()` – функция запускает рабочий цикл ядра логического управления; `ReadSpftPlcIo()` – функция позволяет считать заданную область разделяемой памяти и др.

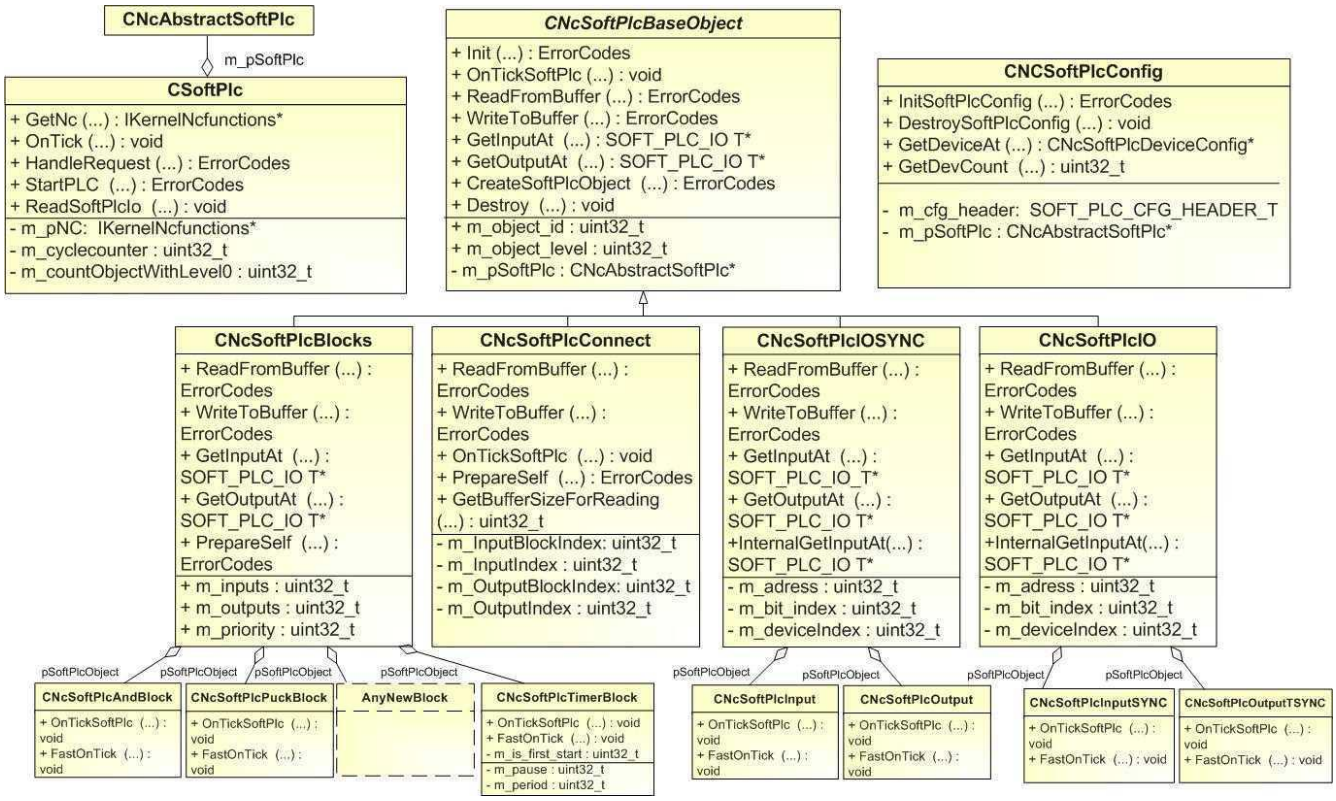


Рисунок 3.19 Диаграмма классов в нотации UML подсистемы логического управления

Класс `CNCSoftPlcConfig` реализует методы по работе с конфигурацией аппаратных входов/выходов в ядре логического управления. Среди функций указанного класса можно выделить следующие: `InitSoftPlcConfig()` – инициализирует конфигурацию аппаратных входов/выходов, полученную из подсистемы программирования (под инициализацией понимается распределение адресации участков разделяемой памяти под каждый аппаратный модуль); `DestroySoftPlcConfig()` – производит очистку конфигурации аппаратных входов/выходов в ядре логического управления; `GetDeviceAt()` – предоставляет конфигурацию аппаратных входов/выходов; `GetDeviceCount()` – предоставляет количество аппаратных модулей входов/выходов в текущей конфигурации и др.

Прототипом всех объектов, используемых при описании программы на языке функциональных блоков является касс `CncSoftPlcBaseObject`, от которого наследуются классы реализующие отдельные элементы языка программирования, такие как: функциональный блок, связь, аппаратный вход/выход, синхронизируемый аппаратный вход/выход. Класс имеет следующие m-поля: `m_object_id` – уникальный идентификатор элемента в рамках программы логического управления, `m_object_level` – уровень программной модели, на котором находится элемент. Класс реализует методы, позволяющие работать с элементами программной модели, в том числе: `OnTickSoftPlc()` – запуск расчета алгоритмов логического управления, `ReadFromBuffer()/WriteToBuffer()` – чтение и запись данных в разделяемую память, `GetInputAt()/GetOutputAt()` – получение объекта входов/выходов программной модели.

CNcSoftPlcBlocks – это базовый класс реализации функциональных блоков, который содержит поля и методы, которые поддерживаются всеми типами функциональных блоков. Среди m-полей указанного класса можно выделить три поля: m_inputs – количества входов блока, m_outputs – количества выходов блока, m_proirity – приоритет выполнения блока в общем списке функциональных блоков проекта. Функции базового класса при реализации класса функциональных блоков переопределяются. На базе класса CNcSoftPlcBlocks реализованы классы функциональных блоков, среди которых блоки: логических элементов, математических элементов, таймеры, счетчики, пользовательские и др.

CNcSoftPlcConnect – это класс, реализующий линии связи между функциональными блоками. Класс содержит следующие m_поля: m_InputBlockIndex / m_InputIndex – уникальные идентификаторы функционального блока и его выхода, из которого исходит связь, m_OutputBlockIndex / m_OutputIndex - уникальные идентификаторы функционального блока и его входа, в которые входит связь. Метод OnTickSoftPlc() вызывается один раз в цикл работы ядра логического управления для каждой связи и позволяет передавать значения с выходов функционального блока, полученные при выполнении алгоритма логического управления, на входы другого функционального блока.

Класс CNcSoftPlcIO содержит поля и методы работы с разделяемой памятью, посредством которых осуществляется доступ к аппаратным входам/выходам. Класс содержит следующие m_поля: m_adress / m_bit_index – адреса байта и номер бита в разделяемой памяти, к которому обращается вход или выход, m_deviceIndex – уникальный идентификатор модуля аппаратных входов/выходов, с которым работает экземпляр класса.

Класс CNcSoftPlcIOSYNC позволяет производить чтение/запись в разделяемую память по сигналу синхронизации. По реализации отличается от класса CNcSoftPlcIO методом OnTickSoftPlc(), в котором при успешности сигнала синхронизации производится чтение/запись данных в разделяемую память.

3.4. Реализация механизма взаимодействия подсистемы программирования и исполнительного ядра системы логического управления

В разделе 2.2 были описаны варианты решений, согласно которым подсистема программирования и подсистема ядра логического управления могут работать как на одном, так и на нескольких компьютерах, связанных между собой сетью. В связи с этим для организации взаи-

модействия между двумя подсистемами необходимо реализовать канал данных, способный работать при обоих вариантах реализации. В разделе 2.2 для указанных целей было предложено использование технологии сокетов.

Сетевое взаимодействие между аппаратными платформами целесообразно организовать посредством общедоступных и широко распространённых средств. На сегодняшний день к такого рода сетям можно отнести сети, реализованные на базе стандарта Ethernet с применением протокола TCP/IP. Взаимодействия между аппаратными устройствами с помощью стека протоколов TCP/IP используют адресацию (32-битный адрес, разбит на 4 поля разделенных точками, поле содержит 1 байт) и порты (номер порта в диапазоне от 0 до 65535). Пара адрес/порт является сокетом для семейства протоколов TCP/IP.

Для работы в рамках единой аппаратной платформы используется стандартное обращение этой платформы к самой себе посредством механизма локального хоста (англ. localhost). Для этого используется зарезервированный IP адрес — 127.0.0.1, который указан в специальном сетевом интерфейсе протокола TCP/IP, который называется «внутренняя петля» (англ. loopback). При установке соединения в рамках сети в которой присутствует лишь одна аппаратная платформа, сетевые протоколы выполняют функции взаимодействия между процессами.

Для реализации трансфера данных между подсистемой программирования и исполнительным ядром логического управления необходим универсальный механизм, который позволяет передавать разнородные типы данных, не требует жесткого специфицирования передаваемых пакетов и базируется на протоколе TCP/IP. Согласно указанным требованиям был выбран многоцелевой канал взаимодействия, реализованный в работе «Расширение функциональных возможностей специализированных систем ЧПУ посредством организации многоцелевого канала взаимодействия их основных компонентов» П.А. Никишечкиным. [133]

Основным элементом, который позволяет реализовать программную логику работы многоцелевого канала взаимодействия, выступает специализированный клиент, который располагается в подсистеме программирования, поддерживает ряд интерфейсов связи как с редактором программ логического управления, так и с исполнительным ядром логического управления (рисунок 3.20).

Для корректной работы системы логического управления между двумя подсистемами должны передаваться следующие наборы данных: объекты программы на языке функциональных блоков (функциональные блоки, блоки аппаратных входов/выходов, связи, пользовательские блоки), специализированные команды управления (запуск/пауза/останов работы программы), конфигурация аппаратных входов/выходов.

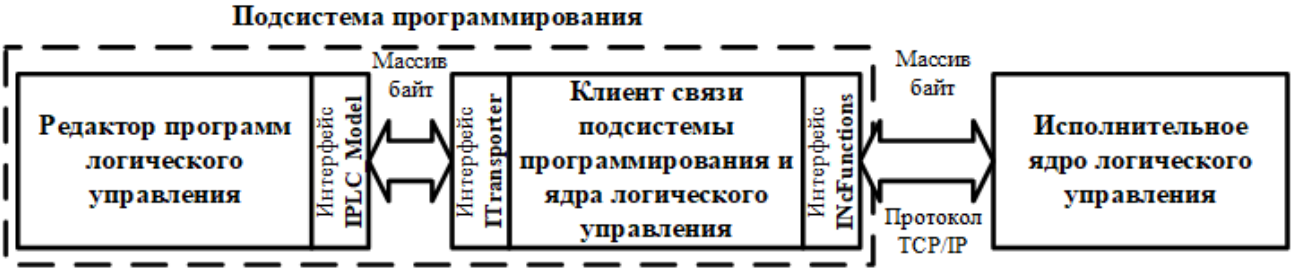


Рисунок 3.20. Схема взаимодействия редактора программ логического управления с ядром системы ЧПУ

При запуске программы логического управления последовательно производятся несколько действий: передача программы в исполнительное ядро логического управления и команды на ее запуск. Объем одного пакета данных не позволяет передать программу целиком, поэтому она передается поэлементно. В качестве примера сформированного пакета для передачи одного функционального блока приведен блок передачи логической операции «И» таблице 3-5.

Таблица 3-5 – Структура пакета данных, содержащего информацию о функциональном блоке логической функции «И»

№	Группа	Поле	Тип данных	Размер, байт	Значение поля	Описание
1	Основной заголовок	MessageSize	uint	4	402	Размер сообщения, включая все заголовки (в байтах)
2		MessageType	ushort	2	5	Тип сообщения (сообщение многоцелевого канала)
3	Заголовок многоцелевого пакета	ID	uint	4	3000	Идентификатор получателя данных – ядро логического управления
4		DataLength	uint	4	392	Длина передаваемых данных в байтах
5	Полезные данные	SoftPLC-Code	uint	4	51	Команда – передача программы логического управления
6			ushort	2	1	Тип объекта (логическое «И»)
7			uint	4	0	ID объекта
8			uint	4	0	Уровень объекта
9			uint	4	2	Системный id объекта (уникальный) – объект «И»
10			byte[]	32	AND	Имя объекта
11			byte[]	256		Дескриптор (описание) объекта
12			uint	4	40	Позиция в поле программы по оси X

Продолжение таблицы 3-5.

№	Группа	Поле	Тип данных	Размер, байт	Значение поля	Описание
13			uint	4	60	Позиция в поле программы по оси Y
14			uint	4	0	Приоритет блока
15			uint	4	2	Количество входов
16			uint	4	0	Индекс контакта
17			byte[]	12	A	Имя контакта
18			uint	4	1	Длина контакта
19			ushort	2	0	Тип контакта (Bit)
20			uint	4	1	Индекс контакта
21			byte[]	12	B	Имя контакта
22			uint	4	1	Длина контакта
23			ushort	2	0	Тип контакта (Bit)
24			uint	4	1	Количество выходов
25			uint	4	0	Индекс контакта
26			byte[]	12	RES	Имя контакта
27			uint	4	1	Длина контакта
28			ushort	2	0	Тип контакта (Bit)

Передаваемый пакет состоит из трех групп полей: основной заголовок, заголовок многоцелевого пакета и полезные данные.

В основной заголовок входят следующие поля: 1 - MessageSize, размер передаваемого пакета данных в байтах (в примере сообщение содержит 402 байта данных) и 2 - MessageType, тип передаваемого сообщения (в примере это сообщение многоцелевого канала, которое имеет идентификатор «5»).

Заголовок многоцелевого пакета содержит поля: 3 - ID, идентификатор модуля, которому предназначено сообщение (в примере сообщение передается ядру логического управления, которое имеет идентификатор 3000); 4 – DataLength, длинна передаваемых полезных данных (в примере 392 байта).

Полезные данные отображают поля, которые описывают передаваемый функциональный блок: поле 5 – SoftPLCCode, код команды для ядра логического управления (в примере это код, передачи программы логического управления), поле 6 - тип передаваемого объекта (это функциональный блок логической операции «И»), поле 7 - ID передаваемого функционального блока в программе логического управления, поле 8 - уровень (приоритет выполнения) передаваемого функционального блока в программе логического управления, поле 9 - системный идентификатор передаваемого объекта (для функционального блока «И» это «2»), поле 10 - текстовое поле с отображаемым в программе именем объекта (для функционального блока «И» это «AND»), поле 11 - тестовое поле с описанием объекта (описание задается программистом в момент добавления функционального блока в программу), поле с 12 по 28 - основные параметры

функционального блока, необходимые для корректного выполнения заданного алгоритма функционирования блока и отображения блока в области программы логического управления.

После подтверждения успешной передачи программы логического управления передается команда «ПУСК», которая запускает цикл работы исполнительного ядра. Структура многоцелевого пакета данных для команды «ПУСК» представлена в таблице 3-6.

Таблица 3-6 – Структура пакета данных с командой запуска цикла работы исполнительного ядра логического управления

№	Группа	Поле	Тип данных	Размер, байт	Значение поля	Описание
1	Основной заголовок	MessageSize	uint	4	18	Размер сообщения, включая все заголовки (в байтах)
2		MessageType	ushort	2	5	Тип сообщения (сообщение многоцелевого канала)
3	Заголовок многоцелевого пакета	ID	uint	4	3000	Идентификатор получателя данных – ядро логического управления
4		DataLength	uint	4	4	Длина передаваемых данных в байтах
5	Полезные данные	SoftPLC-Code	uint	4	54	Команда – запустить исполнительное ядро

Передаваемый пакет с командой состоит из двух групп полей, которые неизменны для всех типов передаваемых сообщений (основной заголовок, заголовок многоцелевого пакета) и полезных данных, которые уникальны для каждой команды. Полезные данные отображают поле 5 – SoftPLCCode, код команды для ядра логического управления (в примере это код «54», который дает команду ядру логического управления на запуск программы логического управления).

После успешного принятия команды запускается цикл работы исполнительного ядра системы логического управления, в котором происходит расчет алгоритмов управления согласно переданной программе. В обратном направлении, от исполнительного ядра логического управления к подсистеме программирования, отправляется служебная информация, например, текущее состояние программы логического управления (запущена, остановлена, программа отсутствует), которое отображается в статусной строке среды программирования. Если программа логического управления запущена в режиме отладки, то с заданной в настройках частотой происходит отправка сообщения о значениях входов, выходов функциональных блоков и линий связи. Это информация позволяет отслеживать нарушения алгоритма выполнения программы логического управления и производить мониторинг актуальных значений аппаратных входов/выходов в режиме реального времени.

3.5. Выводы

1. Построение профиля открытости на начальном этапе проектирования системы логического управления позволяет выбрать базовые стандарты, ориентируясь на их гармонизированные подмножества, предназначенные для проектирования конкретных модулей, функций или группы функций системы.
2. Сложное программное обеспечение системы логического управления реализовано путем разделения на две подсистемы: программирования и исполнительное ядро, между которыми предусмотрен специализированный канал взаимодействия, что позволяет упростить программно-математическое обеспечение и получить возможность устанавливать каждую из подсистем на отдельный компьютер.
3. Задачу организации подсистемы программирования можно решить двумя способами: использовать готовый программный продукт известного производителя или создать решение, исходя из собственных потребностей. Готовые продукты ориентированы на конкретную ОС, имеют дорогостоящую лицензию на коммерческое использование, закрыты на программном уровне, это не позволяет их применять для кроссплатформенных проектов, поэтому необходимо разрабатывать собственную среду проектирования и реализации логических программ стандарта МЭК 61131-3 с возможностью работы в автономном режиме без использования дорогостоящего оборудования.
4. Для привязки системы логического управления к определенному объекту управления предложена подсистема конфигурирования модулей ввода/вывода, которая позволяет привязать модули ввода/вывода к конкретным участкам разделяемый памяти, посредством которой реализован обмен данными с ядром логического управления.
5. Для реализации возможностей работы исполнительного ядра логического управления под управлением различных операционных систем использован подход, который заключается в обеспечении совместимости посредством реализации классов-оболочек, инкапсулирующих платформозависимые вызовы. Базовые классы, реализующие логику работы ядра логического управления, используют предоставляемые уровнем оболочек функции, что обеспечивает им платформонезависимую реализацию.
6. Для реализации трансфера данных между подсистемой программирования и исполнительным ядром логического управления использован многоцелевой канал взаимодействия, который позволяет передавать разнородные типы данных, не требует жесткого специфицирования передаваемых пакетов и базируется на протоколе TCP/IP.

Глава 4 Разработка методологических основ построения систем логического управления технологическим оборудованием

В связи появлением новых типов аппаратных устройств, развитием программной базы и новых подходов к проектированию систем логического управления возникает проблема изменения методологии построения, одним из вопросов которой является формализация этапов проектирования. В качестве цели в четвертой главе диссертационной работы ставится решение методологических проблем создания систем логического управления. Исследование методологических аспектов построения потребовало решения следующих задач:

- формирование базы аппаратных вычислительных устройств для построения систем логического управления;
- разработка основных положений методики построения систем логического управления;
- формирование принципов построения программ логического управления;
- систематизация математического аппарата проектирования программ логического управления;
- разработка основных положений тестирования систем логического управления.

Результаты, полученные при работе над главой, позволят формализовать процесс проектирования систем логического управления с учетом специфики, предъявляемой объектом управления.

4.1 Методологические аспекты выбора аппаратных средств для реализации систем логического управления

Разработка системы логического управления на аппаратном уровне до недавнего времени базировалась на специализированных модульных вычислительных устройствах – ПЛК, анализ которых был произведен в первой главе диссертационной работы. На сегодняшний момент проектирование уходит от жесткой привязки программного и математического обеспечения систем к понятийному аппарату принципиальных электрических схем. В связи с этим появляется возможность использовать в качестве аппаратной вычислительной платформы современные унифицированные решения (персональные, планшетные, одноплатные компьютеры и т.д.), которые позволяют применять стандартный инструментарий проектирования и реализации математического обеспечении систем управления (например, среды программирования Visual Studio, Code Blocks и др.).

Аппаратные платформы персональных компьютеров — это электронные вычислительные машины, обладающие универсальными функциональными возможностями, имеющие унифицированные интерфейсы взаимодействия и отличающиеся эксплуатационными характеристиками бытовых приборов [134]. Бурное развитие программно-аппаратного обеспечения персональных компьютеров напрямую влияет на возможности систем управления. На сегодняшний день в свободном доступе имеются ОСРВ (например, Linix RT), которые производителями систем управления используются в качестве готовой базы для развертывания на ней специализированного программного обеспечения. Это позволяет создать на базе персонального компьютера систему управления реального времени под задачи управления оборудованием.

Развитие аппаратных и программных решений в области персональных компьютеров происходит постоянно: увеличивается мощность вычислительного ядра, растет сложность программных решений. При этом многие производители промышленных систем управления поддерживают свои программно-аппаратные решения, которые отстают по техническим характеристикам от решений, применяемых в бытовых персональных компьютерах. В то же время, себестоимость решений на базе персональных компьютеров ниже, в связи с большим количеством предложений на рынке, которые способствуют снижению цены.

Также стоит отметить, что обмен информацией между промышленными системами управления различных производителей затруднен, в связи с применением разнородных протоколов связи. Это вынуждает интеграторов при внедрении конкретной системы управления в промышленную среду предприятия использовать дополнительные программно-аппаратные устройства согласования. В случае применения в качестве вычислительной платформы аппаратных устройств персональных компьютеров эта проблема не возникает, т.к. обмен информацией в компьютерных сетях стандартизован и для его реализации на рынке имеется большое количество дешевых аппаратных решений.

Наиболее оптимальным путем развития систем управления представляется возможность использования всех преимуществ аппаратной платформы персональных компьютеров. Это позволит потребителям и интеграторам систем управления получить систему, которую можно без больших финансовых затрат внедрить в существующую производственную среду.

При этом надо учитывать, что большинство аппаратных элементов персональных компьютеров не рассчитаны на внешние воздействия промышленной среды, поэтому их применение возможно лишь для систем управления, эксплуатируемых в неагрессивных условиях. Для работы в промышленных условиях аппаратные элементы персональных компьютеров должны быть помещены в специализированные корпуса, которые защищают от внешних воздействий. При этом для технологических процессов, предъявляющих требования к вибробезопасности,

внутри корпуса необходимо предусмотреть элементы виброзащиты. Изготовление специализированного корпуса позволит получить полноценный промышленный компьютер, готовый к эксплуатации и соответствующий по техническим характеристикам передовым решениям в области аппаратного обеспечения персональных компьютеров. Такой компьютер с установленным на него ядром логического управления может использоваться как отдельно (например, в качестве РАС), так и в рамках иных систем управления (например, в составе ЧПУ).

Аппаратная платформа одноплатных компьютеров. В последние десятилетия бурное развитие компьютерных технологий привело к появлению и популяризации в различных сферах жизни новых форм представления электронно-вычислительных машин, таких как: смартфоны, коммуникаторы, планшетные компьютеры, панельные компьютеры, одноплатные компьютеры и др. Все большее распространение в области промышленной автоматизации получают планшетные, панельные и одноплатные компьютеры. В системах логического управления целесообразно рассмотреть возможность использования одноплатных компьютеров, которые могут стать полноценной, многофункциональной и гибкой заменой ПЛК.

Одноплатный компьютер (английское single-board computer, SBC) – электронно-вычислительная машина, выполненная на одной печатной плате, имеющая в своем составе все необходимые аппаратные модули для полноценного функционирования (процессор, оперативная память, системы ввода/вывода и др.). Одноплатные компьютеры, в зависимости от модели и производителя, реализуют определенный набор интерфейсов связи. Одноплатные компьютеры являются безвентиляторными устройствами, что повышает надежность аппаратного обеспечения, питаются от разъёма USB и требуют ток в диапазоне 500-1500 мА. К недостаткам таких вычислительных платформ стоит отнести невозможность расширения их вычислительных ресурсов, т.к. все компоненты устройств монтированы на плату пайкой, однако, такой монтаж положительно сказывается на надежности из-за отсутствия разъемных соединений.

В цифровых производствах системы логического управления на базе одноплатных компьютеров могут быть использованы для реализации принципа «интернет вещей», который является одним из 5 основных в концепции «Индустрия 4.0» (английское «Internet of Things» - IoT). Для этих целей корпорация Microsoft выпустила специальную версию Windows 10 IoT Edition [135]. При использовании указанной ОС система управления сможет взаимодействовать практически с любым интеллектуальным устройством, что позволит убрать устройства маршрутизации при объединении систем в локальные и глобальные вычислительные сети.

Основным средством связи одноплатных компьютеров с датчиками и исполнительными устройствами являются порты GPIO. Это универсальные порты, которые в зависимости от настроек могут быть или входами, или выходами. Стандартный одноплатный компьютер оснащен от пяти до десяти портами GPIO, но их количество можно расширять по необходимости.

4.2 Разработка методики построения систем логического управления технологическим оборудованием

Процесс проектирования и реализации систем управления итеративен, сложен и неоднозначен в выборе методов и средств. На рынке нет готовых решений для построения систем логического управления. В связи с этим возникает необходимость в разработке инструментария построения указанного типа систем. [136-137] Для этого предложена методика, которая определяет фиксированный набор практических шагов, приводящих к получению системы управления, удовлетворяющей требуемым характеристикам. В качестве входных данных методика использует техническое задание на разработку системы, в котором содержатся характеристики и функциональные возможности будущей системы. На выходе, после прохождения всех шагов методики, имеем систему, соответствующую заявленным характеристикам. Графически методика построения представлена на рисунке 4.1. Рассмотрим пошагово каждый из её этапов.

Шаг 1. Адаптация моделей системы управления под конкретный объект управления. На начальном этапе производится анализ и адаптация разработанного в разделе 2.3 инструментария общих моделей проектирования системы логического управления к конкретному объекту управления. На основании предложенной последовательности действий по созданию и трансформации моделей осуществляется выбор методов и способов разработки системы управления. В результате моделирования получим ряд основных модулей системы логического управления, определим их функции и взаимосвязь модулей между собой, что позволит существенно упростить процесс дальнейшей разработки.

Шаг 2. Разработка сетевой структуры системы управления. Процесс проектирования и разработки системы управления характеризуется: итеративностью, многоуровневостью и многоэтапностью. В распределенных системах управления применяется модульный принцип организации управления на основе иерархической многоуровневой схемы, в основе которого лежат понятия процесса, уровня управления, интерфейса и протокола связи. Основываясь на указанных понятиях предлагается на втором шаге разработки формировать сетевую структуру системы логического управления. Разрабатываемая сетевая структура должна однозначно определять: иерархические уровни из которых состоит система управления; оборудование, датчики и средства диагностики объекта управления; необходимые аппаратные вычислительные ресурсы; аппаратные модули ввода/вывода и протоколы их подключения к ядру логического управления; связи с локальными системами управления и системами управления верхнего уровня.

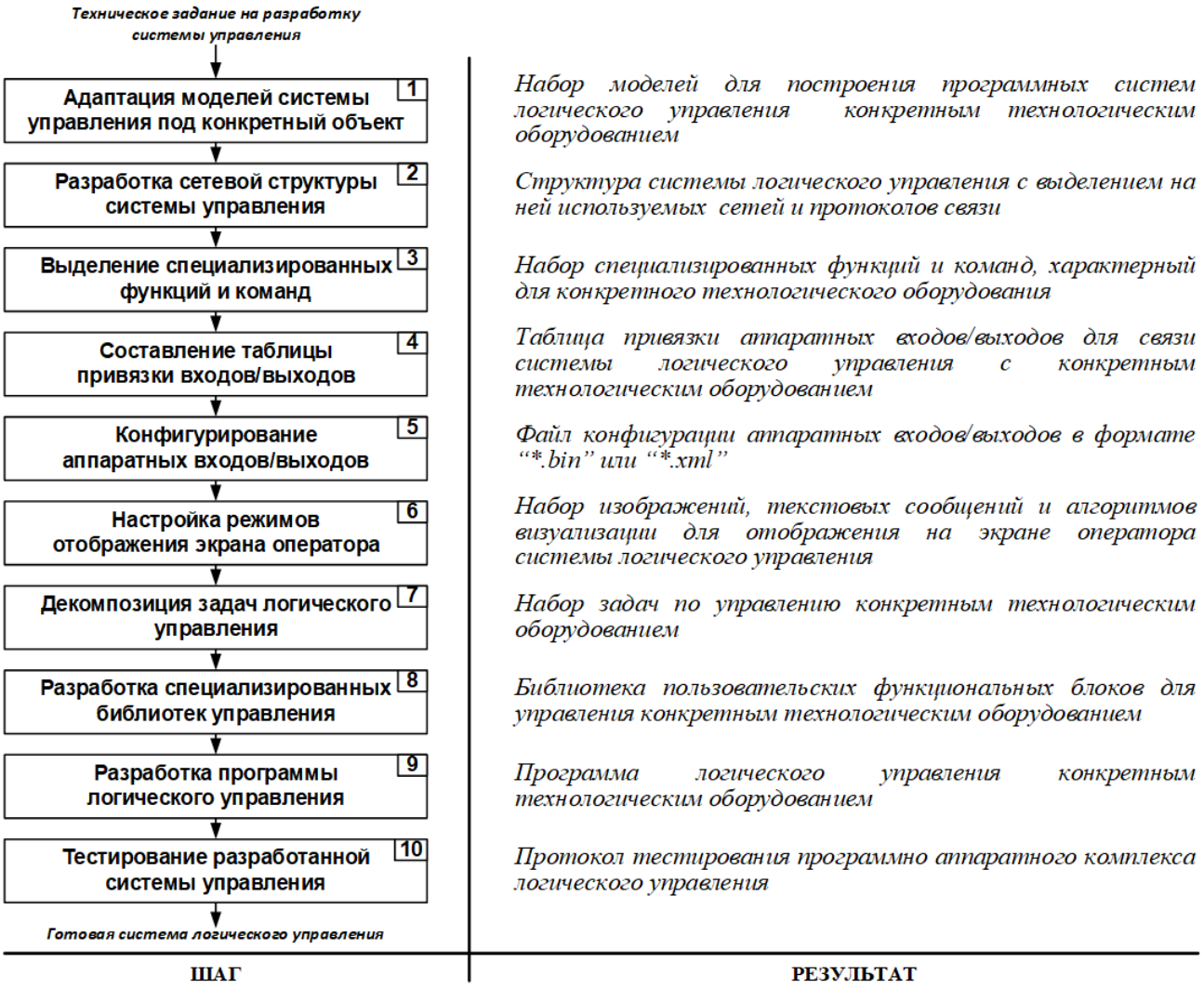


Рисунок 4.1 Методика построения систем логического управления

Шаг 3. Выделение специализированных функций и команд. Система логического управления применяется для автоматизации широкого круга технологического оборудования и технологических процессов, каждый из которых имеет специфические особенности и функции. Для осуществления поддержки полного набора функций объекта управления необходимо определить набор специфичных для объекта управления команд (например, команды управления лазером, для лазерного оборудования). Указанные команды должны быть систематизированы и по ним должны быть определены условия их активации и деактивации.

Шаг 4. Составление таблицы привязки входов/выходов. Для выполнения следующего шага необходимо провести анализ принципиальной электрической схемы объекта управления с целью выделения узлов объекта и определения количества и типа аппаратных входов/выходов. Результатом анализа является таблица привязки аппаратных входов/выходов, которая содержит: количество головных модулей (модулей организации связи); количество слотов вхо-

дов/выходов, физически подключаемых к конкретному головному модулю; тип слотов входов/выходов (аналоговый, дискретный, подключения термосопротивлений и др.); адресацию слотов входов/выходов.

Шаг 5. Конфигурирование аппаратных входов/выходов. При работе с программой логического управления требуется определить зависимость между ячейками разделяемой памяти ядра логического управления и аппартными входами/выходами. Для этого необходимо воспользоваться инструментарием конфигурирования аппаратных входов/выходов описанным в разделе 3.2.2.

Шаг 6. Настройка режимов отображения экрана оператора. Для взаимодействия с оператором технологического оборудования в системе управления предусматривают специализированный экран. На сегодняшний момент разнообразие экранов оператора достаточно велико, от семисегментных ячеек, способных отображать один символ, до экранных панелей с высоким графическим разрешением и функцией “touchscreen”. Каждый из этих экранов нуждается в настройке режимов отображения, которые задаются программно в ядре системы управления. Механизм настройки определяется типом экрана оператора.

Для цветных графических панелей оператора формируется графический интерфейс пользователя - GUI, который представляет собой доступные пользователю системные объекты и функции в виде графических компонентов (иконки, текстовые сообщения, меню, кнопки и т.д.). При этом оператор имеет доступ с помощью вспомогательных устройств (клавиатура, мышь и т.д.) ко всем отображаемым объектам экрана. Обычно GUI экрана оператора – это отдельный программный продукт, со своей архитектурой и программными компонентами, на котором есть выделенные области, которые могут быть изменены из программы логического управления. Механизм взаимодействия программы логического управления и GUI экрана оператора определяется разработчиками программного обеспечения графического интерфейса оператора. [138]

Шаг 7. Декомпозиция задач логического управления. На начальном этапе проектирования программы логического управления производится декомпозиция исходной задачи. Декомпозиция позволяет заменить решение одной сложной задачи решением ряда взаимосвязанных задач меньшего объема и позволяет анализировать любой объект управления как состоящий из отдельных взаимосвязанных подсистем, которые могут быть реализованы отдельно. Указанный подход позволяет получить независимые функциональные блоки, реализующие работу отдельных подсистем, с возможностью их дальнейшего объединения в библиотеки пользовательских функциональных блоков.

Шаг 8. Разработка специализированных библиотек управления. В разделе 3.2 была рассмотрена среда проектирования и разработки программ логического управления, последующие

два шага выполняются в указанной среде. Язык функциональных блоков, реализованный в среде программирования, поддерживает принцип декомпозиции программных компонент и позволяет разделять программу на подпрограммы, называемые пользовательскими функциональными блоками. Пользовательские блоки реализованы на базе стандартного набора функциональных блоков, объединенных в единый блок с выделением на нем входов и выходов (рисунок 4.2). Каждый пользовательский блок также может содержать в себе специализированные пользовательские функциональные блоки, при этом допускается до семи уровней вложенности. [139]

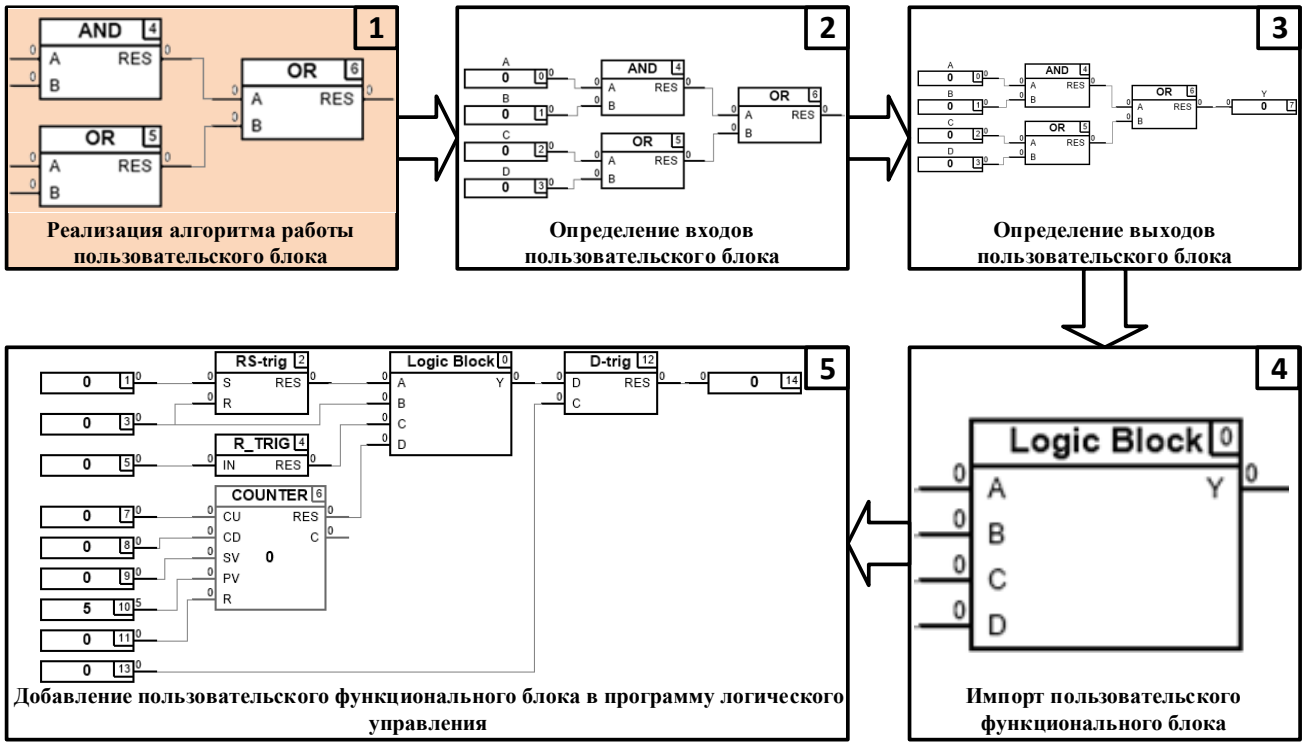


Рисунок 4.2 Последовательность создания пользовательского функционального блока в среде разработки программ логического управления

К преимуществам реализации пользовательских функциональных блоков можно отнести следующие:

- Обозримость и удобство работы с кодом программы логического управления. Визуальные языки программирования, к которым относится и язык функциональных блоков, интуитивно понятны, однако становятся громоздкими и неудобными в использовании с ростом количества блоков. Пользовательские функциональные блоки позволяют скрыть часть кода, относящуюся к конкретному узлу или технологическому элементу, что позволяет получить визуально структурированную программу логического управления и значительно сократить её объем.

- Возможность повторного применения набора библиотек функциональных блоков, реализующих логику работы с технологическим оборудованием или отдельными технологическими узлами. Такой подход позволяет повторно использовать ранее полученный программный код и дает возможность, при наличии набора пользовательских библиотек, существенно сократить время разработки программ логического управления.

Пользовательские функциональные блоки объединяются в библиотеки и хранятся в специализированных файлах с расширением «*.fbl», которые в дальнейшем могут быть импортированы в любую программу логического управления.

Шаг 9. Разработка программы логического управления. Программы логического управления разрабатываются исходя из технического задания, которое содержит алгоритмы и циклограммы работы объекта управления. Программа состоит из основной программы, которая хранится в отдельном файле файловой системы с расширением «*.fbv», и множества библиотек пользовательских подпрограмм, хранящихся в файлах с расширением «*.fbl». Разработка программы логического управления выполняется в специализированном редакторе (описан в разделе 3.2.1). Первоначально производится добавление функциональных блоков из меню в рабочую область редактора с использованием технологии «drag & drop». Меню редактора содержит следующие типы функциональных блоков, доступных разработчику: входы/выходы, стандартные функциональные блоки, пользовательские функциональные блоки.

Функциональные блоки входов/выходов могут содержать константы или текущие значения привязанных к ним аппаратных входов/выходов. Каждый вход/выход имеет следующие параметры для настройки: тип функционального блока (аппаратных вход/выход, константа), тип хранимого значения (целое число, вещественное число, битое поле) и адрес ячейки разделяемой памяти (для аппаратных входов/выходов).

Стандартный набор функциональных блоков, представленных в редакторе. позволяет реализовать полноценную программу логического управления и включает следующие типы блоков: вход и выход, логические операции, математические операции, триггеры, счетчики, таймеры блоки управления движением.

Следующим этапом создания программы логического управления является отладка, которая может быть выполнена в режиме эмуляции без непосредственного подключения к объекту управления. При удачном завершении отладки программа загружается в ядро системы и может быть запущена.

Шаг 10. Тестирование разработанной системы управления. Под тестированием системы логического управления будем понимать процесс обнаружения ошибок в программной и аппаратной частях путем исполнения программного кода на вычислительных ресурсах си-

стемы с входным набором тестовых данных. В ходе тестирования осуществляется: сбор динамических характеристик системы управления на базе конкретной операционной системы; выявление дефектов, ошибок и отказов, причиной которых становятся нерегулярные ситуации или аварийное прекращение работы.

4.3 Систематизация математических методов, используемых при проектировании программ логического управления

Область применения систем логического управления накладывает свои ограничения и требования к разработке программ управления, что определяет различие сложившихся подходов к применению математических методов, применяемых при их проектировании. Единый подход к использованию математического аппарата для проектирования программ логического управления невозможен в связи с тем, что каждый из методов имеет свои преимущества и недостатки, которые должны быть учтены при решении определенного класса задач. В связи с этим возникает необходимость систематизации математических методов, используемых при проектировании программ логического управления. Для этого рассмотрим процесс программирования систем логического управления.

Проектирование программного обеспечения для любой системы управления должно начинаться с изучения технического задания, в котором указаны основные критерии и характеристики будущего программного продукта. Определение функций системы осуществляется по результатам анализа технического задания и обсуждения с заказчиком особенностей работы технологического процесса и оборудования. Функции программы логического управления сводятся к выполнению цикла работы логического контроллера, описанному в разделе 3.3.1. Для успешного выполнения указанного функционала необходимо реализовать:

- первичную обработку и верификацию входных сигналов;
- программную разработку алгоритмов логического управления;
- формирование выходных сигналов, в том числе аналоговых;
- поддержку обмена данными с системами управления верхнего уровня.

Рассмотрим этап программной реализации алгоритмов логического управления, который включает в себя этапы: декомпозиции исходной задачи на подзадачи, проектирования и математического моделирования, верификации математической модели, программной реализации на конкретном языке программирования и тестирования.

На начальном этапе реализации программы производится декомпозиция исходной задачи управления. Декомпозиция позволяет заменить решение одной сложной задачи решением

ряда взаимосвязанных задач меньшего объема и позволяет анализировать любой объект управления как состоящий из отдельных взаимосвязанных подсистем, которые могут быть также разделены на части.

При проведении декомпозиции необходимо руководствоваться следующими правилами:

- разделение задач производится по уровням декомпозиции, при этом исходный объект управления находится на нулевом уровне;
- разделение производится по единому признаку внутри уровня;
- полученные подсистемы должны в полной мере характеризовать исходный объект управления или подсистему предыдущего уровня;
- глубина декомпозиции определяется исходя из квалификации программистов, которые будут реализовывать полученные подсистемы.

В качестве примера декомпозиции рассмотрим вариант разбиения задач логического управления для многофункциональных станков (рисунок 4.3).

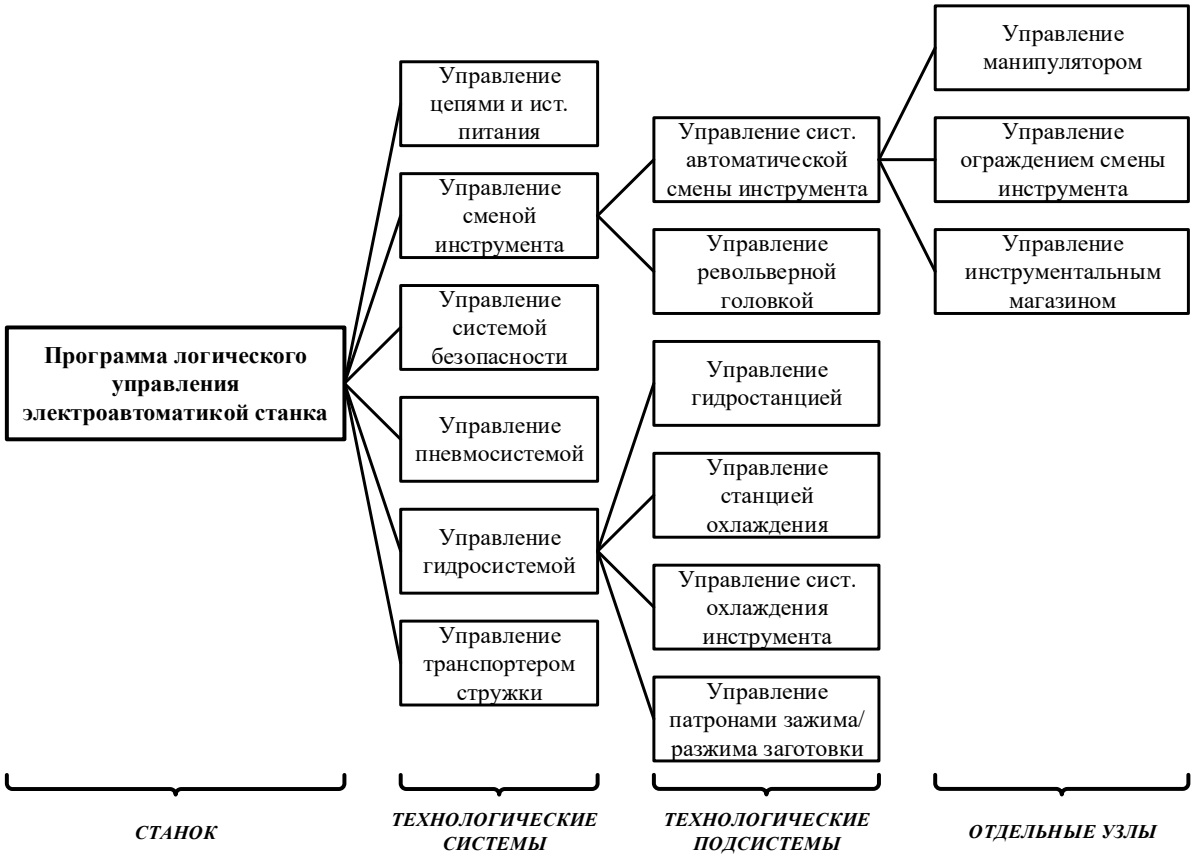


Рисунок 4.3 Декомпозиция задач логического управления (на примере станков)

Каждая из полученных при декомпозиции задач может быть разделена на более мелкие, вплоть до «атомарных» задач, состоящих из одного действия. Для программирования систем логического управления был предложен специализированный инструментарий, описанный в разделе 3.2, в котором для реализации программ используется язык функциональных блоков,

который позволяет осуществлять прямой переход от принципиальных электрических схем к программе логического управления. Каждая из отдельных подсистем объекта управления, полученная в результате декомпозиции, реализуется отдельным функциональным блоком, для формализации описания которого можно использовать различные математические аппараты. Выбор математического аппарат должен быть обусловлен имеющимися инструментальными средствами, которые позволят наиболее оптимально реализовать полученные математические модели. К числу наиболее часто применяемых при разработке программ логического управления математических аппаратов можно отнести следующие: Булева алгебра, автоматные модели, сети Петри, операторные формулы. В связи с выбором в работе в качестве базового языка программирования функциональных блоков нецелесообразно применять математический аппарат сетей Петри и операторных формул.

Сети Петри - визуальный аппарат для моделирования динамических дискретных систем. Математическое представление в виде сетей Петри легло в основу языка реализации программ логического управления SFC (англ. Sequential Function Chart, язык последовательных функциональных схем), поэтому математическое описание дискретной системы в виде сетей Петри целесообразно применять именно с языком программирования SFC. Операторный формулы – математический аппарат описания дискретных систем, который позволяет осуществить легкий переход от задания систем в виде циклограммы (временной диаграммы сигналов) к программной реализации на языке РКС.

В этой связи для программной реализации систем логического на языке функциональных блоков возможно использование Булевой алгебры и автоматных моделей в качестве базовых математических аппаратов. Однако это касается лишь дискретных систем, в которых квантование сигналов производится по уровням логической единицы и нуля, т.е. для процессов, управляемых логическим контроллером посредством цифровых модулей ввода/вывода. Для непрерывных процессов, которые можно рассматривать в качестве дискретных с квантованием сигналов по времени и управляемых логическим контроллером посредством аналоговых модулей ввода/вывода, необходим иной математический аппарат. Такие процессы чаще всего можно задать с помощью дифференциальных уравнений различного порядка. В таблице 4-1 приведена систематизация математических методов, используемых при программировании систем логического управления. Каждый из выделенных методов целесообразно использовать для определенных групп задач логического управления. Рассмотрим варианты выбора каждого из методов.

Математический аппарат Булевой алгебры применяется для наиболее простых задач логического управления, когда выходные сигналы зависят только от сигналов на входе системы управления в текущий момент времени. Такие системы в электронике реализованы на основе

комбинационных схем. Задание начальных условий производится с помощью таблицы истинности. Далее производится переход от табличного задания функций к математическому заданию, используя либо дизъюнктивную, либо конъюнктивную нормальную форму функций. По полученной функции можно построить функциональную схему, выразив которую в терминах языка функциональных блоков можно получить программу логического управления. Это не требует высокой квалификации персонала, с такой задачей может справиться техник, имеющий среднее специальное образование с небольшим опытом работы.

Таблица 4-1 Систематизация методов проектирования программ логического управления

	Мат. аппарат	**Область применения**	**Способ задания**	**Графическое представление**	**Требования к квалификации**
Комбинационные схемы	Булева алгебра	Объекты управления, в которых комбинация сигналов на выходе в любой момент времени однозначно определяется комбинацией сигналов на входе.	таблица истинности	функциональная схема	низкие
Технологические объекты	Временные булевы функции	Объекты, поведение которых определяется, в том числе, сигналами задержки времени, реализация которых осуществляется с использованием таймеров.	временная диаграмма, таблица истинности	временная диаграмма	средние
Цикловая электроавтоматика	Автоматные модели	Объекты управления, поведение которых определяется множеством дискретных операций, повторяющихся циклически с определенным периодом.	циклограмма	автоматный граф	средние
Дискретные системы	Разностные уравнения	Объекты управления, в которых можно осуществить квантование непрерывных входных и выходных сигналов по времени.	математическая модель	структурная схема	высокие

В задачах, когда требуется работа с сигналами задержки времени, реализация которых осуществляется с использованием таймеров, прямое применение функций Булевой алгебры невозможно. В этом случае целесообразно воспользоваться временными Булевыми функциями. Для этого осуществляется квантование выходного сигнала таймера по времени и этот сигнал используется как одна из переменных в Булевой функции. Начальные условия решаемой задачи могут быть заданы таблицей истинности или временной диаграммой. В качестве временной диаграммы используют участок циклограммы технологического оборудования, на котором выходные сигналы определяются сигналами на входе системы управления в текущий момент времени и сигналами с таймера. Для программной реализации указанного типа задач необходимо привлекать техников, имеющих большой опыт разработки программ логического управления.

Для объектов управления, поведение которых определяется множеством дискретных операций, повторяющихся циклически с определенным периодом, применение описанных выше методов невозможно. Наиболее полно характеризует условия задач такого типа математический аппарат автоматных моделей, который позволяет учитывать циклические процессы. Условия работы технологического оборудования в этом случае задается циклограммами, анализ которых позволяет построить автоматный граф, содержащий состояния и условия перехода между ними. При программировании системы логического управления на языке функциональных блоков каждое из полученных состояний может быть выражено отдельным функциональным блоком. Для программной реализации указанного типа задач необходимо привлекать техников и инженеров, имеющих существенный опыт разработки программ логического управления.

Программирование объектов, которые управляются непрерывными сигналами, квантование которых не производится по уровням логического нуля и единицы, осуществить перечисленными выше методами невозможно. Зачастую закон управления таких объектов задается дифференциальным уравнением, прямое решение которого ограниченными средствами систем логического управления не представляется возможным. Одним из примеров такой задачи могут служить регуляторы различного назначения (например, ПИД регулятор). Для программной реализации указанного типа задач целесообразно осуществлять преобразование дифференциального уравнения в разностное с помощью дискретизации функции, при условии, что квантование будет осуществляться с малой длительностью такта. Полученное рекуррентное уравнение можно решить программно, используя инструментарий программирования логической задачи управления. Рассмотрим подробнее каждый из математических аппаратов с примерами объектов и программ логического управления.

4.3.1 Программирование комбинационных схем с использованием математического аппарата Булевой алгебры

В простейших случаях при программировании системы логического управления используется математический аппарат на основе Булевой алгебры. Например, для программирования устройств, у которых комбинация сигналов на выходе в любой момент времени однозначно определяется комбинацией сигналов на входе, можно использовать теорию (n, m) - полюсников. Пусть имеется система из *m* функций:

$$\begin{cases} \varphi_1(x_1, x_2, ..., x_n) = y_1; \\ \varphi_2(x_1, x_2, ..., x_n) = y_2; \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ \varphi_m(x_1, x_2, ..., x_n) = y_m. \end{cases} \quad (4.1)$$

Требуется построить схему, в которой работа *i*-го выхода определялась бы функцией φ_i. Такие схемы, имеющие *n* входов и *m* выходов, называются (*n, m*) -полюсниками. Функция φ_i определяет некоторый (*n,* 1) - полюсник. Схема для всей системы будет совокупность *m* таких (*n,* 1) - полюсников. Рассмотрим пример программирования некоторых функций электроавтоматики токарного станка с ЧПУ СА-700 производства ОАО «САСТА»: функции аварийного останова $\varphi_{АВАР}$, функции подачи сигнала включения приводов φ_{ENABLE} и функции готовности гидравлической системы $\varphi_{ГИДР}$. Для программирования функционала нам понадобятся следующие сигналы, представленные в таблице.

Таблица 4-2 Сигналы электроавтоматики станка СА-700

Обозначение сигнала	**Описание сигнала**
X_0	Аварийный конечный выключатель в начале оси X
X_{MAX}	Аварийный конечный выключатель в конце оси X
Y_0	Аварийный конечный выключатель в начале оси Y
Y_{MAX}	Аварийный конечный выключатель в конце оси Y
A_{DOOR}	Датчик закрытия защитного кожуха
$A_{ПИТ}$	Сигнал наличия питания на приводах
B_X	Сигнал готовности привода оси X
B_Y	Сигнал готовности привода оси Y
$A_{ДАВ}$	Сигнал наличия давления в гидравлической системе

Функция аварийного останова активируется и останавливает привода в тот момент, когда выполняется одно из условий: нажаты конечные выключатели X_0, X_{MAX}, Y_0, Y_{MAX} или не закрыт защитный кожух, т.е. активен сигнал A_{DOOR}. Указанная функция может быть задана таблицей истинности следующего вида:

Таблица 4-3 Таблица истинности функции «авария»

X_0 **v** X_{MAX}	Y_0 **v** Y_{MAX}	A_{DOOR}	$\varphi_{АВАР}$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

По таблице истинности составляем Булеву функцию:

$$\varphi_{ABAP} = (X_0 \vee X_{MAX}) \vee (Y_0 \vee Y_{MAX}) \vee A_{DOOR} \quad (4.2)$$

Функция подачи сигнала включения приводов активна тогда, когда имеется сигнал готовности с обоих приводов B_X, B_Y и имеется сигнал наличия питания на приводах $A_{ПИТ}$. Функция задается следующей таблицей истинности:

Таблица 4-4 Таблица истинности функции подачи питания на привод

АПИТ	BX	BY	φENABLE
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

По таблице истинности получаем функцию следующего вида:

$$\varphi_{ENABLE} = A_{ПИТ} B_X B_Y \quad (4.3)$$

Функции готовности гидравлической системы зависит от двух сигналов: наличия питания $A_{ПИТ}$ и наличия необходимого давления $A_{ДАВ}$ и задается следующей таблицей истинности.

Таблица 4-5 Таблица истинности функции готовности гидравлической системы

$A_{ПИТ}$	$A_{ДАВ}$	$\varphi_{ГИДР}$
0	0	0
0	1	0
1	0	0
1	1	1

По таблице истинности получаем функцию следующего вида:

$$\varphi_{ГИДР} = A_{ПИТ} A_{ДАВ} \quad (4.4)$$

Объединив, получим систему вида:

$$\begin{cases} \varphi_{ABAP} = (X_0 \vee X_{MAX}) \vee (Y_0 \vee Y_{MAX}) \vee A_{DOOR}; \\ \varphi_{ENABLE} = A_{ПИТ} B_X B_Y; \\ \varphi_{ГИДР} = A_{ПИТ} A_{ДАВ}. \end{cases} \quad (4.5)$$

Построив схему для каждой функции в отдельности в среде программирования, получим совокупность схем, показанную на рисунке 4.4.

Такой синтез не всегда будет оптимальным, при наличии большого количества сигналов необходимо пользоваться методами простых импликант или каскада для синтеза (n, m) - полюсников, широко применяемыми в технике.

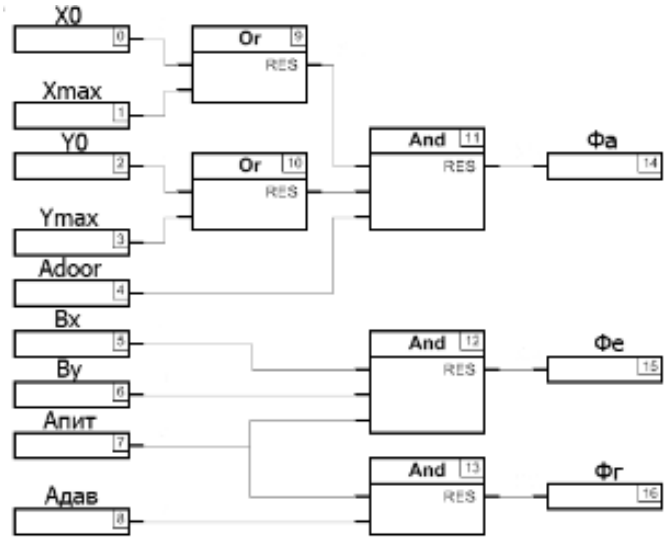


Рисунок 4.4 Логическая программа реализации (n, m) - полюсника для управления отдельными функциями станка СА-700

4.3.2 Программирование цикловой электроавтоматики с использованием математического аппарата временных Булевых функций

«Рассмотрим пример запуска системы со смазочно-охлаждающей жидкостью (СОЖ) на токарном станке СА-700. Часть циклограммы работы станка представлена на рисунке 4.5.

Для написания программы логического управления по временной диаграмме работы системы СОЖ (позиция 9) необходимо определить зависимость функции включения узла СОЖ - $\varphi_{СОЖ}$ от других функций и сигналов, представленных на циклограмме (таблица 4-6).

Таблица 4-6 Сигналы электроавтоматики циклограммы работы СОЖ

Обозначение сигнала	Описание сигнала
$X_{СТАРТ}$	Сигнал начала работы
$X_{СТОП}$	Сигнал окончания работы
$\varphi_{СТАНКА}$	Функция работы станка
$X_{ОШИБКА}$	Сигнал возникновения ошибки
$X_{ШПИНДЕЛЬ}$	Сигнал запуска шпинделя
$\varphi_{СОЖ}$	Функция запуска системы СОЖ

Работа станка начинается после нажатия на кнопку СТАРТ – сигнал $X_{СТАРТ}$, при этом активируется реле с самоподхватом, отвечающее за работу станка – функция $\varphi_{СТАНКА}$. Для осуществления формообразования на станке необходимо запустить шпиндель, сигнал $X_{ШПИНДЕЛЬ}$, при активации данного сигнала автоматически должна запуститься система подачи СОЖ. Возникновение ошибки при работе оборудования отмечается сигналом $X_{ОШИБКА}$, при этом будет

отключено вращение шпинделя и подача СОЖ, после устранения ошибки работа станка продолжится, однако, для вращения шпинделя необходимо активировать сигнал $X_{ШПИНДЕЛЬ}$. Подача СОЖ прекращается либо при возникновении ошибки, либо через 3 секунды после остановки шпинделя. Задержка в 3 секунды необходима для гарантированного окончания движения формообразования без нагрева детали или инструмента.

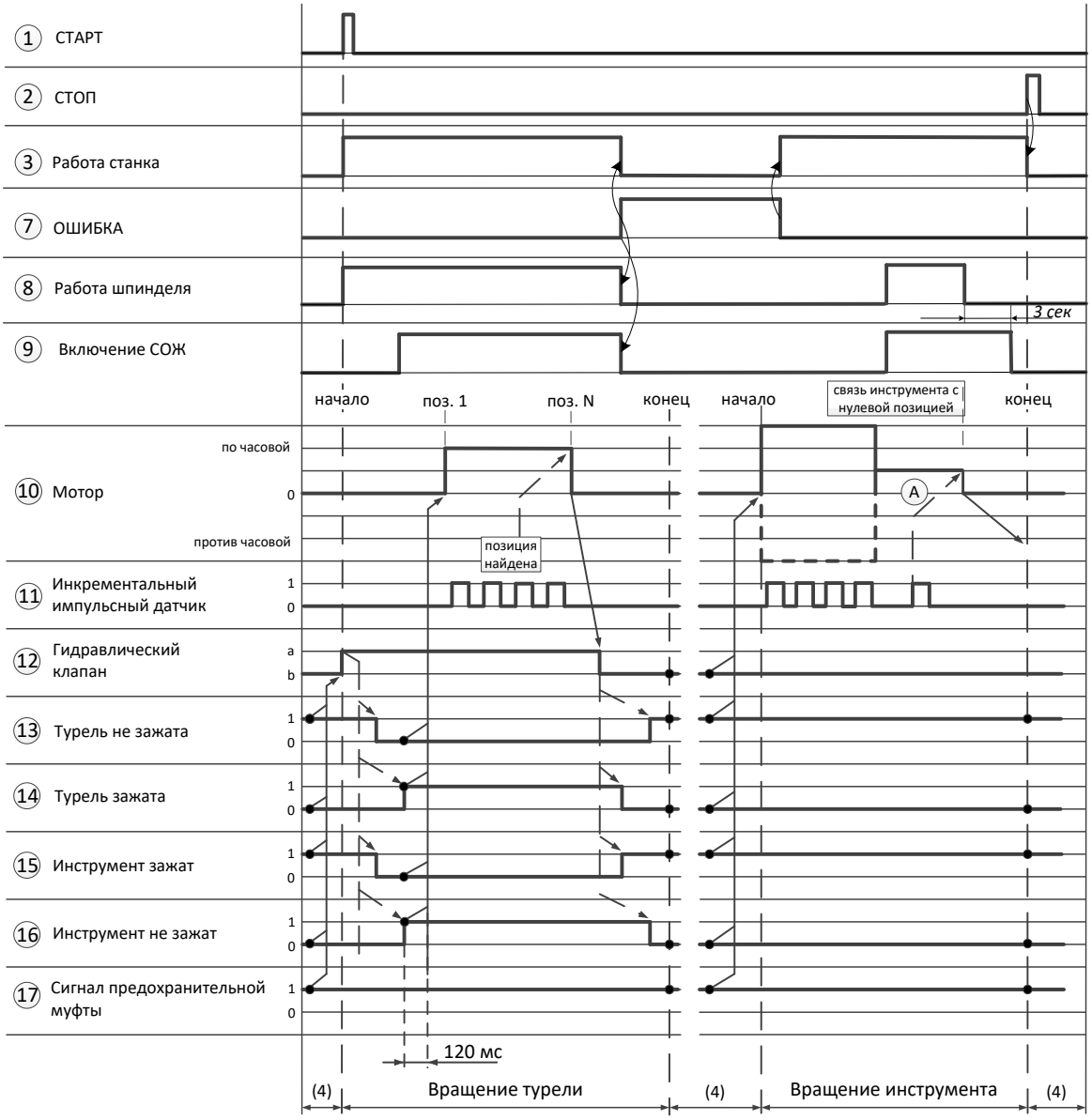


Рисунок 4.5 Циклограмма работы станка СА - 700

Программная реализация работы циклограммы, выполненная в среде разработки программ логического управления, и функциональный блок, её реализующий представлены на рисунке 4.6.

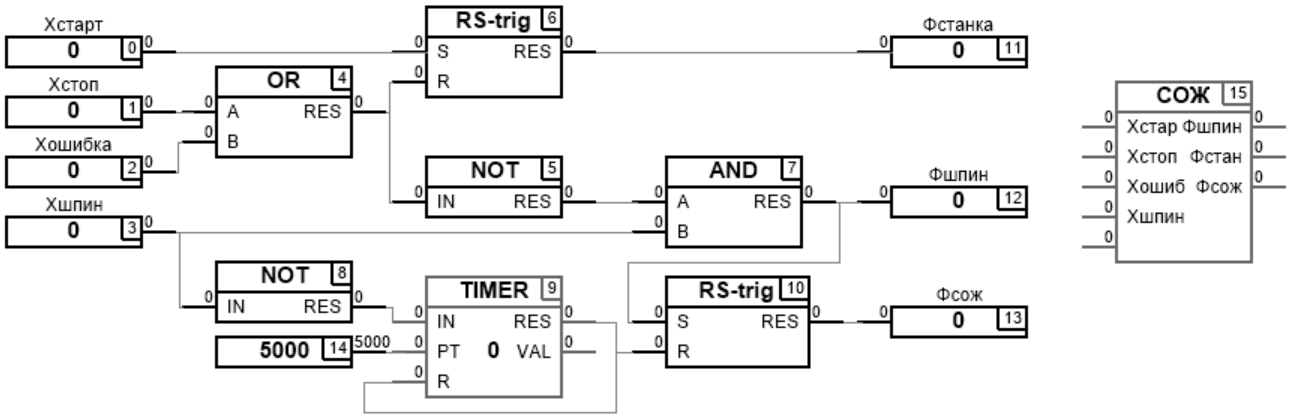


Рисунок 4.6 Логическая программа реализации включения СОЖ

Анализ представленной выше схемы и циклограммы целесообразно разделить на две составляющие: комбинационную и временную. К комбинационной составляющей будут относиться функции работы станка $\varphi_{СТАНКА}$ и шпинделя $\varphi_{ШПИНДЕЛЬ}$, т.к. их можно описать комбинационными схемами. Данное описание будет выглядеть в виде системы Булевых функций:

$$\begin{cases} \varphi_{СТАНКА} = (X_{СТАРТ} \vee \varphi_{СТАНКА})\overline{X}_{ОШИБКА}; \\ \varphi_{ШПИНДЕЛЬ} = X_{ШПИНДЕЛЬ}\overline{X}_{ОШИБКА}. \end{cases} \quad (4.6)$$

Для анализа временной составляющей системы целесообразно применять временные Булевы функции (ВБФ). В этом случае используется дискретное время с различным интервалом дискретизации – секунда, минута, час, год и т.д. Единицу дискретного времени обозначим целыми положительными числами – t (t = 0, 1, 2…*l*).

ВБФ при этом будет иметь вид:

$$y = f(x_1, x_2, ..., x_n, t) \quad (4.7)$$

где X_i - двоичные аргументы; t - аргумент, принимающий значения 0, 1, 2…*l*. Переменная t рассматривается как дискретное время, и означает моменты времени, в которые происходит смена режимов, переключение, переход в новое состояние и т. д. При фиксации времени t ВБФ вырождается в статическую булеву функцию. Придавая t ряд последовательных значений от 0 до *l*, можно получить последовательность булевых функций $y_0, y_1, \ldots, y_l$. Таким образом, каждой ВБФ соответствует последовательность обычных булевых функций. ВБФ можно описать с помощью таблицы истинности. Число строк в таблице должно быть $2^n (l + 1)$. Табличная запись наглядна, однако очень громоздка, поэтому для анализа используем другой способ. Введем специальную функцию, определенную соотношением:

$$t_\alpha = \begin{cases} 1, & t = \alpha; \\ 0, & t \neq \alpha. \end{cases} \quad (4.8)$$

Тогда ВБФ можно выразить следующим образом:

$$y = y_0 t_0 \vee y_1 t_1 \vee ... \vee y_l t_l \tag{4.9}$$

В любой фиксированный момент времени t = α (0 ≤ α ≤ *l*), $t_\alpha = 1$, все остальные $t_i=0$. Функции y_0, y_1, … y_l - обычные функции алгебры логики при t=0, t=1, …, t=*l*. При выражении работы системы СОЖ с применением ВБФ получим следующие функции:

В момент времени t_0 функция будет иметь вид

$$y_0 = \varphi_{ШПИНДЕЛЬ} \overline{X}_{ОШИБКА}; \tag{4.10}$$

В момент времени t_1 функция будет иметь постоянное значение 1.

Объединив данные функции, получим ВБФ работы системы СОЖ:

$$y = y_0 t_0 \vee y_1 t_1 = \varphi_{ШПИНДЕЛЬ} \overline{X}_{ОШИБКА} t_0 \vee t_1 \tag{4.11}$$

Указанная функция является периодической с периодом q=2, при этом будет иметь место соотношение $y_t = y_{t+q}$. Если объединить функцию с функциями, описывающими логику работы станка и шпинделя, то получим систему (формула 4.12), описывающую логику работы представленной циклограммы в части включения станка и подачи СОЖ.

$$\begin{cases} \varphi_{СТАНКА} = (X_{СТАРТ} \vee \varphi_{СТАНКА}) \overline{X}_{ОШИБКА}; \\ \varphi_{ШПИНДЕЛЬ} = X_{ШПИНДЕЛЬ} \overline{X}_{ОШИБКА}; \\ y = \varphi_{ШПИНДЕЛЬ} \overline{X}_{ОШИБКА} t_0 \vee t_1. \end{cases} \tag{4.12}$$

4.3.3 Программирование цикловой электроавтоматики с использованием математического аппарата автоматных моделей

При программировании логического управления часто удобно использовать формальное описание дискретного управляющего автомата, который в сочетании с технологическим объектом образует систему управления циклическим процессом. При проектировании дискретного управляющего автомата следует учитывать технологические параметры объекта и их изменения. К технологическим параметрам можно отнести: давление, температурные характеристики, уровень рабочих жидкостей, пространственное положение рабочих органов и др. Изменения технологических параметров непрерывны и принимают значения, принадлежащие некоторой области. На изменение параметров влияют исполнительные механизмы, к которым относятся: электро-, гидро- и пневмодвигатели, распределительные устройства и др. [140-141]

Контроль значений технологических параметров осуществляется с использованием датчиков. Функционирование технологического объекта представляется в виде последовательных

шагов, на каждом из которых можно определить закон изменения технологических параметров. Переход между шагами осуществляется посредством переработки: формируемой датчиками информации, сигналов от оператора, сигналов с таймеров и счетчиков. Последовательность шагов в литературе часто называют циклическим процессом, а отдельный шаг – технологической операцией. Технологическая операция осуществляет дискретное воздействие на объект управления («пуск-останов», «включение-выключение» и др.), которое осуществляется через: исполнительные механизмы, изменение уставок на таймерах и счетчиках и т.д. Последовательность технологических операций циклического процесса может быть линейной и с ветвлениями. Циклическим процесс называется в следствии того, что при любом варианте работы ТО всегда возвращается в начальное положение, т.е. последовательность операций состоит из конечного числа замкнутых циклов. В качестве примера построения дискретного управляющего объекта рассмотрим технологическую схему управления револьверной головкой для осевого инструмента, представленную на рисунке 4.7.

Револьверная головка состоит из следующих узлов: 1 – синхронный электродвигатель; 2 – датчик положения вала электродвигателя (инкрементальный энкодер); 3а, 3б – двухпозиционный гидрораспределитель; 4 – датчик нулевой позиции инструментальной головки; 5 – датчик «диск револьверной головки зафиксирован»; 6 – датчик «диск револьверной головки расфиксирован»; 7 – датчик «вращение инструмента заблокировано»; 8 – датчик «вращение инструмента разблокировано»; 9 – датчик срабатывания предохранительной муфты.

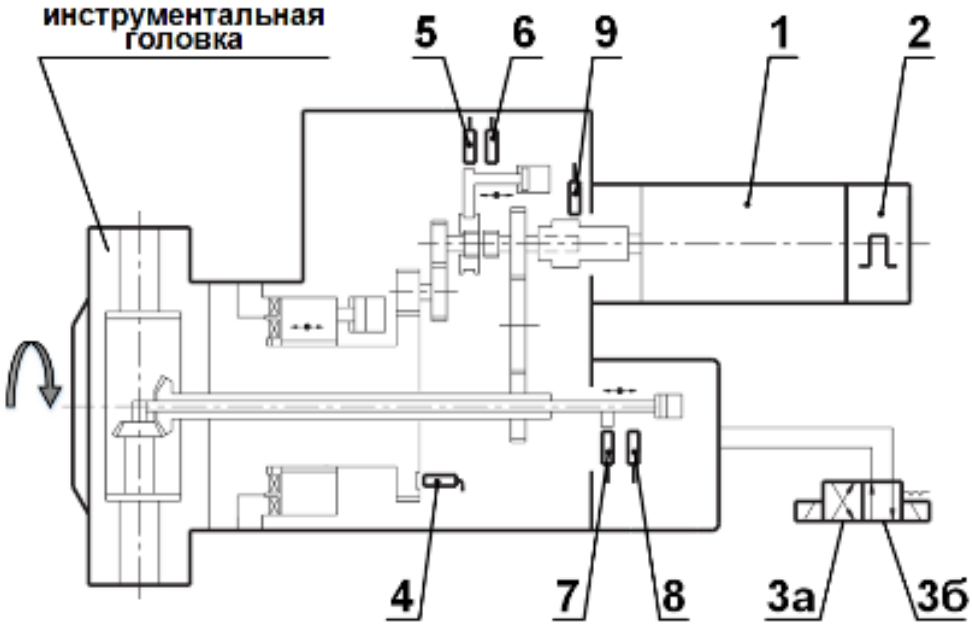


Рисунок 4.7 Схема работы револьверной головки для осевого инструмента

Работу револьверной головки описывают циклограммы, представленные на рисунке 4.8. Первая часть циклограммы отображает процесс вывода приводного инструмента в нулевую позицию, включающая в себя запуск электродвигателя 1 по часовой стрелке с последующим его остановом по сигналу от инкрементального энкодера. Процесс вывода инструментальной головки в нулевую позицию представлен на второй части циклограммы и состоит из следующей последовательности действий: перевод гидрораспределителя 3 в положение «а»; получение контрольных данных с датчиков 5,6,7,8 о том, что диск револьверной головки расфиксирован; после подтверждения расфиксации диска револьверной головки запуск электродвигателя 1 по часовой стрелке с последующей его остановкой по согласованному сигналу от датчика нулевой позиции револьверной головки 4 и инкрементального энкодера 2; после нахождения нулевой позиции револьверной головки перевод гидрораспределителя 3 в положение «б» и контроль перевода по датчикам 5,6,7,8.

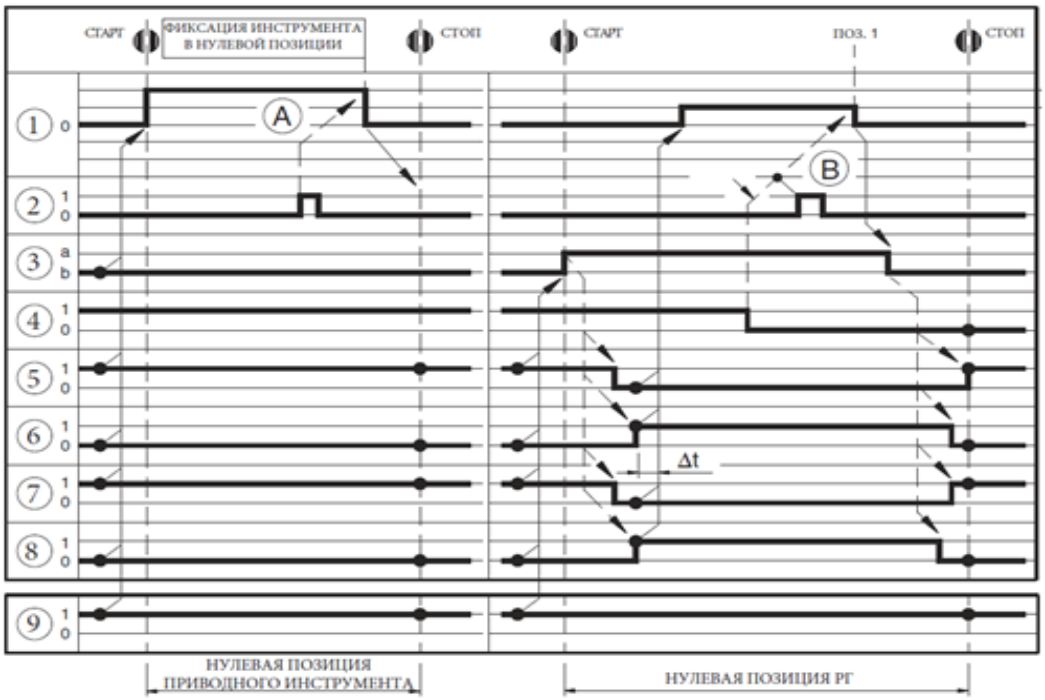


Рисунок 4.8 Циклограмма работы револьверной

Циклограмма выполнения рабочих операций револьверной головки состоит из двух частей. В первой части представлена последовательность рабочих операции при смене инструмента: перевод гидрораспределителя 3 в положение «а»; контроль перевода гидрорасрпделителя по датчикам 5,6,7,8; после временной задержки Δt = 120мс запуск вращения электродвигателя 1 по часовой стрелке на угол соответствующий позиции требуемого инструмента; после нахождения требуемого инструмента – останов электродвигателя 1; перевод гидрораспределителя 3 в положение «б» с последующим контролем перевода по датчикам 5,6,7,8.

Вторая часть отображает последовательность действий при работе револьверной головки в режиме вращения приводного инструмента: после контроля данных с датчиков 5,6,7,8 включается электродвигатель по/против часовой стрелки (в зависимости от заданной команды), с последующим остановом по окончании обработки и выходом инструмента в нулевую позицию (при необходимости). При работе системы управления револьверной головкой используются следующие информационные каналы, представленные в таблице 4-7.

Таблица 4-7 Информационные каналы работы револьверной головки

Обозначение	**Описание**
Выходные сигналы (сигналы управления)	
Z1	вращение электродвигателя на заданный угол
Z2.1	вращение электродвигателя по часовой стрелке
Z2.2	вращение электродвигателя против часовой стрелки
Z3	включить зажим диска
Z4	отключить зажим диска
Z5	остановить вращение электродвигателя
X100	ошибка
Обозначение	**Описание**
Входные данные с датчиков и элементов управления	
X1	срабатывание предохранительной муфты
X2.1	привод вращения инструмента отключен
X2.2	привод вращения инструмента включен
X3.1	диск револьверной головки зафиксирован
X3.2	диск револьверной головки расфиксирован
X4	нулевая позиция
α0	сигнал начала работы
Команды управления	
M03	вращение шпинделя по часовой стрелке
M04	вращение шпинделя против часовой стрелки
M05	остановка вращающегося шпинделя
M06	смена инструмента
K	вычислить угол поворота двигателя
Вспомогательные информационные каналы	
k	угол поворота двигателя вычислен
β06	подтверждение завершения смены инструмента
U1	воздействие на таймер
Y1	Сигнал истечения времени таймера

Технологический граф работы револьверной головки представлен на рисунке 4.9. Вершины технологического графа соответствуют следующим технологическим операциям: A_0 – все исполнительные механизмы револьверной головки находятся в исходном состоянии; A_1 – нахождение нулевой позиции инструментальной головки; A_2 – режим готовности; A_3 – работа револьверной головки в режиме вращения инструмента (шпиндельный режим); A_4 – режим смены инструмента.

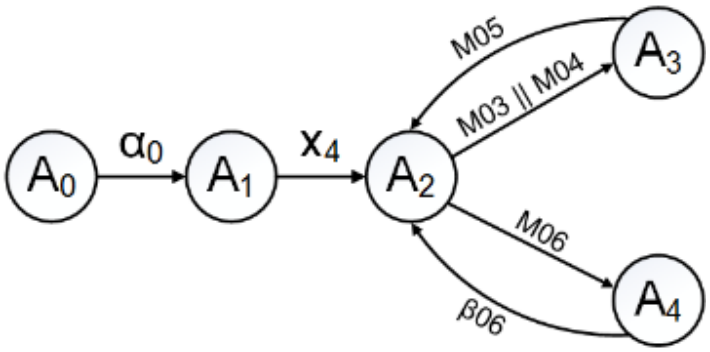


Рисунок 4.9 Технологический граф работы револьверной головки

Технологический граф работы револьверной головки дилогический. Переход из вершины A_0 в вершину A_1 отмечен блокировкой α_0, переход из вершины A_1 в вершину A_2 блокировкой X_4, переход из A_2 в вершины A_3 и A_4 по конъюнктивным ребрам отмечены блокировками M03 || M04 и M06, после завершения обоих операция по конъюнктивным ребрам в вершину A_2. Разложение технологических операций на элементарные операции показано на рисунке 4.10.

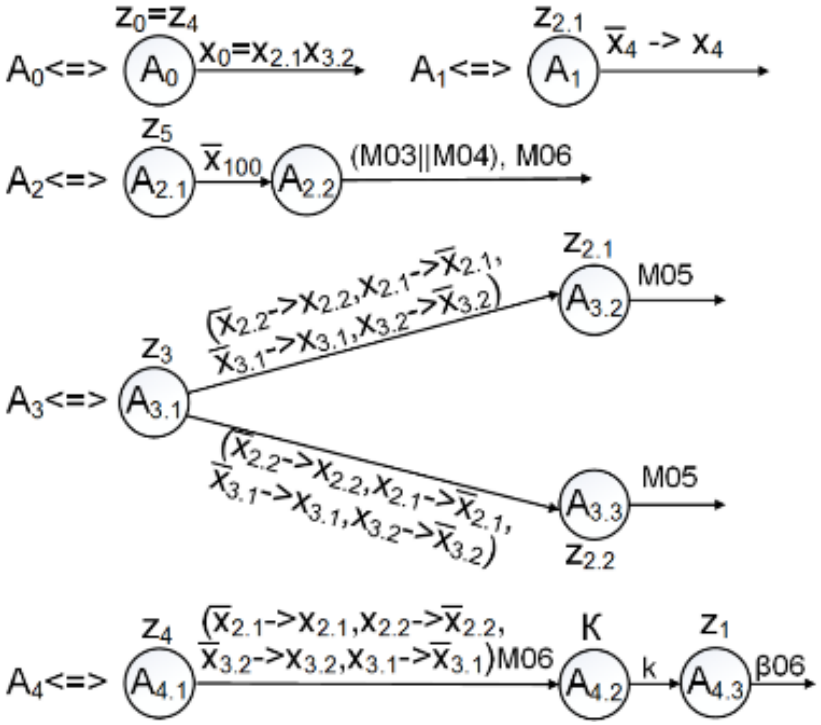


Рисунок 4.10 Разложение исходных операций револьверной головки

Операция A_0 является элементарной и характеризуется воздействием $Z_0 = Z_4$ и реакцией $X_0 = X_{2.1}X_{3.2}$. Операция A_1 является элементарной и характеризуется воздействием $Z_{2.1}$ и реакцией $\overline{x}_4 \rightarrow X_4$. Операция A_2 состоит из двух операций: $A_{2.1}$ – останов вращения электродвигателя с воздействием Z_5 и реакцией $\overline{x}_{100}$ и $A_{2.2}$ – режим готовности с воздействием M03 || M04 для вращения в режиме шпинделя или M06 для смены инструмента.

Операция A_3 состоит из трех элементарных операций и является дилогической. $A_{3.1}$ включает зажим диска револьверной головки воздействием Z_3 и характеризуется реакцией ($\overline{x}_{2.2}\rightarrow x_{2.2}$, $x_{2.1}\rightarrow \overline{x}_{2.1}$, $\overline{x}_{3.1}\rightarrow x_{3.1}$, $x_{3.2}\rightarrow \overline{x}_{3.2}$) M03 для вращения инструмента по часовой стрелке и ($\overline{x}_{2.2}\rightarrow x_{2.2}$, $x_{2.1}\rightarrow \overline{x}_{2.1}$, $\overline{x}_{3.1}\rightarrow x_{3.1}$, $x_{3.2}\rightarrow \overline{x}_{3.2}$) M04 для вращения инструмента против часовой стрелки. $A_{3.2}$ запускает вращение электродвигателя по часовой стрелке с реакцией на команду M05. $A_{3.3}$ запускает вращение электродвигателя против часовой стрелки с реакцией на команду M05.

Операция A_4 состоит из трех элементарных операций: $A_{4.1}$, с воздействием Z_4 отключающим зажим диска револьверной головки и характеризующимся реакцией $\overline{x}_{2.1}\rightarrow x_{2.1}$, $x_{2.2}\rightarrow\overline{x}_{2.2}$, $\overline{x}_{3.2}\rightarrow x_{3.2}$, $x_{3.1}\rightarrow \overline{x}_{3.1}$, $A_{4.2}$ с воздействием K (процедура вычисления угла поворота диска револьверной головки) и реакцией k, $A_{4.3}$ с воздействием Z_1 – поворотом электродвигателя на заданный угол и реакцией $\beta 06$.

Реакция на аварийную ситуацию будет представляться следующим соотношением:

$$\forall i\ [\mathrm{Ai} => X_{100}\ A_5],\ \text{где } i\neq 0. \qquad (4.13)$$

Операция A_5 (обработка ошибочных ситуаций) характеризуется воздействием Z_5, вызывая остановку электродвигателя. Переход из состояния A_5 возможен только в состояние A_0, по реакции $X_{100}\rightarrow\overline{x}_{100}$ (ошибка устранена). На графе операций представленном на рисунке 4.11 для описанного режима внесена дополнительная вершина, соединенная с вершиной A_0.

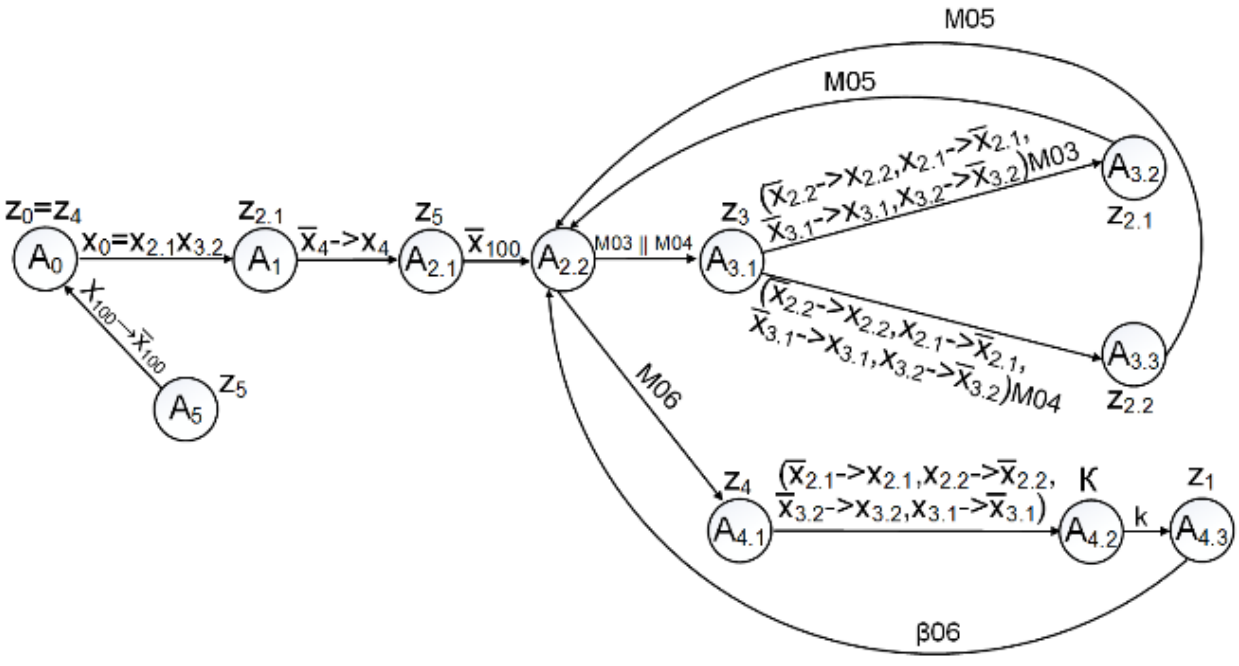


Рисунок 4.11 Граф операций работы револьверной головки

На основе спроектированного графа операций работы револьверной головки реализуется программа логического управления на языке функциональных блоков (рисунок 4.12).

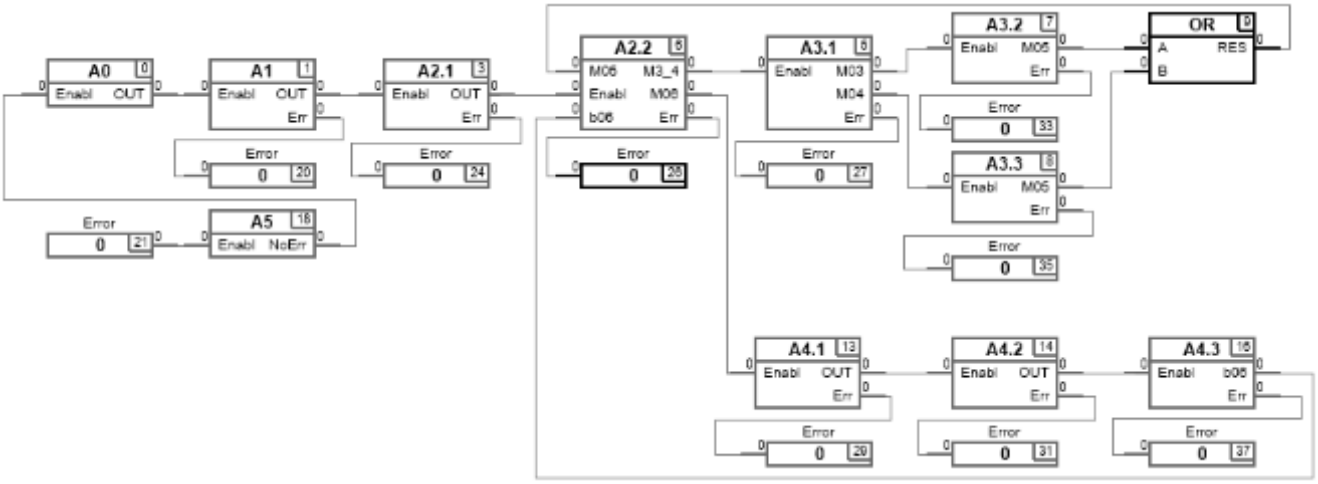


Рисунок 4.12 Программа управления электроавтоматикой револьверной головки на языке функциональных блоков

Программа логического управления полностью повторяет структуру графа операций, каждое из состояний графа операций (A_0, A_1, …) заменено на пользовательский функциональный блок (с именами A_0, A_1, …), в котором активируются воздействия на исполнительные механизмы револьверной головки и обрабатываются реакции с датчиков. В некоторых из реализованных пользовательских функциональных блоков активируются рабочие алгоритмы. Например, в функциональном блоке $A_{4.2}$ активируется алгоритм вычисления угла поворота электродвигателя для поиска требуемого инструмента.

Каждый из функциональных блоков имеет вход Enable, на который приходит разрешение на выполнение операций внутри блока, и выход OUT, с которого активируется сигнал на выполнение работы для следующего функционального блока. Все функциональные блоки за исключением A_0 и A_5 имеют выходы Error, на которые поступает сигнал о наличии ошибки при работе револьверной головки. Сигнал о наличии ошибки с выхода функционального блока записывается в разделяемую память, а из разделяемой памяти поступает на вход блока A_5. В блоке A_5 происходит останов электродвигателя и активируются механизмы сигнализации об ошибках.

4.3.4 Программирование дискретных систем с использованием математического аппарата разностных уравнений

При управлении электроавтоматикой технологических объектов возникает необходимость контроля изменений параметров объекта управления и формирования управляющего воздействия, основываясь на полученных данных. Задачи контроля параметров с последующим изменением управляющего воздействия целесообразно решать с применением регуляторов -

специализированных аппаратно-программных устройств. [142-144] В рамках решения логической задачи управления на базе классического ПЛК реализовать регулятор без применения специализированных аппаратных решений не представляется возможным. Например, для контроллеров производства ОВЕН, реализующих функции регулирования, необходимо дополнительно покупать аппаратный модуль ПИД регулирования ТРМ-210. [190] Приведенный ниже подход позволяет реализовать ПИД регулятор с использованием ограниченных ресурсов инструментальных средств программирования системы логического управления без привлечения специализированных дополнительных аппаратных модулей.

В качестве примера рассмотрим гамму токарно-фрезерных обрабатывающих центров высокой точности производства ОАО «Саста», в которых для достижения класса точности А применяется регулирование температуры отдельных узлов (станина, шпиндель, инструмент). Техническое задание требует разработать алгоритм термокомпенсационного регулирования станины токарно-фрезерного обрабатывающего центра высокой точности наклонной компоновки, который будет способен обеспечить поддержание постоянной температуры в контуре охлаждения станины величиной 22±0.5°C. [145]

Контур охлаждения представляет собой замкнутую систему, включающую в себя трубопровод из медного сплава протяженностью в 20000 мм и ⌀15 мм, охлаждение производится посредством рабочей жидкости, температурой 10°C, подаваемой в контур из станции охлаждения.

Регулирование расхода потока охлаждающей жидкости (воды) выполняется с помощью регулятора расхода прямого действия с электронным пропорциональным управлением RPCED1 фирмы Duplomatic Oleodinamica, который управляется электронным блоком управления для пропорциональных распределителей без обратной связи EDM-M111/20E0 фирмы Duplomatic Oleodinamica (рисунок 4.13).

Электронный блок представляет собой цифровой усилитель для управления пропорциональными распределителями без обратной связи. Усилитель подаёт ток, прямопропорциональный опорному сигналу и не зависящий от колебаний температуры или сопротивления нагрузки. Ступень широтноимпульсного модулятора позволяет снизить гистерезис клапана, тем самым улучшая точность управления. Показания температуры в контуре охлаждения снимаются и передаются в электронный блок управления датчиком температуры TT-103A-G1/8-L16-H114-L013 фирмы Turck, который имеет следующие характеристики: точность 0.1 % f.s, рабочий диапазон -50…+500 °C, возможность пользовательских настроек, выходной сигнал 4…20 мА (2-контакта), первоначальная настройка 0…150 °C, выходной переключатель PNP, класс защиты IP67.

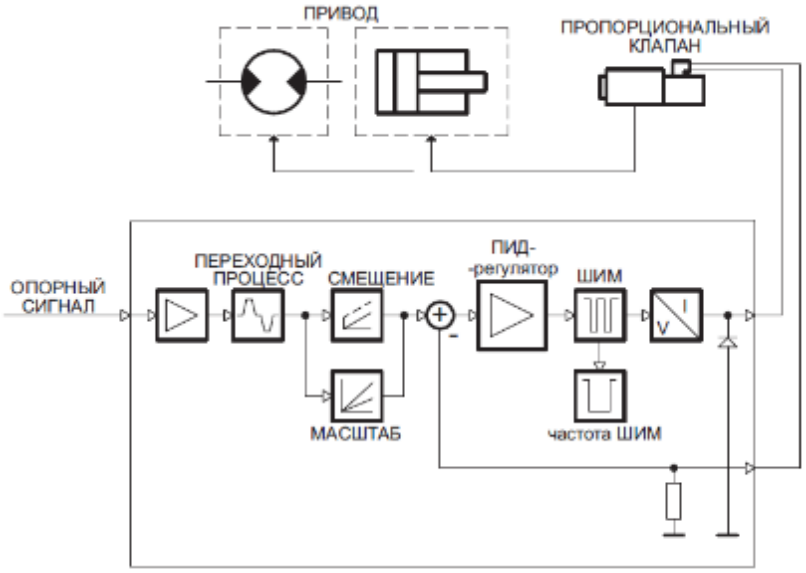


Рисунок 4.13 Схема работы электронного блока регулятора расхода

Получение передаточной функции объекта управления. При регулировании температуры станины обрабатывающих центров используется физический процесс конвективного теплообмена (теплопередача), при котором происходит перенос тепловой энергии между поверхностью твердого тела (станины) и теплоносителем (рабочая жидкость). В технологической системе станка процесс теплоотдачи происходит при движении рабочей жидкости под действием насоса станции охлаждения, в следствии этого происходит вынужденная конвекция.

Основным законом при конвективном теплообмене между рабочей жидкостью и контуром охлаждения станины является закон Ньютона-Рихмана в дифференциальном виде:

$$c_T \frac{dT}{dt} = \sum_{i=1}^{n} P_i - c_o\,(T - T_{oc}) \tag{4.14}$$

где c_T – теплоемкость охлаждаемого объекта;

$\frac{dT}{dt}$ - производная температуры объекта по времени;

$\sum_{i=1}^{n} P_i$ – сумма тепловых мощностей, действующих на охлаждаемый объект;

c_o - константа теплообмена (вычисляется экспериментальным путем, нахождением оптимального значения);

T – температура объекта управления;

T_{oc} - температура окружающей среды.

Преобразуем полученное дифференциальное уравнение:

$$c_T \frac{dT}{dt} + c_o(T - T_{oc}) = \sum_{i=1}^{n} P_i \tag{4.15}$$

$$\frac{c_T}{c_o}\frac{dT}{dt} + (T - T_{oc}) = c_0^{-1} \sum_{i=1}^{n} P_i \tag{4.16}$$

$T_{ob} = \frac{c_T}{c_o}(c)$ - динамическая характеристика времени реакции объекта

$T_{\text{пер}} = T - T_{\text{ос}}$ – температура перегрева;

$$p = \frac{d}{dt} \tag{4.17}$$

$$\frac{dT_{\text{пер}}}{dt} = \frac{dT}{dt},\ \text{т. к. } T_{\text{ос}} = const \tag{4.18}$$

$$T_{ob}\frac{dT_{\text{пер}}}{dt} + T_{\text{пер}} = \text{c}_0^{-1}\sum_{i=1}^{n} P_i \tag{4.19}$$

$$T_{ob}pT_{\text{пер}} + T_{\text{пер}} = c_o^{-1}\sum_{i=1}^{n} P_i \tag{4.20}$$

$$T_{\text{пер}}(T_{ob}p + 1) = c_0^{-1}\sum_{i=1}^{n} P_i \tag{4.21}$$

$$\frac{T_{\text{пер}}}{\sum_{i=1}^{n} P_i} = \frac{c_0^{-1}}{T_{ob}p+1} \tag{4.22}$$

где $\sum_{i=1}^{n} P_i$ – входное воздействие,

$T_{\text{пер}}$ – выходное воздействие. В ходе преобразований получаем передаточную функцию объекта управления:

$$W(p) = \frac{c_o^{-1}}{(1+pT_K)} = \frac{k_{ob}}{T_{ob}p+1} \tag{4.23}$$

Математическая модель объекта управления представляет собой апериодическое звено первого порядка, которое откликается на входное воздействие с задержкой, пропорциональной постоянной времени. В связи с тем, что зоны нагрева станины не локализованы и заранее не известны, то точку приложения, величину и мощность возмущающих воздействий оценить не представляется возможным. В следствии этого принцип управления по возмущению для решения задачи не применим. В этом случае мы будем осуществлять управление по отклонению.

При статическом положении станка (без движения осей) для термокомпенсации можно использовать пропорциональный регулятор, алгоритм работы которого будет поддерживать температуру в оптимальном диапазоне. При активной работе технологических узлов станка статическая ошибка регулирования будет прямо пропорциональна мощности нагрева, а значит в регуляторе целесообразно использовать интегральную составляющую.

По формуле (4.23) можно заметить, что динамическая характеристика времени реакции объекта (постоянная времени объекта управления как передаточного звена) велика из-за высокой теплоемкости и большой массы станины, в этом случае дифференциальную составляющую регулятора можно исключить. Поэтому для регулирования параметров термокомпенсации станины применяется ПИ регулятор, интегральная составляющая которого позволит избежать статической ошибки регулирования.

Математическая модель регулятора в дискретном виде. Рассмотрим передаточную функцию ПИД-регулятора:

$$W(p) = \text{Кп} + \frac{1}{T_и p} + T_д p \tag{4.24}$$

где Кп – коэффициент передачи пропорциональной части регулятора,

Ти - постоянная времени интегрирования,

Тд – постоянная времени дифференцирования.

Управляющий сигнал ПИД-регулятора (4.16) можно записать в виде дифференциального уравнения:

$$U(t) = \text{Кп}\varepsilon(t) + \frac{1}{T_и}\int_0^t \varepsilon(t)dt + T_д\frac{d\varepsilon}{dt} \quad (4.25)$$

где U(t) – выходная величина регулятора (управляющий сигнал),

$\varepsilon(t)$ - сигнал рассогласования (ошибка регулирования).

Преобразуем дифференциальное уравнение в разностное с помощью дискретизации функции, при условии, что квантование будет осуществляться с малой длительностью такта Т. Для этого заменим производную разностью первого порядка, а интеграла - интерполяционной формулой по методу прямоугольников:

$$U[n] = \text{Кп}\varepsilon[n] + \frac{T}{T_и}\sum_{i=1}^{n}\varepsilon[i] + \frac{T_д}{T}(\varepsilon[n] - \varepsilon[n-1]) \quad (4.26)$$

При программной реализации в логическом контроллере необходимо преобразовать формулу (4.18) с применением рекуррентных (разностных) алгоритмов, выразив значение управляющего сигнала U[n] через предыдущее значение U[n-1] и поправочный член ΔU[n]:

$$\Delta U[n] = U[n] - U[n-1] = q_0\,\varepsilon[n] + q_1\,\varepsilon[n-1] + q_2\,\varepsilon[n-2], \quad (4.27)$$

где согласно уравнению (4.26) имеем:

$$q_0 = \text{Кп} + \frac{T}{Tи} + \frac{T_д}{T}, \quad (4.28)$$

$$q_1 = -\text{Кп} - \frac{2T_д}{T}, \quad (4.29)$$

$$q_2 = \frac{T_д}{T} \quad (4.30)$$

Используя поправочный член ΔU[n] получим рекуррентное уравнение описания управляющего сигнала ПИД регулятора в дискретном виде.

Реализация регулятора на языке функциональных блоков. По полученному рекуррентному выражению, описывающему модель ПИД регулятора в дискретном виде, реализуется пользовательский функциональный блок на языке программирования FBD, с использованием ограниченных программных средств и ресурсов. В качестве входных параметров будут выступать общие параметры и параметры регулятора. К общим параметрам относятся: Тзад – оптимальная температура станины (22 градуса Цельсия), Ттек – текущая температура станины (с датчика установленного на станине), Т – время цикла работы ПЛК (10 мс). К параметрам регулятора относятся: Кп – коэффициент передачи пропорциональной части регулятора, Ти - постоянная времени интегрирования, Тд – постоянная времени дифференцирования. Для решения задачи терморегулирования дифференциальная составляющая исключается, в этом случае по-

стоянную времени дифференцирования принимаем равной 0 (Тд =0). В работе предложена последовательность действий, которая позволяет получить на языке FBD блок ПИД-регулятора (рисунок 4.14).

Шаг 1. Передаточная функция W(p) ПИД-регулятора записана в виде:

$\boldsymbol{W(p)} = \text{Кп} + \frac{\mathbf{1}}{\boldsymbol{T_и p}} + \boldsymbol{T_д p}$, где Кп – коэффициент передачи пропорциональной части, Ти, Тд - постоянная времени интегрирования и дифференцирования.

Шаг 2. Управляющий сигнал ПИД-регулятора представляется в виде дифференциального уравнения: $\boldsymbol{U(t)} = \text{Кп}\varepsilon(\boldsymbol{t}) + \frac{\mathbf{1}}{\boldsymbol{T_и}}\int_{\mathbf{0}}^{\boldsymbol{t}} \varepsilon(\boldsymbol{t})\boldsymbol{dt} + \boldsymbol{T_д}\frac{\boldsymbol{d\varepsilon}}{\boldsymbol{dt}}$, где U(t) – выходная величина регулятора (управляющий сигнал), $\varepsilon(\mathbf{t})$ - сигнал рассогласования (ошибка регулирования).

Шаг 3. Управляющий сигнал ПИД регулятора представляется в виде разностного уравнения: $\boldsymbol{U[n]} = \text{Кп}\varepsilon[\boldsymbol{n}] + \frac{\boldsymbol{T}}{\boldsymbol{T_и}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \varepsilon[\boldsymbol{i}] + \frac{\boldsymbol{T_д}}{\boldsymbol{T}}(\varepsilon[\boldsymbol{n}] - \varepsilon[\boldsymbol{n-1}])$

Шаг 4. Управляющий сигнал ПИД регулятора записывается в рекуррентном виде: ΔU[n] = U[n] - U[n-1] = q0 ε[n] + q1 ε[n-1] + q2 ε[n-2], где $q0 = \text{Кп} + \frac{\boldsymbol{T}}{\boldsymbol{T_и}} + \frac{\boldsymbol{T_д}}{\boldsymbol{T}}$, $q1 = -\text{Кп} - \frac{\boldsymbol{2T_д}}{\boldsymbol{T}}$, $q2 = \frac{\boldsymbol{T_д}}{\boldsymbol{T}}$ - коэффициенты поправочного члена.

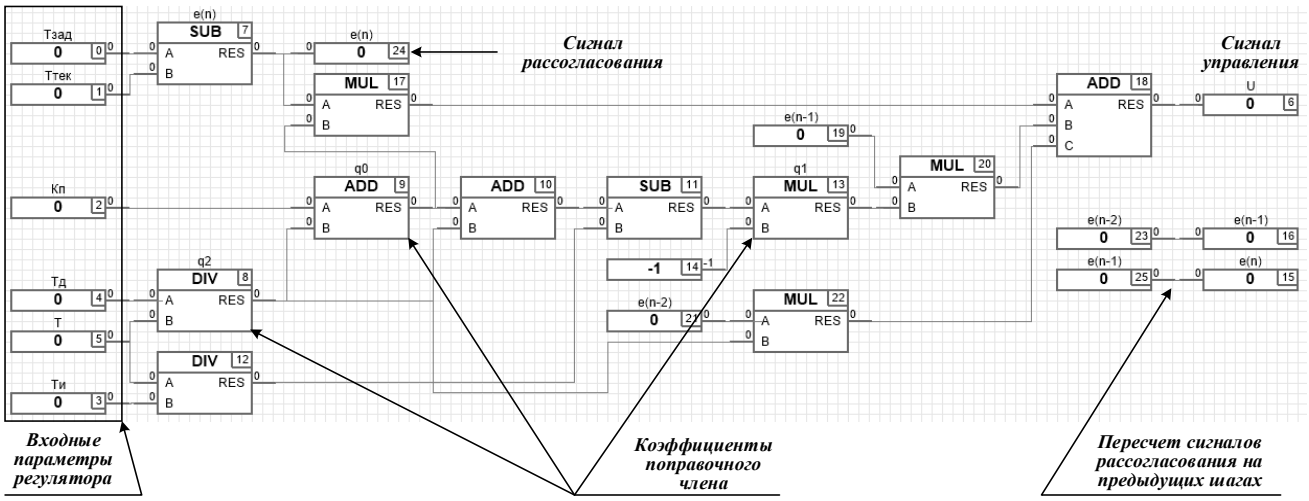


Рисунок 4.14 Реализация ПИД регулятора на языке функциональных блоков

4.4 Разработка методики тестирования систем логического управления технологическим оборудованием

Существует большое количество методик тестирования как программного, так и аппаратного обеспечения, среди которых можно выделить: функциональное, структурное, регрессионное, нагрузочное, тестирование наработки на отказ, приемо-сдаточные испытания и др. Комплексное тестирование систем автоматизации включает в себя: нагрузочное тестирование,

тестирование на отказ и проведение приемо-сдаточных испытаний. Остальные виды тестирования проводятся для отдельных модулей программного обеспечения системы логического управления на этапе его разработки и для инженеров-проектировщиков систем управления не актуальны. [148-151]

4.4.1 Разработка методики нагрузочного тестирования ядра системы логического управления

Под нагрузочным тестированием (англ. load testing) будем понимать тестирование производительности системы управления, при котором производится сбор показателей, определение времени отклика системы и производительности в ответ на внешнее воздействие. Целью нагрузочного тестирования является проверка соответствия системы предъявляемым требованиям. Нагрузочное тестирование позволяет получить интегральную оценку качества работы системы и оценить скрытые дефекты. Цель нагрузочного тестирования – наблюдение за показателями производительности системы при заданной нагрузке. Обычно нагрузочное тестирования производится для многопользовательских систем с целью определения количества одновременно обслуживаемых клиентов без потери качества. В случае системы логического управления целесообразно оценивать зависимость используемых вычислительных ресурсов от сложности системы. Сложность системы может характеризовать несколько показателей, среди которых: количество аппаратных входов/выходов, количество используемых функциональных блоков в программе логического управления. Для расчета каждого блока в цикле управления расходуется часть вычислительных ресурсов, с ростом числа функциональных блоков увеличивается сложность программы и растет расход вычислительных ресурсов, поэтому для проведения нагрузочного тестирования системы логического управления целесообразно использовать указанный показатель в качестве базового. Нагрузочное тестирование необходимо для формирования требований к производительности вычислительного ядра на стадии разработки функциональных требований. Зачастую четко сформулировать требования до начала разработки системы управления не представляется возможным. В этом случае необходимо проанализировать сложность разрабатываемой системы и вынести предположение об ожидаемой нагрузке и потреблении аппаратных ресурсов, на основе которых и проводить пробное нагрузочное тестирование (англ. exploratory load testing).

В случае с системой логического управления нагрузочному тестированию подвергалось ядро, как приложение наиболее критичное к имеющимся вычислительным ресурсам. Для проведения нагрузочного тестирования выбраны следующие показатели производительности приложения:

- Потребление ресурсов центрального процессора (оценивается в процентах). Указывает относительное время, затраченное процессором на вычисления для нужд выбранного процесса. На сегодняшний момент процессы могут функционировать в нескольких потоках, это позволяет процессору производить параллельные вычисления. Полученные метрики потреблений ресурсов процессора позволяют проанализировать: влияние на производительность системы количества потоков, конфигурации приложения, типа операционной системы, ее настроек и др.
- Потребление оперативной памяти (оценивается в Мб). Показатель оценивает абсолютное значение расходуемой приложением оперативной памяти. Мониторинг указанного показателя позволяет оптимизировать используемую приложением память. Во время работы приложения в памяти находятся ссылки на объекты, неиспользуемые ссылки очищаются процессом, который называется “сборщик мусора” (англ. Gargabe Collector, GC), на который также тратятся ресурсы процессора. При этом, если приложению выделены большие объемы памяти, то время, затрачиваемое процессором, может быть значительным. Во время очистки страниц памяти, доступ процессов основного приложения к ним заблокирован, что влияет на конечное время работы приложения. GC не применяется в ядре системы управления.
- Время цикла программы логического управления (оценивается в мс). Является наиболее критичным показателем производительности системы логического управления, т.к. напрямую характеризует процессы реального времени.

Стоит отметить, что аналогичные исследования представлены в диссертационной работе «Модели и алгоритмы синтеза кроссплатформенного контроллера автоматизации технологических процессов» Ковалевым И. А. [84] и касались вопросов тестирования на конечных этапах разработки, проводимых при синтезе кроссплатформенных программно-реализованных контроллеров. Для ядра системы управления, рассматриваемой в диссертации, проводилось тестирование на стадии ранней разработки с целью определения верности выбранных архитектурных решений при проектировании. Такое тестирование называется тестирование проверки концептуальных решений (англ. proof-of-concept).

Статистические данные, получаемые при тестировании, имеют вид таблиц с числовыми данными большого объема, которые тяжело поддаются анализу. При этом свойства полученных данных зачастую тяжело выявить, т.к. представление информации в виде больших массивов данных не наглядно и не дает визуальной характеристики о наличии закономерностей. В работе

для анализа данных нагрузочных тестов использовался инструментарий гистограмм, который наиболее часто применяется в статистических методах анализа.

Потребление ресурсов центрального процессора. Рассмотрим тестовый сценарий тестирования потребления ресурсов центрального процессора. Для проведения тестирования был разработан ряд программ логического управления, имеющих различное количество используемых функциональных блоков: 100, 850, 1500, 3500, 5000, 7500, 10000, 12500, 15000, 17500 и 20000 блоков. Размер программы логического управления в 20000 функциональных блоков, выбранный в качестве максимального, практически не достижим при программировании реальных объектов. Результаты тестирования приведены на рисунке 4.15.

При тестировании использовались четыре вычислительные платформы: персональный компьютер с процессором Core i7 частотой 1,8 ГГц (выделен оранжевым на гистограмме), ноутбук с процессором Pentium Dual 2,2 ГГц (выделен синим на гистограмме), одноплатный компьютер Orange Pi (выделен серым на гистограмме), одноплатный компьютер Odroid XU4 (выделен желтым на гистограмме). Персональный компьютер соответствует требованиям, предъявляемым к вычислительным узлам автоматизации технологического оборудования. Ноутбук позволяет оценить зависимость производительности вычислительной платформы от форм-фактора. Два одноплатных компьютера выбраны для того, чтобы оценить нагрузочную способность мобильных платформ различной производительности.

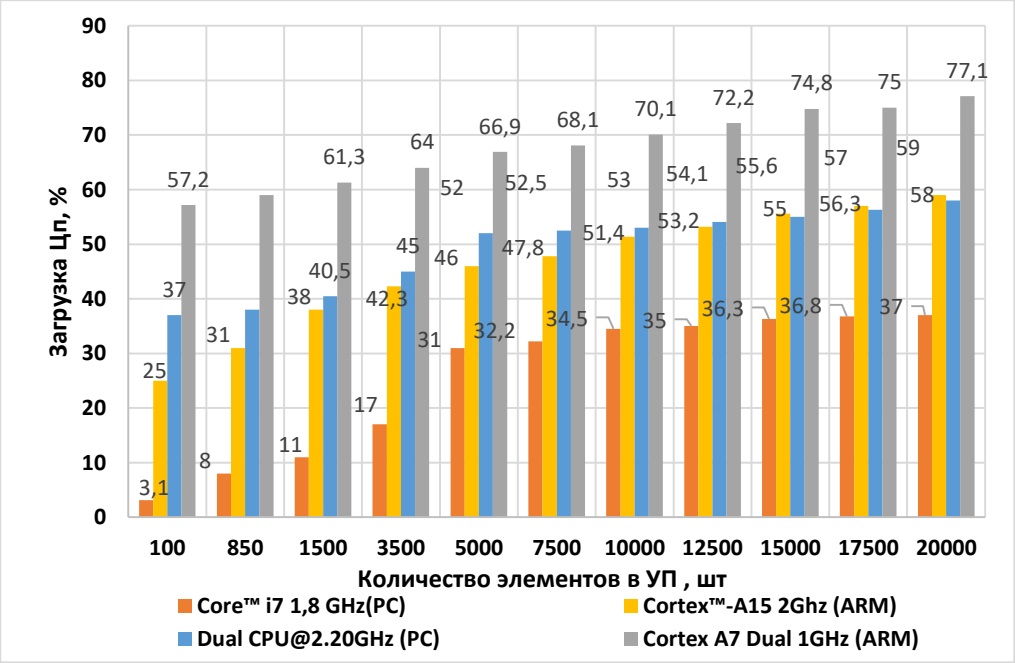


Рисунок 4.15 Зависимость потребления ресурсов центрального процессора от сложности программы

Анализ результатов нагрузочного тестирования показывает, что на всех вычислительных платформах с ростом сложности программы логического управления наблюдается экспоненциальный рост потребления ресурсов процессора. На наименее производительном одноплатном компьютере с ARM архитектурой при программе в 20 000 функциональных блоков процессор был загружен на 77%, что позволяет делать выводы о том, что все четыре выбранные для тестирования платформы могут применяться для реализации систем логического управления. Однако решение о том, какую вычислительную платформу целесообразнее применять, должно приниматься в зависимости от сложности системы. Нагрузочное тестирование также показало, что тактовая частота процессора напрямую не влияет на размер потребляемых ресурсов процессора. Это говорит о том, что применяемые в системе аппаратные вычислительные устройства должны быть максимально сбалансированы с точки зрения всех характеристик производительности. Оценивая относительный рост количества потребляемых ресурсов процессора на каждой из вычислительных платформ можно сделать вывод о том, что в среднем рост составил 25-30%. Из этого можно сделать вывод, что наиболее существенным фактором, влияющим на потребляемые ресурсы процессора, является не только сложность алгоритмов программы логического управления, но и количество ресурсов, привлекаемых самим приложением ядра для обслуживания внутренних потребностей.

Потребление оперативной памяти. Ядро системы логического управления может работать под управлением как 32-х битной, так и 64-х битных операционных систем. При этом 64 - битные версия операционной системы использует много больше памяти, в связи с тем, что переменные, которыми она оперирует не 32-х, а 64-х битные. При проектировании системы важно оценить не приводит ли использование 64-х битной версии операционной системы к критическому росту объема занимаемой оперативной памяти. Потребление оперативной памяти оценивалось для двух одинаковых вычислительных платформ (ПК с процессором Core i7 частотой 1,8 ГГц), но под управлением операционных систем с 32-х и 64-х битных (рисунок 4.16).

Результаты нагрузочных тестов показали, что зависимость объема занимаемой оперативной памяти от сложности программы логического управления линейная. В 64-х битной версии операционной системы (выделен оранжевым на гистограмме) объем занимаемой оперативной памяти ядром логического управления выше, чем в 32-х битной на 10-35% в зависимости от сложности программы логического управления.

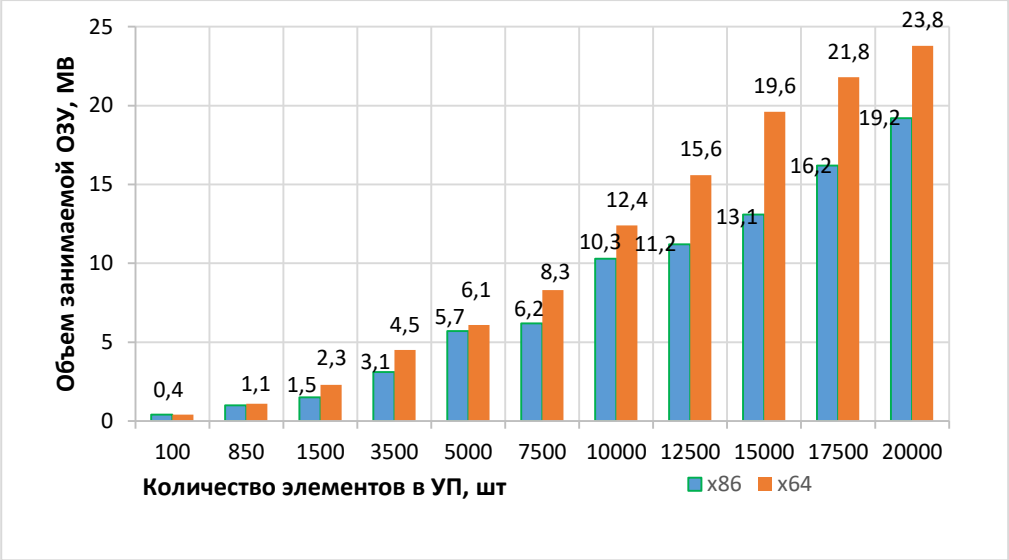


Рисунок 4.16 Результаты тестирования потребления оперативной памяти

Время цикла программы логического управления. Для оценки параметров надежности системы необходимо определить степень влияния сложности программы логического управления на время цикла управления, для этого были проведены тесты с использованием программ логического управления различной сложности (от 100 до 20000 элементов). Результаты тестовых испытаний представлены на рисунке 4.17. Тестирование проводилось с целью оценки влияния типа операционной системы на время цикла логического управления, поэтому тесты проводились под операционными системами, использующими машинное время (Windows XP - синий на гистограмме, Ubuntu 11.04 - оранжевый на гистограмме) и реальное время (Linux RT - серый на гистограмме).

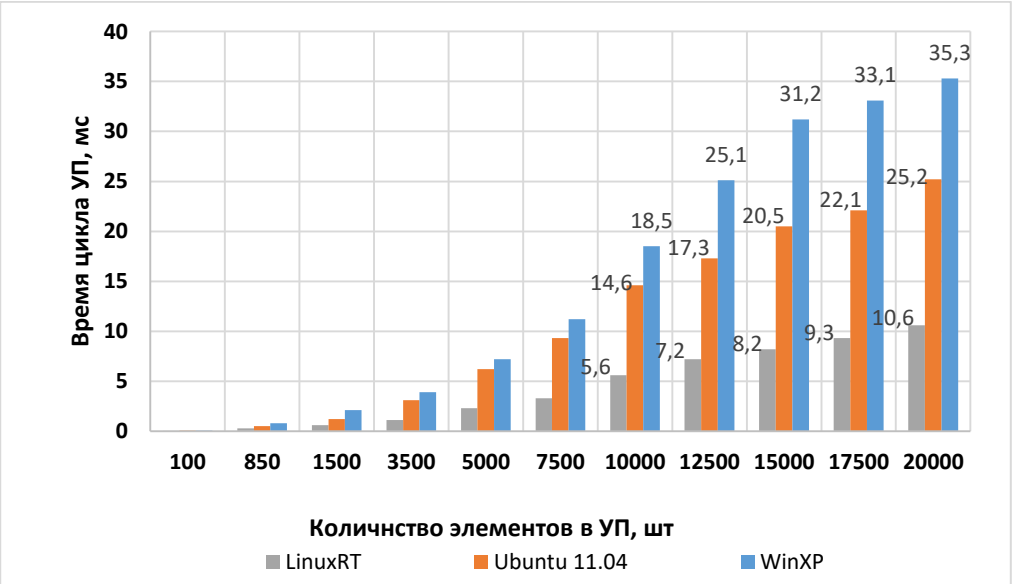


Рисунок 4.17 Зависимость времени цикла управления от сложности программы

Время цикла логического управления с увеличением сложности программы меняется линейно. Работа ядра логического управления под операционной системой реального времени показала, что для гарантированной работы в режиме жесткого реального времени программа логического управления не должна превышать 20000 функциональных блоков. Операционные системы семействаWindows и Ubuntu могут использоваться лишь для отладки программ логического управления или для учебных целей, т.к. не удовлетворяют условиям функционирования системы в режиме реального времени.

4.4.2 Разработка программы и методики испытаний системы логического управления

Готовая к эксплуатации система логического управления должна поставляться с комплектом документации, в который также входит “Программа и методика испытаний системы логического управления”, разработанная в соответствии с ГОСТ 16504-81 “Система государственных испытаний продукции. Испытания и контроль качества продукции. Основные термины и определения.” и ГОСТ 34.603-92 " Виды испытаний автоматизированных систем". [148-149] Документ содержит общие требования к проведению трех групп испытаний: “приемочные, аттестационные, инспекционные испытания”, “приемо-сдаточные испытания”, “периодические испытания”. Область применения и используемые при проектировании аппаратные и инструментальные средства накладывают свои ограничения и определяют специфику применения системы логического управления. Исходя из индивидуальных особенностей системы управления определяются частные требования к проведению тестовых испытаний. Далее приведен примерный порядок проведения тестовых испытании систем логического управления, разработанных исходя из особенностей их реализации и применения.

Перед проведением испытаний составляется таблица (таблица 4-8), в которой отражены виды тестовых кейсов и состояния системы управления, в которых они проводятся. В рамках тестирования проводятся: статические испытания (без включения питания), испытания без подключения объекта управления, испытания с подключенным объектом управлвнеия. Привести полный текст методики испытаний в диссертации не представляется возможным в связи с большим объемом. Далее рассматриваются основные аспекты проведения проверок и выборочно приведены некоторые тестовые кейсы.

Таблица 4-8 Виды испытаний и проверок системы логического управления

№	Наименование проверки	Приемочные, аттестационные, инспекционные испытания	Приемо-сдаточные испытания	Периодические испытания
1	**Испытания системы в статическом состоянии**			
1.1	Основные параметры			
1.1.1	Количество и тип аппаратных модулей ввода/вывода	*		*
1.1.2	Технические характеристики аппаратных вычислительных устройств	*		*
1.1.3	Количество и тип датчиков	*		*
1.1.4	Количество и тип панелей	*		*
1.1.5	Количество и тип сетевого оборудования	*		*
1.1.6	Суммарная мощность установленного электрооборудвоания	*		
1.2	Правильность подключения системы к сети и надежного заземления	*	*	*
1.3	Сопротивление изоляции электрооборудования	*	*	*
1.4	Электрическая прочность изоляции	*	*	*
1.5	Подключение системы к электросети	*	*	*
2	**Испытания системы без подключения объекта управления**			
2.1	Проверка правильности функционирования аппаратных входов	*	*	*
2.2	Проверка правильности функционирования аппаратных выходов	*	*	*
2.3	Проверка алгоритмов работы системы	*	*	*
2.4	Проверка правильности функционирования панелей оператора	*	*	*
2.5	Проверка показателей надежности			
2.6.1	Установленная безотказная наработка в сутки	*	*	*
2.6.2	Установленная безотказная наработка в неделю	*		
2.6.3	Установленная безотказная наработка	*		
3	**Испытания системы с подключенным объектом управления**			
3.1	Проверка датчиков и исполнительных устройств объекта управления	*	*	*
3.2	Проверка алгоритмов работы системы	*	*	*
3.3	Проверка правильности функционирования панелей оператора	*	*	*

Испытания системы в статическом состоянии.

Проверка количества и типа аппаратных модулей ввода/вывода. Метод: систематизация аппаратных модулей ввода/вывода по типу и подсчет их количества. *Условия приемки:* количество и тип модулей ввода/вывода должно совпадать с документацией.

Суммарная мощность установленного электрооборудования. Метод: суммарная мощность определяется арифметическим сложением мощностей всех потребителей, установленных на объекте управления. *Условия приемки:* суммарная мощность электрооборудования должна быть не более указанной в паспортных данных проверяемого объекта управления.

Правильность подключения системы к сети и наличие заземления. Метод: проверить на соответствие технической документации и измерить электрическое сопротивление. *Условия приемки:* наличие винтов заземления, табличек, надежность контактных соединений защитных цепей, отсутствие между головками винтов заземления и заземляющими частями электроизолирующих материалов или коррозии, наличие антикоррозионного покрытия или смазки, надежное крепление винтов заземления. Электрическое сопротивление между винтом заземления и металлическими частями объекта управления должно быть не более 0,1 Ом.

Сопротивление изоляции электрооборудования. Метод: измерить электроизмерительными приборами. Объект управления должен быть отключен от электросети и электронного оборудования. *Условия приемки:* сопротивление изоляции должно быть не менее 1 мОм.

Подключение системы к электросети. Метод: Проверить правильность заземления и подсоединения системы управления, правильность чередования фаз A, B, C на вводном выключателе (при наличии трёхфазного подключения). *Условия приемки:* должно соответствовать технической документации.

Испытания системы без подключения объекта управления

Проверка правильности функционирования аппаратных входов. Метод: подключение среды разработки программ к ядру и выход в отладочный режим. Подача на аппаратные входы электрических сигналов допустимого диапазона (верхний и нижний пороги). *Условия приемки:* совпадение заданных значений электрических сигналов и данных о сигнале в среде разработки программ логического управления.

Проверка правильности функционирования аппаратных выходов. Метод: подключение среды разработки программ к ядру и выход в отладочный режим. Последовательная активация в среде программирования функциональных блоков, привязанных к аппаратным выходам. Подключение мультиметра к проверяемому аппаратному выходу. *Условия приемки:* совпадение значения электрического сигнала, заданного в среде разработки и показаний мультиметра.

Проверка алгоритмов работы системы. Метод: подключение среды разработки программ к ядру и выход в отладочный режим. Пуск программы логического управления. Последовательный запуск пользовательских функциональных блоков, отвечающих за работу отдельных узлов объекта управления. *Условия приемки:* соответствие работы системы техническому заданию.

Проверка правильности функционирования панелей оператора. Метод: подключение панелей оператора к ядру системы управления. Активация функционального блока, отвечающего за формирование экрана оператора. Изменение условий, влияющих на формирование экрана оператора. *Условия приемки:* соответствие сформированных экранов оператора техническому заданию.

Проверка показателей надежности. Проверка производится в соответствии с ГОСТ 27.410-87 «Надежность в технике. Методы контроля показателей надежности и планы контрольных испытаний на надежность» [152].

Установленная безотказная наработка в сутки. Метод: установленная безотказная наработка оценивается по результатам подконтрольной эксплуатации не менее 3-х систем управления в реальных условиях. Исчисляется с момента ввода в эксплуатацию или после восстановления исправного состояния как сумма суточных наработок. При этом безотказная наработка в сутки должна быть не менее 21 часа. *Условия приемки:* наработка на отказ 21 час в сутки.

Установленная безотказная наработка в неделю. Метод: испытания проводятся в реальных условиях в три смены в течение недели. Безотказная наработка в неделю исчисляется как сумма суточных наработок с момента начала испытаний. *Условия приемки:* безотказная наработка в неделю должна быть не менее 126.

Установленная безотказная наработка. Метод. Установленная безотказная наработка оценивается по результатам подконтрольной эксплуатации не менее 3-х систем в реальных условиях. Исчисляется с момента ввода в эксплуатацию или после восстановления исправного состояния как сумма суточных наработок. *Условия приемки.* безотказная наработка должна быть не менее 1500 ч.

4.4.3 Разработка методики расчета средней наработки на отказ

В качестве одного из критериев надежности систем используют технический параметр, называемый наработка на отказ, который характеризует среднюю продолжительность работы системы между ремонтами и отражает время средней работы, приходящейся на один отказ (вы-

ражается в часах). В работе проведена методика, разработанная на основе рекомендаций, изложенных в ГОСТ 27.301-95. «Надежность в технике. Расчет надежности. Основные положения». [152-154] Для ядра системы логического управления под наработкой на отказ будем понимать срок до возникновения необходимости полного перезапуска системы. В качестве критерия, значение которого оценивается и по которому принимается решение о перезагрузке системы, примем время цикла работы ядра логического управления. При отказе системы (возникновение критической ошибка) или увеличении времени цикла более допустимого значения (10 мс для жесткого реального времени и 100 мс для мягкого реального времени) необходимо производить перезагрузку системы. Расчет показателей надежности проводится для конкретной выборки систем логического управления. Начальными данными для проведения расчета показателей надежности являются значения наработок исследуемой выборки.

Для расчета вероятности безотказной работы используют формулу:

$$P_k = (1 - \frac{r_1}{s_1})(1 - \frac{r_2}{s_2})(1 - \frac{r_k}{s_k}) \tag{4.1}$$

Где P_k - оценка вероятности безотказной работы в момент времени t_k, r_i -число отказов, S_i - число объектов в изучаемой выборке оборудования, наработка которых более t_{k-1}. P_k вычисляется для каждого из интервалов t_{i-1} до ti.

Зависимость вероятности безотказной работы от времени определяется следующей функцией:

$$P(t) = \exp(-a_1 t - a_2 t^2) \tag{4.2}$$

Где a_1 и a_2 - постоянные коэффициенты. Функция позволяет сделать прогноз зависимости вероятности безотказной работы от времени за пределами времени, изученного в ходе экспериментов. Средняя наработка до отказа определяется по:

$$T_{cp.} = \int_0^\infty P(t)dt = \int_0^\infty \exp(-a_1 t - a_2 t^2)dt \tag{4.3}$$

Геометрическим смыслом Тср является площадь под графиком зависимости безотказной работы. Далее рассмотрим пример расчета средней наработки системы логического управления на отказ. Для этого проводились тестирования на пяти однотипных прототипах. Для них имеется выборка наработок, завершившихся отказом. Временные отрезки выборки разделены следующим образом: 1000, 1200, 1500, 1700 и 2000 часов. Согласно формуле (4.1) проведем расчет вероятности безотказной работы для каждой из наработок полученных в ходе тестовых испытаний:

$$P_0 = (1 - \frac{0}{5}) = 1 \tag{4.4}$$

$$P_{1000} = (1-\frac{0}{5})(1-\frac{0}{5}) = 1 \quad (4.5)$$

$$P_{1200} = (1-\frac{0}{5})(1-\frac{0}{5})(1-\frac{0}{5}) = 1 \quad (4.6)$$

$$P_{1500} = (1-\frac{0}{5})(1-\frac{0}{5})(1-\frac{1}{5})(1-\frac{0}{5}) = 0{,}8 \quad (4.7)$$

$$P_{1700} = (1-\frac{0}{5})(1-\frac{0}{5})(1-\frac{1}{5})(1-\frac{0}{5})(1-\frac{1}{5}) = 0{,}64 \quad (4.8)$$

$$P_{2000} = (1-\frac{0}{5})(1-\frac{0}{5})(1-\frac{1}{5})(1-\frac{0}{5})(1-\frac{1}{5})(1-\frac{1}{5}) = 0{,}512 \quad (4.9)$$

На рисунке 4.18 представлен теоретический график экспоненциальной функции, построенный по формуле (4.2), на графике оранжевым выделены точки, полученные в ходе тестирования. Результаты, полученные в ходе экспериментальных исследований, близки к теоретическим.

Расчет средней наработки на отказ показал, что безотказная работы ядра логического управления превышает 1000 часов, что соответствует требованиям к системе. Вероятность наступления отказа после непрерывной работы более 1500 часов существенно возрастает. Основными причинами отказов стали «зависания» ядра логического управления, что связано с увеличением потребления вычислительных ресурсов с ростом времени наработки. В качестве профилактической меры при работе с ядром системы логического управления для устранения вероятных отказов стоит перезагружать систему не реже одного раза в 1200 часов работы.

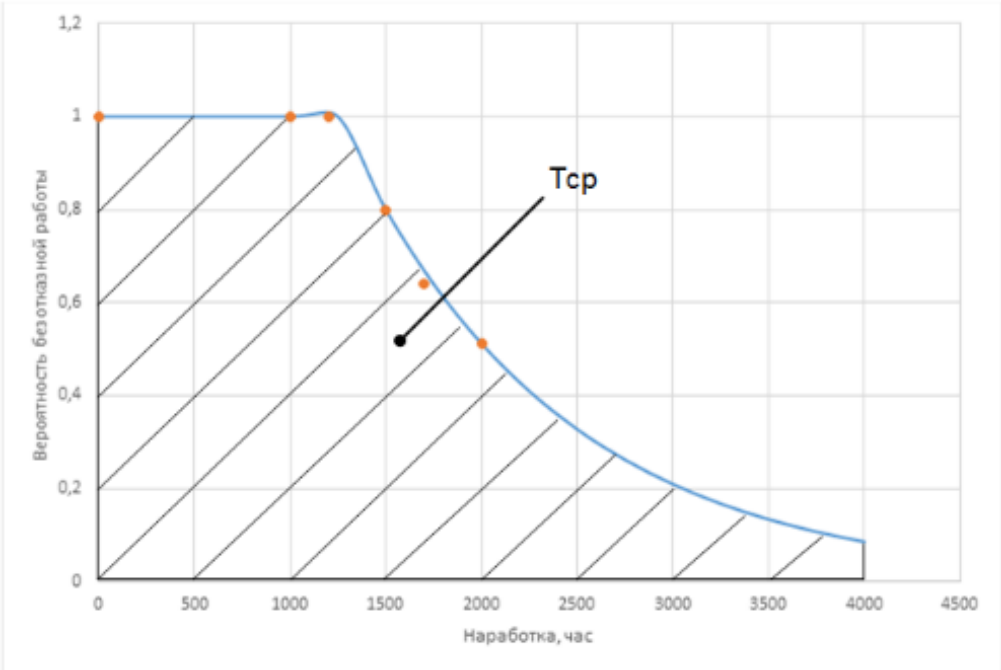


Рисунок 4.18 Зависимость вероятности безотказной работы от наработки

4.5 Выводы

1. Новые принципы, лежащие в основе построения систем логического управления, возникшие с развитием вычислительной техники, предполагают отличную от существующих форму организации систем, что потребовало формулирования новых методологических основ их проектирования.
2. В качестве аппаратной вычислительной платформы для построения систем логического управления могут быть использованы современные унифицированные решения, которые позволяют применять стандартный инструментарий проектирования и реализации математического обеспечения. В качестве унифицированных решений могут выступать: персональные, промышленные и одноплатные компьютеры.
3. Процесс проектирования и реализации систем логического управления итеративен, сложен и неоднозначен в выборе методов и средств. В связи с этим возникает необходимость в разработке инструментария проектирования систем. В качестве такого инструментария предложена методика, которая определяет фиксированный набор практических шагов, приводящих к требуемому результату.
4. Разработанные методологические аспекты позволяют предложить комплексное решение проблемы проектирования и реализации систем логического управления. Каждая фаза процесса разработки формирует конкретное решение, набор которых охватывает весь итерационный процесс проектирования и реализации системы.
5. Сформирован набор формализованных математических описаний, который позволяет производить проектирование программ логического управления любой сложности с использованием разработанной среды проектирования на языке функциональных блоков.
6. Комплексное тестирование системы логического управления должно включать: нагрузочное тестирование, тестирование на отказ и проведение приемо-сдаточных испытаний. Указанные виды тестовых испытаний позволяют сделать однозначные выводы о возможности эксплуатации разрабатываемой системы.

Глава 5 Практические аспекты реализации систем логического управления технологическим оборудованием

Глава посвящена решению практических задач реализации систем логического управления на основе разработанных принципов и их интеграции в работающее технологическое оборудование. Адекватность разработанных принципов построения систем первоначально проверена на экспериментальных стендах, что позволило до реализации рабочего варианта системы управления провести корректировку программного и аппаратного обеспечения. Концепция, на которой основана разработка, предполагает как автономное использование систем для управления отдельными технологическими узлами или процессами, так и встроенное решение для работы в составе комплекса систем управления (например, в рамках ЧПУ). [155-156] Для каждого из видов систем логического управления необходим специализированный экспериментальный стенд.

Поэтому было реализовано два стенда:

- комплексный стенд в составе которого дискретные и аналоговые модули ввода/вывода и различные типы объектов управления, который позволит проверить автономное решение;
- экспериментальный стенд проверки работоспособности системы логического управления, интегрированной в состав системы ЧПУ.

Стендовые испытания позволили установить, что разработанная система логического управления: работоспособна, корректно работает с аппаратным обеспечением, имеет время наработки на отказ свыше 1200 часов. После получения положительных результатов в рамках стендовых испытаний был реализован ряд систем управления реальным технологическим оборудованием. Это позволило оценить возможность применения разработанных принципов для решения разнородных задач управления, среди которых:

- Управление электроавтоматикой для установки гидроабразивной резки позволило проверить возможность использования ПЛК в качестве пассивного модуля ввода/вывода и работу аппаратного обеспечения в рамках конфигурации «ведущий-ведомый».
- Управление электроавтоматикой прецизионного станка наклонной компоновки позволило проверить работу системы управления с большим количеством аппаратных модулей ввода/вывода различного типа (дискретные, аналоговые, модули термосопротивлений и т.д.).
- Управление электроавтоматикой вертикально-фрезерного обрабатывающего центра позволило проверить работоспособность системы логического управления при работе в рамках мультипротокольной промышленной сети.

Каждая из разработанных систем позволила повысить характеристики как процесса проектирования и реализации системы (например, сокращение времени разработки программ логического управления), так и характеристики технологического процесса (информативность процесса, надежность и др.).

5.1 Применение систем логического управления как автономного решения для управления технологическим оборудованием

Практический опыт применения систем логического управления технологическим оборудованием базировался на экспериментальных исследованиях, которые были выполнены в рамках научно-исследовательских и опытно-конструкторских работ в «Московском государственном технологическом университете «СТАНКИН». Процесс проектирования, изготовления и проведения тестовых испытаний на экспериментальных стендах и установках позволил провести усовершенствование и уточнение структуры отдельных программных и аппаратных компонентов систем управления, а также был получен опыт в настройке и отладке систем управления нового класса. Основной задачей исследования на экспериментальных стендах была задача установления возможности промышленной эксплуатации разработанных систем. [157]

5.1.1 Разработка комплексного экспериментального стенда проверки работоспособности системы логического управления технологическим оборудованием

При проектировании испытательных стендов для проверки работоспособности систем логического управления наибольшее внимание было уделено вопросам выбора аппаратной платформы и совместимости аппаратного и программного обеспечения. Исходя из этого, было принято решение разработать комплексный стенд, который можно было бы переориентировать на работу с различными аппаратными платформами за короткий период времени [158]. В четвертой главе было продемонстрировано, что в качестве вычислительных устройств, на которые можно устанавливать ядро системы логического управления, были выбраны: одноплатный компьютер, персональный компьютер и компьютер промышленного исполнения. Эти же платформы были заложены в качестве аппаратной базы при проектировании экспериментального стенда (рисунок 5.1).

В качестве объектов управления были выбраны:

- Группа восьмисегментных индикаторов с элементами управления (кнопки) – «Объект управления 1». Объект управления позволяет проверить работу системы управления с дискретными сигналами ввода (кнопки) и вывода (сегменты индикаторов).
- Сервопривод на основе протокола EtherCAT – «Объект управления 2». Объект управления позволяет проверить функциональные блоки и алгоритмы управления движением, реализованные в системе управления.

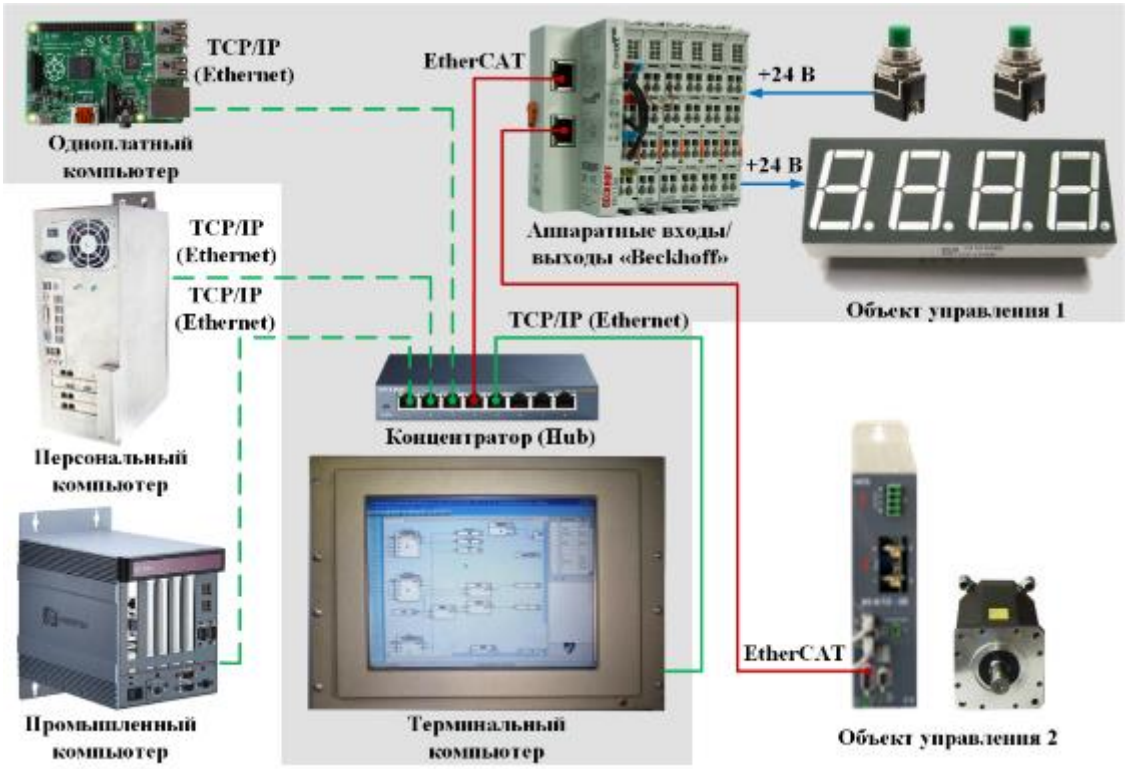


Рисунок 5.1 Распределенная модель экспериментального стенда проверки работоспособности системы логического управления

Экспериментальный стенд - это быстро модифицируемый объект, в котором есть постоянно работающие (соединены сплошными линиями) и подключаемые элементы (соединены пунктирными линиями). Основной задачей системы логического управления является контроль систем электроавтоматики, в котором более 70% сигналов являются дискретными, поэтому аппаратные входы/выходы и «Объект управления 1» - постоянно подключенные элементы, к которым также относится и терминальный компьютер со средой программирования.

В зависимости от проводимых тестов к стенду подключается одна из трех аппаратных платформ с установленным ядром управления. Все оборудование объединено в единую сеть посредством сетевого концентратора (hub).

Терминальный компьютер содержит среду разработки программ логического управления, в которой до начала работы разрабатывается программа на языке функциональных блоков.

Терминальный компьютер связан посредством сети Ethernet с базовым вычислительным модулем на котором установлено ядро системы логического управления. Разработанная программа логического управления передается в ядро системы, производится отладка программы и последующий запуск отлаженной версии программы логического управления. В дальнейшем связь с терминальным компьютером необходима только для визуального отображения работы программы логического управления в процессе работы стенда. В соответствии с методикой тестирования, разработанной в главе 4 для экспериментальных исследований выбирался один из вариантов распределенной модели стенда (выделен фоном на рисунке 5.1). На рисунке 5.2 представлен вариант экспериментального стенда с вычислителем на базе одноплатного компьютера и объектом управления в виде группы семисегментных индикаторов. [159]

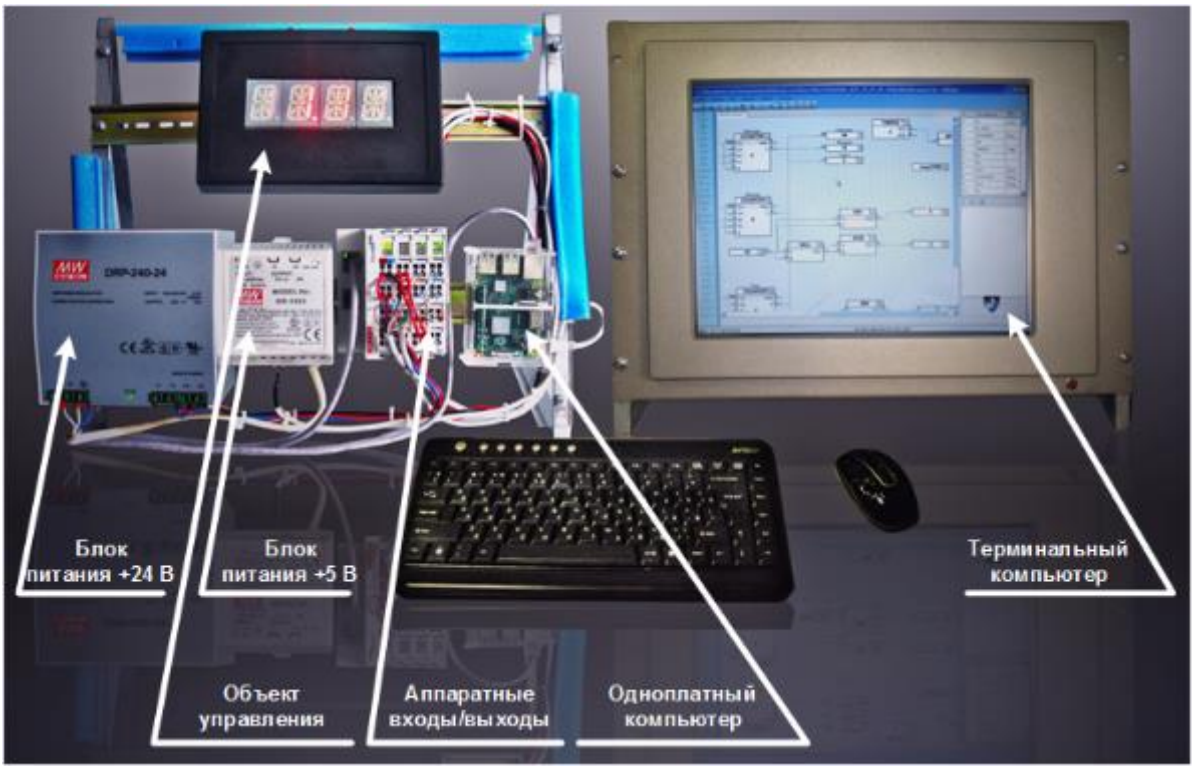


Рисунок 5.2 Стенд проверки системы логического управления на базе одноплатного компьютера

Стенд дополнительно снабжен элементами питания: для аппаратных входов/выходов блок питания +24В, 10А; для одноплатного компьютера блок питания +5В, 5А. На данной модификации стенда проверялась возможность работы системы логического управления на базе одноплатного компьютера с дискретными входами/выходами и быстродействие выбранной конфигурации системы.

Были проведены стендовые испытания на базе методики тестирования, разработанной в разделе 4.3, в ходе которых было установлено, что:

- разработанная системы логического управления работоспособна и представляет практический интерес;
- достигнута согласованность и стабильность при совместной работе среды разработки программ логического управления и ядра логического управления, что позволяет загружать и отлаживать программы даже большого размера (более 20 тысяч функциональных блоков);
- подтверждена корректная работа системы с аппаратными входами/выходами при обработке дискретных и аналоговых сигналов;
- время наработки на отказ ядра логического управления более 1000 часов, что соответствует требованиям, предъявляемые к промышленным системам;
- время выполнения цикла логического управления при работе в операционной системе реального времени составляет менее 10 мс даже на программах логического управления большого размера (более 20 тысяч функциональных блоков). Этот результат укладываете в требования, предъявляемые к промышленным системам жесткого реального времени.

5.2 Практический опыт применения систем логического управления для решения задач управления электроавтоматикой станков

Внедрение разработанных принципов реализации систем логического управления технологическим оборудованием было осуществлено в рамках опытно-конструкторских работ, проводимых на кафедре «Компьютерных систем управления» ФГБОУ ВО «МГТУ «СТАНКИН». Разработка прикладных решений в области управления электроавтоматикой технологического оборудования осуществлена автором совместно членами кафедры «Компьютерных систем управления» при научных консультациях д.т.н., проф. Мартинова Г.М. Во всех случаях практические решения разрабатывались по методике, описание которой приведено в разделе 4.2 для конкретных производственных задач, и представляли собой законченные системы управления. [160-162] В процессе практического внедрения результатов теоретических и экспериментальных исследований были учтены замечания и пожелания, полученные от конечных пользователей систем управления, был приобретен опыт в наладке и эксплуатации систем управления нового типа. [163-164]

5.2.1 Разработка экспериментального стенда проверки работоспособности системы логического управления, интегрированного в состав системы ЧПУ

Один из возможных вариантов применения систем логического управления – это программно реализованный логический контроллер, интегрированный в систему ЧПУ. [165-166] При управлении сложными объектами, к числу которых относятся станки и обрабатывающие центры, необходимо убедиться в корректности работы основного функционала и надежности работы системы. Для этих целей необходимо провести моделирование системы управления, интегрированной в состав системы ЧПУ на основе моделей, разработанных в главе 2. Изменения в структуре функциональной модели, разработанной в разделе 2.4, представлены на рисунке 5.3, среди них можно выделить следующие:

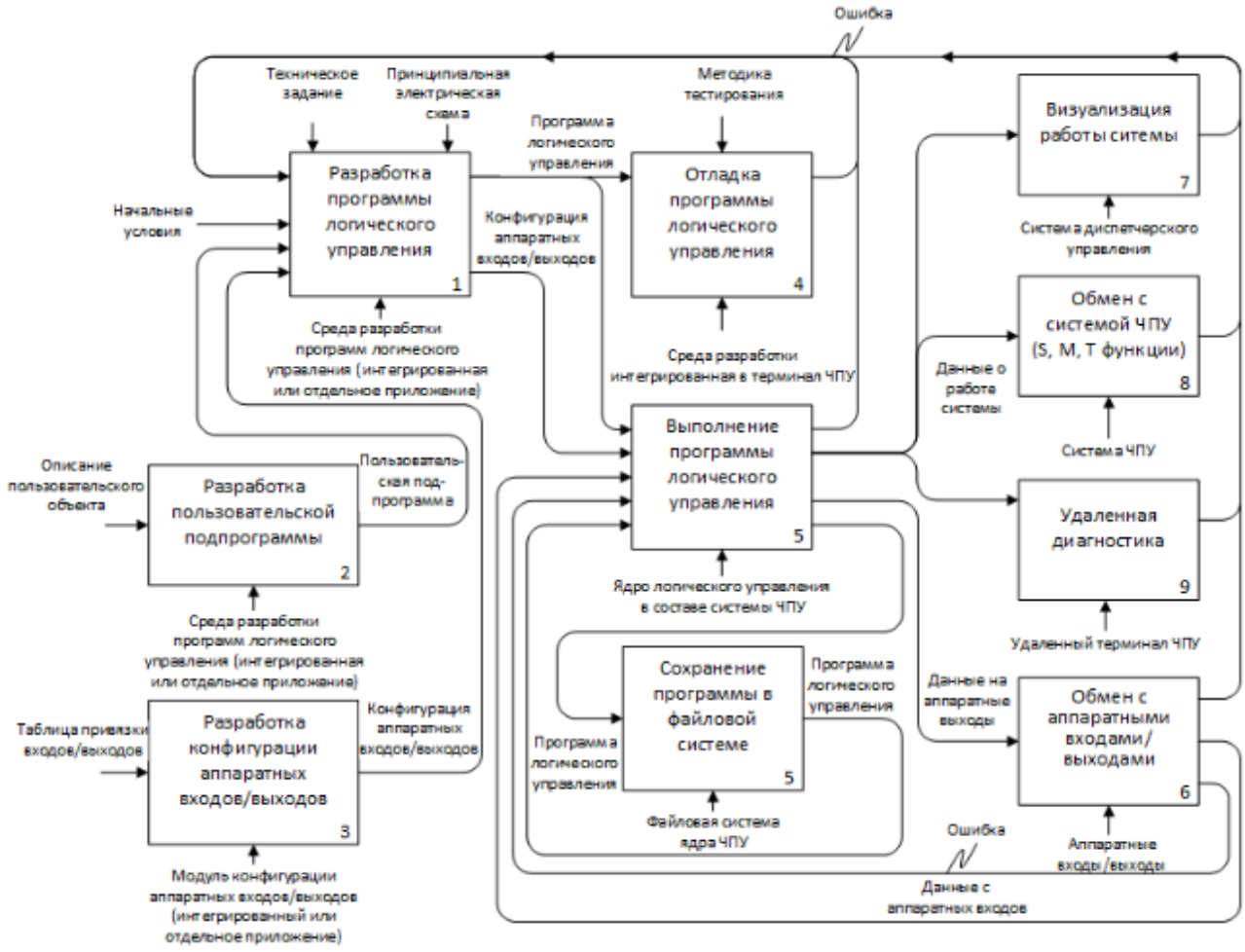


Рисунок 5.3 Уточненная функциональная модель программно реализованного контроллера, интегрированного в систему ЧПУ

- В качестве системы управления верхнего уровня выступает система ЧПУ, которая передает на программно-реализованный контроллер управляющие воздействия в виде S, M и T функций, указываемых в кадре управляющей программы. S, M, T функции относятся к группе

вспомогательных и позволяют при программировании систем ЧПУ осуществлять следующие действия: S – задает скорость вращения шпинделя, M – включает/отключает дополнительное технологическое оборудование станка, T – задает номер инструмента при его смене.

- В качестве системы удаленной диагностики выступает удаленный терминал ЧПУ, посредством которого можно получать диагностическую информацию, в том числе и с логического контроллера.
- Разработка программ логического управления осуществляется в среде разработки программ, выполненной либо в виде отдельного приложения, либо интегрированной в состав терминального клиента ЧПУ. При этом отладку программ целесообразно осуществлять на терминальном клиенте системы ЧПУ.
- Модуль конфигурации аппаратных входов/выходов имеет две версии, как в виде отдельного приложения, так и в интегрированном в состав терминального клиента ЧПУ виде.
- Выполнение программы логического управления осуществляется в ядре логического управления, интегрированном в состав программно-математического обеспечения системы ЧПУ.
- Сохранение программ логического управления осуществляется в файловую систему ядра ЧПУ.

Для интеграции системы логического управления в состав системы ЧПУ необходимо следующим образом преобразовать модель по типу виртуальной машины, разработанной в разделе 2.5 (рисунок 5.4):

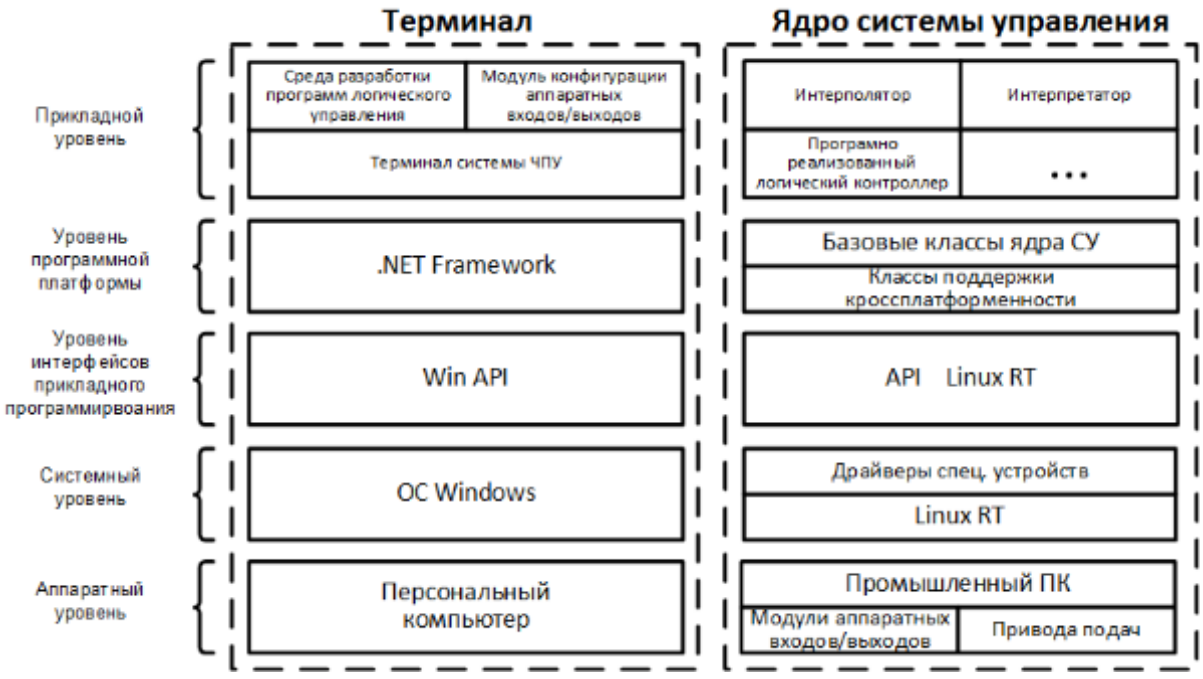


Рисунок 5.4 Уточненная модель по типу виртуальной машины для системы ЧПУ с интегрированным программно реализованным контроллером

- В качестве аппаратной базы для установки яра системы ЧПУ с интегрированным программно-реализованным логическим контроллером должен использоваться промышленный компьютер, который имеет физическую связь с аппаратными входами/выходами и приводами подач. Необходимость компьютера промышленного исполнения обусловлена тем, что вычислительных ресурсов одноплатного компьютера для одновременного выполнения функций системы ЧПУ и логического контроллера недостаточно, а персональный компьютер не подходит для применения в промышленных условиях из-за недостаточной защиты корпуса по индексу IP (Ingress Protection Rating — степень защиты от проникновения).
- На прикладном уровне в состав терминала системы ЧПУ интегрирована среда разработки программ логического управления и модуль конфигурации аппаратных входов/выходов.
- В ядре системы управления на прикладном уровне в программно-математическое обеспечение ЧПУ интегрирован программно реализованный логический контроллер.

Потоковая модель данных системы ЧПУ с интегрированным программно реализованным контроллером (рисунок 5.5) имеет следующие изменения по сравнению с базовой моделью, рассмотренной в разделе 2.6:

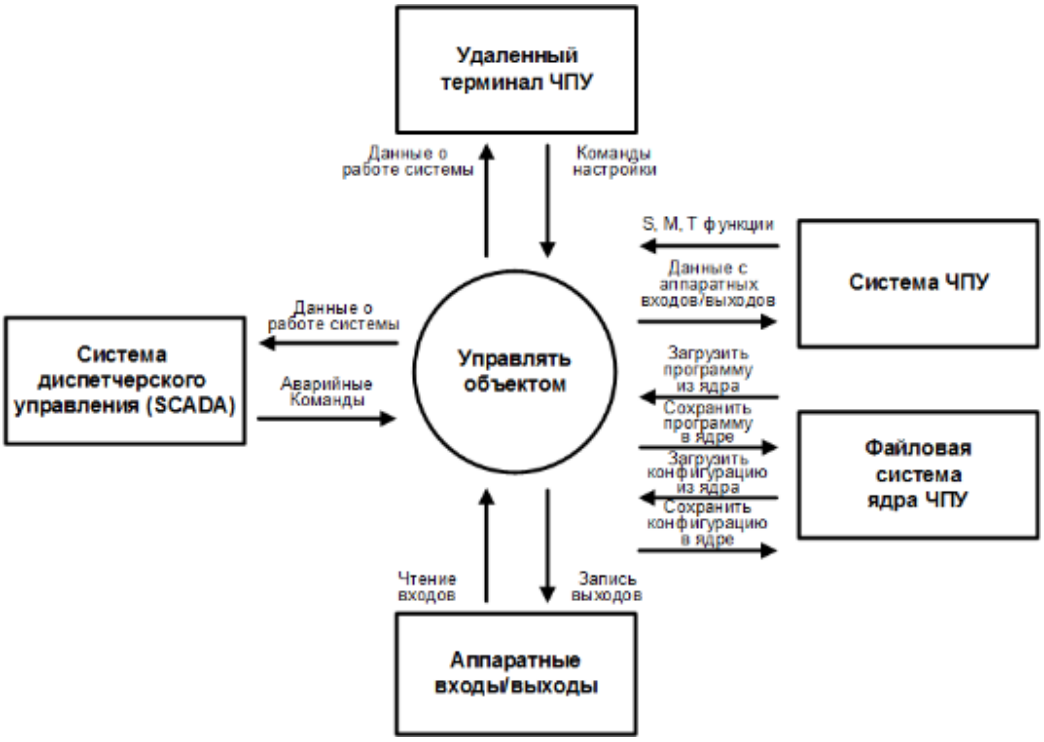


Рисунок 5.5 Уточненная потоковая модель данных системы ЧПУ с интегрированным программно реализованным контроллером

- Удаленный терминал системы ЧПУ выполняет функцию удаленной диагностики и настройки. При этом для передачи информации о работе программно реализованного логического контроллера и получения команд настройки используется стандартный канал взаимодействия.

- Системой управления верхнего уровня является система ЧПУ, для взаимодействия с ней используются внутренние программные средства (например, разделяемая память). По внутренним каналам взаимодействия осуществляется получение управляющих воздействий в виде S, M и T функций и передаются данные с аппаратных входов/выходов;
- Сохранение и загрузка программ логического управления производятся в файловой системе ЧПУ, для этого используются стандартные механизмы взаимодействия с файловой системой.

Архитектура системы ЧПУ с интегрированным программно реализованным логическим контроллером представлена на рисунок 5.6. и имеет следующие особенности:

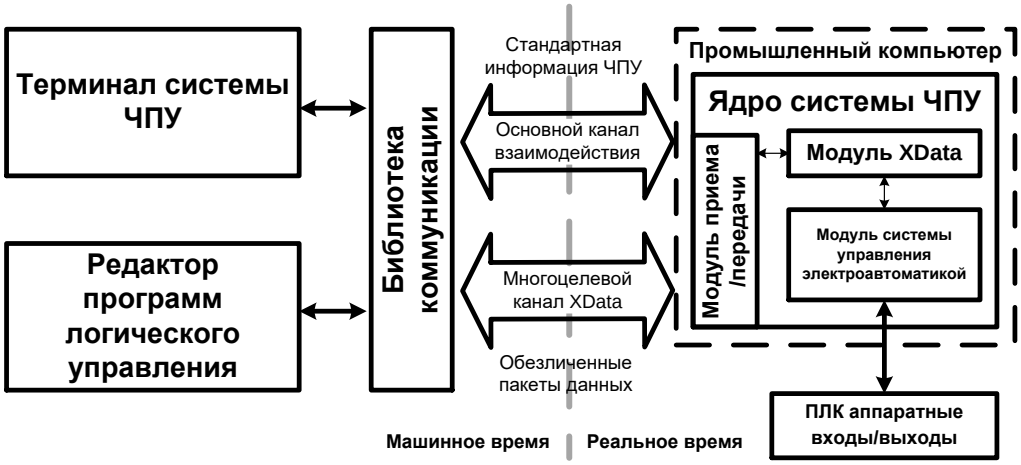


Рисунок 5.6 Архитектура системы ЧПУ с интегрированным программно реализованным контроллером

- редактор программ логического управления может быть как встроен в состав терминала системы ЧПУ, так и работать автономно. Встраивание производится за счет программных интерфейсов, которые позволяют расширять функционал терминала за счет внедрения пользовательских приложений;
- терминал системы ЧПУ и редактор программ логического управления используют для связи с ядром системы ЧПУ в единую библиотеку коммуникации. При этом информация о работе системы ЧПУ передается по основному каналу взаимодействия, а данные с системы логического управления передаются по дополнительному многоцелевому каналу «XData»;
- в ядре системы ЧПУ встроен модуль логического управления, который выполняет основные функции по решению логической задачи управления.

На основе разработанных моделей был реализован экспериментальный стенд проверки работоспособности системы логического управления, интегрированной в состав ЧПУ (рисунок 5.7). Экспериментальный стенд состоит из: металлической стойки с установленными на ней терминальным компьютером, промышленной клавиатурой и рядом объектов управления, объ-

единенных в едином корпусе. В качестве объектов управления выбраны: комплектные электроприводы (электродвигатель + контроллер привода) и аппаратные входы/выходы производства «СТАНКИН NC».

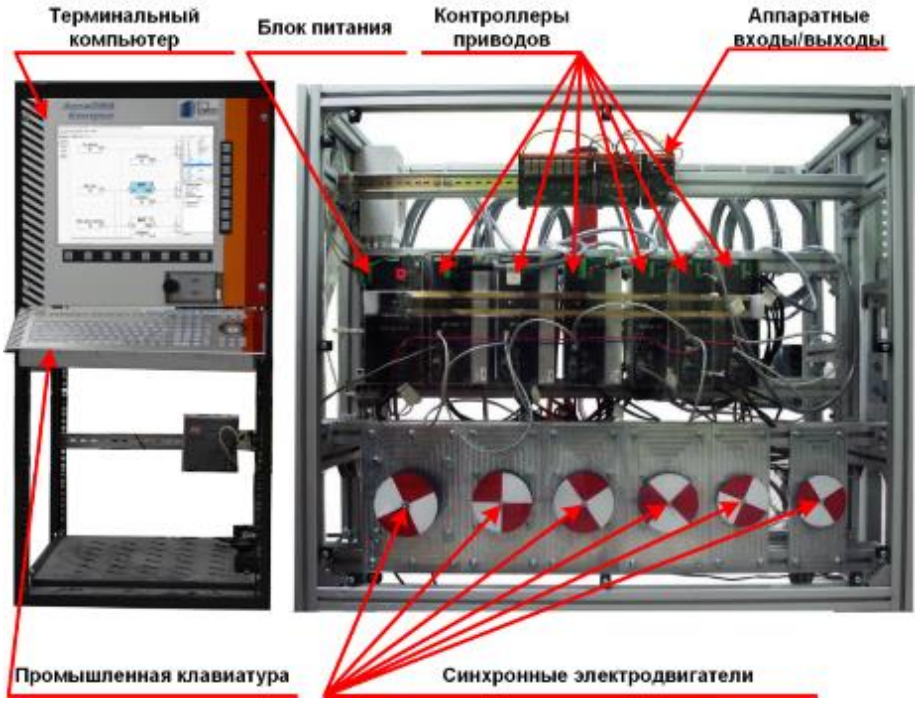


Рисунок 5.7 Экспериментальный стенд проверки работоспособности системы логического управления, интегрированной в состав ЧПУ

На терминальном компьютере запускается среда разработки программ логического управления или система ЧПУ, которая включает в себя стандартную и специализированную аппаратуру, а также ОСРВ [1]. Терминал оператора с рядом функциональных (F-keys) и машинных (M-keys) клавиш, расположенных снизу и справа от дисплея соответственно, служит для: выбора режима управления, ввода/редактирования управляющих программ (УП), быстрого вызова команд логического управления и отображения текущего состояния системы. В результате многочисленных экспериментов, проведенных по методике тестирования разработанной в главе 4, установлено что:

- Удалось экспериментально реализовать систему логического управления, встроенную в программно-математическое обеспечение ЧПУ;
- Система логического управления, интегрированная в состав ЧПУ работоспособна, обладает высокой надежностью, отказоустойчивостью, легко встраивается в современную систему ЧПУ,
- представляет практический интерес и может быть осуществлено внедрение системы при управлении технологическим оборудованием в промышленных условиях.

- Выполнение цикла управления в отдельном потоке не приводит к увеличению времени выполнения цикла решения геометрической задачи ЧПУ.
- Время выполнения цикла управления при работе в ядре системы ЧПУ в рамках ОСРВ составляет менее 10 мс. Результат соответствует требованиям, предъявляемым к промышленным системам жесткого реального времени.
- Интеграция среды разработки программ логического управления в терминал системы ЧПУ не привела к возникновению внештатных ситуаций при работе терминала. Терминал имеет устойчивую связь с ядром ЧПУ.
- Среда реализации программ логического управления, встроенная в терминал ЧПУ, позволяет сократить время отладки программ за счет оперативного поиска и исправления незначительных ошибок и опечаток в реализованных функциональных блоках, «не отходя» от технологического оборудования.
- Время наработки на отказ системы ЧПУ с интегрированным в неё ядром логического управления составляет более 1200 часов, что укладывается в требования, предъявляемые к промышленным системам.

5.2.2 Реализация системы логического управления электроавтоматикой экспериментального станка гидроабразивной резки

Одним из первых опытов применения разработанных принципов построения систем логического управления была система управления электроавтоматикой комплекса гидроабразивной резки УГСР (аббревиатура, Установка Гидро Струйной Резки), разработанная ОАО «Савеловский машиностроительный завод» (г. Кимры, Тверская область) в рамках совместного опытно конструкторского проекта с МГТУ «СТАНКИН» и ОАО «НИАТ».

Гидроабразивная резка – универсальная технология обработки, позволяющая получать детали и изделия различной конфигурации из металлов, сплавов, бетона, гранита, стекла и других материалов с высокой точностью и производительностью. В настоящее время является не только альтернативой механической, лазерной и другим видам обработки, но и единственным возможным в ряде случаев способом воздействия на заготовку без ее нагрева. Высокопроизводительный пятикоординатный комплекс гидроабразивной резки имеет шесть управляемых осей: X и Z – обычные линейные, Y и V – портальные, A и C – круговые, что позволяет обрабатывать детали сложной формы (рисунок 5.8).

УГСР обладает следующими техническими характеристиками: номинальное давление воды 600 МПа; рабочая зона 2500x1500 мм; допустимая высота детали 1000 мм; максимальная

скорость перемещения по линейным координатам 25 000 мм/мин; погрешность перемещений по линейным координатам, 50 мкм. Задача логического управления разрабатывалась как составная часть программно-математического обеспечения системы ЧПУ «АксиОМА Контрол» (разработка МГТУ «Станкин»), которой оснащается комплекс УГСР.

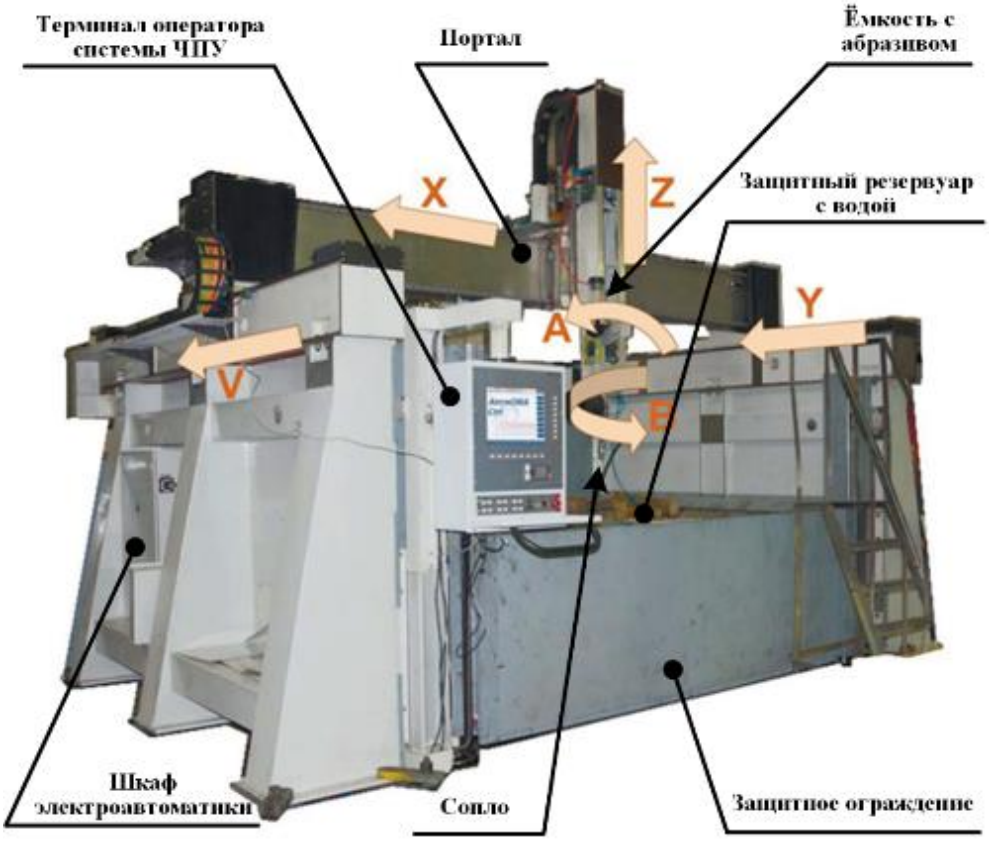


Рисунок 5.8 Комплекс гидроабразивной резки УГСР

Разработка сетевой структуры системы управления. Согласно методике построения систем логического управления (раздел 3.2) разработана сетевая структура системы (рисунок 5.9), в состав которой входят: ядро со встроенным программно реализованным контроллером, терминал оператора, аппаратные модули входов/выходов, система управления автономной станцией высокого давления, контроллеры приводов подач (для 6 осей, с учетом портальной кинематики) и станочная панель управления. [167]

Автономное управление станцией высокого давления при наладке осуществляется посредством установленного на ней пульта. В штатном режиме управление происходит с помощью станочной панели на терминале системы ЧПУ [168]. Станочная панель позволяет управлять положением защитного ограждения, отсекателем струи, подачей питания к станции высокого давления (СВД) и гидростанции. Управляющие сигналы через ядро системы ЧПУ передаются в программно реализованный контроллер или в систему управления СВД для отработки заложенных алгоритмов.

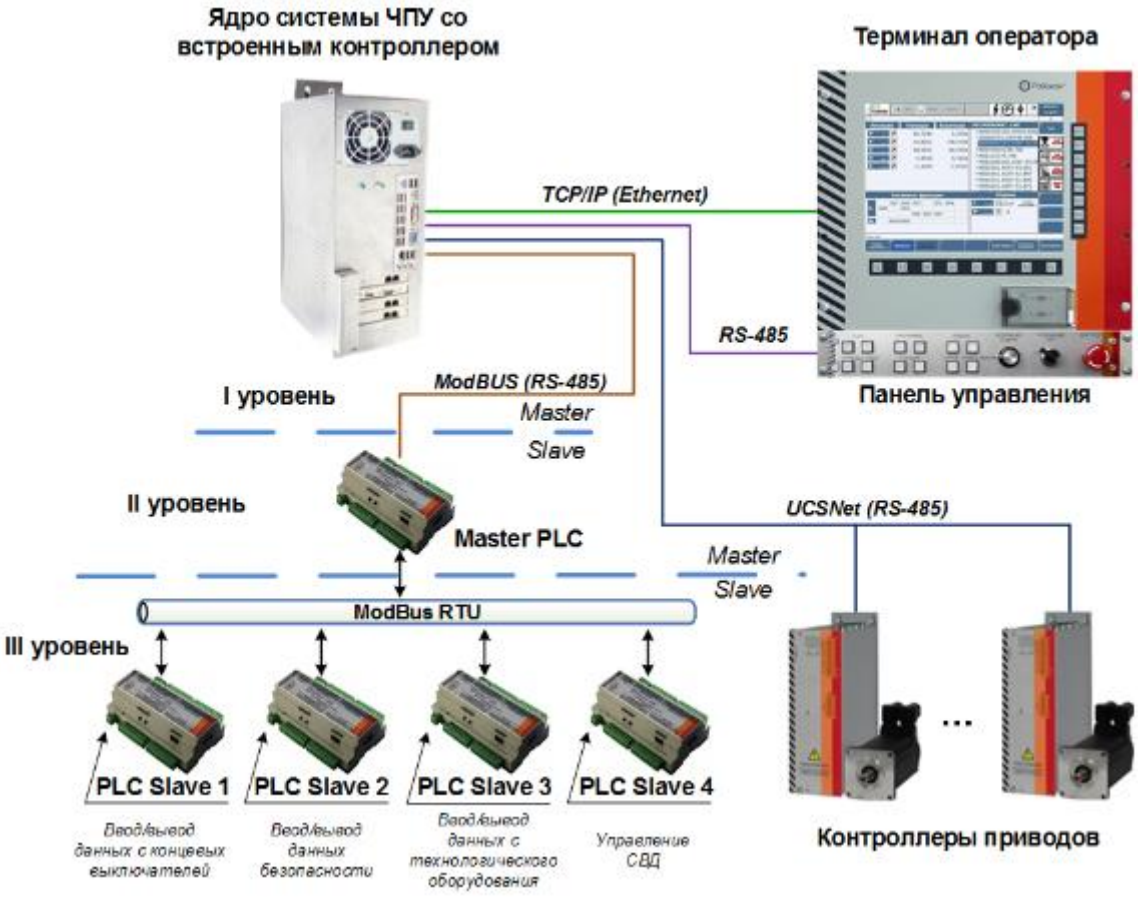


Рисунок 5.9 Сетевая структура системы управления комплексом УГСР

Логическая задача ЧПУ решается посредством распределенной системы управления, представляющей собой совокупность из программно-реализованного контроллера и модулей аппаратных входов/выходов, непосредственно влияющих на технологический процесс. [169] Сбор и обмен данными в сети между вычислительными устройствами осуществляется на базе коммуникационного протокола ModBus с применением на физическом уровне стандарта последовательной линии связи RS-485. Взаимодействие автономных модулей входов/выходов организовано по принципу «master - slave» («ведущий – ведомый») с образованием многоранговой сети. [167] На верхнем уровне располагается система ЧПУ, управляющая всем технологическим процессом. Программно-реализованное ядро системы логического управления, встроенное в систему ЧПУ реализует функции «Master» устройства, к числу которых относится диспетчеризация команд управления и данных поступающих со «Slave» устройств. На втором уровне расположен «ведущий» ПЛК, отвечающий за согласованность работы совокупности элементов распределенной системы. На третьем уровне располагаются «ведомые» ПЛК, реализующие управление питанием приводов подач, СВД, контроль безопасности и т.д.

Выделение специализированных функций и команд. На третьем шаге методики разработки производится учет особенностей работы объекта управления. Это позволило помимо

стандартных М-функций в системе управления комплексом УГСР разработать специализированные функции управления электроавтоматикой (таблица 5-1).

Таблица 5-1 Систематизация вспомогательных М-функций комплекса УГСР

М-функция	**Назначение**
М108	Открыть заслонку подачи воздуха с абразивом в режущую головку.
М109	Закрыть заслонку подачи воздуха с абразивом в режущую головку после выполнения всех инструкций заданных в кадре.
М110	Задать количество абразивного материала, подаваемого в режущую головку. Команда имеет один параметр, принимающий значения в диапазоне 0…100 и указывает степень открытости заслонки.

Составление таблицы привязки входов/выходов. В связи с необходимостью управлять двумя взаимосвязанными технологическими объектами (установка УГСР и станция СВД), а также в связи наличием многоранговой сети, было принято решение в качестве аппаратных модулей ввода/вывода использовать ПЛК отечественного производства «Робокон». Непосредственная коммуникация осуществляется только между смежными уровнями (соответственно, 1-2 и 2-3). Механизм взаимодействия базируется на использовании разделяемой памяти (применяется внутренняя память подчиненных контроллеров), доступной для чтения и записи данных как «master»-, так и «slave» устройствам.

В зависимости от конкретной реализации «master» контроллер сети может:

1. Самостоятельно выполнять поступающие команды по заложенным в его программе алгоритмам. При этом «slave» устройства фактически выступают в роли пассивных аппаратных модулей входов/выходов, выполняя лишь арифметические и/или логические операции над данными с целью их предварительной фильтрации (например, для передачи только определенного диапазона значений).
2. Только распределяет поступающие команды между «ведомыми». Обработкой и последующим выполнением команд занимаются устройства третьего уровня. При обмене данными между ними, а также с системой числового управления станка «master» контроллер также выступает и в роли коммутатора.
3. Реализует обе вышеописанные задачи, что позволит сократить количество применяемого оборудования за счет рационального использования функциональных возможностей контроллеров.

В соответствии с сетевой моделью (рисунок 5.9) контроллеры PLC_Slave1 – PLC_Slave3 используются для управления УГСР, а контроллер PLC_Slave4 – для управления станцией высокого давления. Для каждого из контроллеров на четвертом шаге методики разработки была составлена таблица привязки аппаратных входов/выходов (таблица 5-2).

Таблица 5-2 Привязки аппаратах входов выходов УГСР

Master_PLC (Робокон 1455)			
Дискретные входы		Дискретные выходы	
I0.0	Питание приводов включено	Q0.0	Не задействован
I0.1	Контроль питания 24В	Q0.1	Не задействован
I0.2	Контроль питания ПЛК	Q0.2	Не задействован
I0.3	Питание датчиков двигателей	Q0.3	Не задействован
I0.4	Контроль питания выходов 24В	Q0.4	Не задействован
I0.5	Общая авария	Q0.5	Не задействован
I0.6	Авария шкафа управления	Q0.6	Не задействован
I0.7	Не задействован	Q0.7	Не задействован
Аналоговые входы		Аналоговые выходы	
AI0	Перегрев двигателя M1	AQ1.0	Управление подачей абразива
AI1	Перегрев двигателя M2	AQ1.1	Не задействован
AI2	Перегрев двигателя M3	AQ1.2	Не задействован
AI3	Перегрев двигателя M4	AQ1.3	Не задействован
AI4	Перегрев двигателя M5	AQ1.4	Не задействован
AI5	Перегрев двигателя M6	AQ1.5	Не задействован
AI6	Не задействован	AQ1.6	Не задействован
AI7	Не задействован	AQ1.7	Не задействован
PLC_Slave1 (Робокон 1456)			
Дискретные входы		Дискретные выходы	
I0.0	Привод U1 к работе готов (ось X)	Q0.0	Питание 380В приводов вкл.
I0.1	Привод U2 к работе готов (ось Y)	Q0.1	Работа U1 разрешена (ось X)
I0.2	Привод U3 к работе готов (ось V)	Q0.2	Работа U2 разрешена (ось Y)
I0.3	Привод U4 к работе готов (ось Z)	Q0.3	Работа U3 разрешена (ось V)
I0.4	Привод U5 к работе готов (ось A)	Q0.4	Работа U4 разрешена (ось Z)
I0.5	Привод U6 к работе готов (ось B)	Q0.5	Работа U5 разрешена (ось A)
I0.6	Не задействован	Q0.6	Работа U6 разрешена (ось B)
I0.7	Не задействован	Q0.7	Сброс ошибки приводов U1-U6
I1.0 – I1.7	Не задействованы	Q1.0	Деблокировка тормоза оси Z
		Q1.1– Q1.7	Не задействованы
PLC_Slave2 (Робокон 1456)			
Дискретные входы		Дискретные выходы	
I0.0	Привод оси X в нулевой точке	Q0.0	Включить гидростанцию
I0.1	Привода Y, V в нулевой точке	Q0.1	Поднять ограждение
I0.2	Привод оси Z в нулевой точке	Q0.2	Опустить ограждение
I0.3	Не задействован	Q0.3	Управление отсекателем струи
I0.4	SQ1 привод оси X авария "+"	Q0.4	Не задействован
I0.5	SQ2 привод оси X авария "-"	Q0.5	Не задействован
I0.6	SQ3 привод оси Y, V авария "+"	Q0.6	Не задействован
I0.7	SQ4 привод оси Y, V авария "-"	Q0.7	Не задействован
I1.0	SQ5 привод оси Z авария	Q1.0	Не задействован
I1.1	SQ6 привод оси A авария	Q1.1	Не задействован
I1.2	SQ7 привод оси A авария "+"	Q1.2	Не задействован
I1.3	SQ8 привод оси A авария "-"	Q1.3	Не задействован

Продолжение таблицы 5-2.

I1.4	SB13_2 УГСР аварийный стоп	Q1.4	Не задействован
I1.5	SB3_2 СВД аварийный стоп	Q1.5	Не задействован
I1.6	Не задействован	Q1.6	Не задействован
I1.7	Не задействован	Q1.7	Не задействован
PLC_Slave3 (Робокон 1456)			
Дискретные входы		Дискретные выходы	
I0.0	СВД питание подано (QF2)	Q0.0	Не задействован
I0.1	Гидростанция питание подано	Q0.1	Не задействован
I0.2	Подготовка воды питание подано	Q0.2	Не задействован
I0.3	Гидростанция уровень масла	Q0.3	Не задействован
I0.4	Гидростанция фильтр в норме	Q0.4	Не задействован
I0.5	Гидростанция давление масла	Q0.5	Не задействован
I0.6	Подготовка воздуха давление	Q0.6	Не задействован
I0.7	Подготовка воздуха осушка	Q0.7	Не задействован
I1.0	SQ15 ограждение вверху	Q1.0	Не задействован
I1.1	SQ16 ограждение внизу	Q1.1	Не задействован
I1.2	Уровень абразива в норме	Q1.2	Не задействован
I1.3	СВД питание подано (QF2)	Q1.3	Не задействован
I1.4	Гидростанция питание подано	Q1.4	Не задействован
I1.5	Не задействован	Q1.5	Не задействован
I1.6	Не задействован	Q1.6	Не задействован
I1.7	Не задействован	Q1.7	Не задействован
PLC_Slave4 (Робокон 1456)			
Дискретные входы		Дискретные выходы	
I0.0	Уровень масла в баке в норме	Q0.0	Включить главный насос
I0.1	Температура масла в норме	Q0.1	Включить центробежн. насос
I0.2	Давление воды мультиплексора	Q0.2	СВД к работе готова
I0.3	Масляный фильтр загрязнён	Q0.3	Мультипликатор влево
I0.4	Масляный фильтр чист	Q0.4	Мультипликатор вправо
I0.5	Поршень мультиплексора слева	Q0.5	Включить насос
I0.6	Поршень мультиплексора справа	Q0.6	Не задействован
I0.7	Управление внешнее	Q0.7	Не задействован
I1.0	Управление местное	Q1.0	Не задействован
I1.1	Давление высокое	Q1.1	Не задействован
I1.2	Давление низкое	Q1.2	Не задействован
I1.3	Пуск насоса	Q1.3	Не задействован
I1.4	Мультипликатор работа	Q1.4	Не задействован
I1.5	Мультипликатор СТОП	Q1.5	Не задействован
I1.6	Двигатель главного насоса вкл.	Q1.6	Не задействован
I1.7	Аварийная кнопка нажата	Q1.7	Не задействован

Система управления использует два типа контроллеров для организации ввода/вывода данных: «Робокон 1456» с 16 дискретными входами и 16 дискретными выходами и «Робокон 1456» с 8 дискретными входами, 8 дискретными выходами, 8 аналоговыми входами и 8 анало-

говыми выходами. Таблица привязки проектировалась таким образом, чтобы каждый из контроллеров имел резервные входы и выходы, которые можно использовать в дальнейшем при модернизации. Распределенное управление организованно с применением независимо работающих ПЛК и позволяет автономно вести разработку программного обеспечения для каждого отдельного контроллера с последующей интеграцией в систему и вводом оборудования в эксплуатацию. Это актуально при объединении в единые комплексы нескольких технологических объектов, каждый из которых должен иметь независимые средства управления.

Настройка режимов отображения экрана оператора. Пятый шаг методики построения определяет настройку режимов отображения экрана оператора. Каждому состоянию отдельного узла или системы в целом сопоставлены графическое изображение и/или текстовое сообщение со своим уникальным идентификатором, которые отображаются в отдельной области экрана оператора, называемой строка состояний (анг. Status Bar). На основе обработанных данных программно реализованный контроллер записывает в область внутренней памяти идентификаторы изменившихся состояний. Аналогично работает механизм индикации машинных клавиш на панели управления: статус управляемых от станочной панели объектов записывается в биты памяти программируемого контроллера; система ЧПУ считывает данные и передает инструкции контроллеру станочной панели, отвечающему за подсветку. Настройка параметров отображения состояний на экране оператора осуществляется с помощью XML-файлов. В них сопоставлены указатели внутренней области памяти контроллера с таблицами, содержащими набор текстовой и графической информации, необходимой для визуализации. Текущее и возможные состояния системы отображаются на терминале оператора (рисунок 5.10).

Система использует три XML файла: PlcConfig.xml – массив адресов используемых изображений, массив строк сообщений и массив используемых M команд; M_key_text.xml – текст, отображаемый на M кнопках экрана оператора; PLCMessages.xml – массив сообщений об ошибках, массив текстовых предупреждений и массив информационных сообщений. Строка состояний экрана оператора УГСР содержит пять полей, в которых отображается следующая информация: статус разрешения движения, наличие питания приводов, наличие давления СВД, уровень абразива, состояние защитного ограждения. Машинные клавиши экрана оператора настроены на работу со следующими исполнительными устройствами УГСР: откр./закр. заслонку подачи жидкости, откр./закр. ограждение, вкл./откл. питания приводов, вкл./выкл. станцию высокого давления.

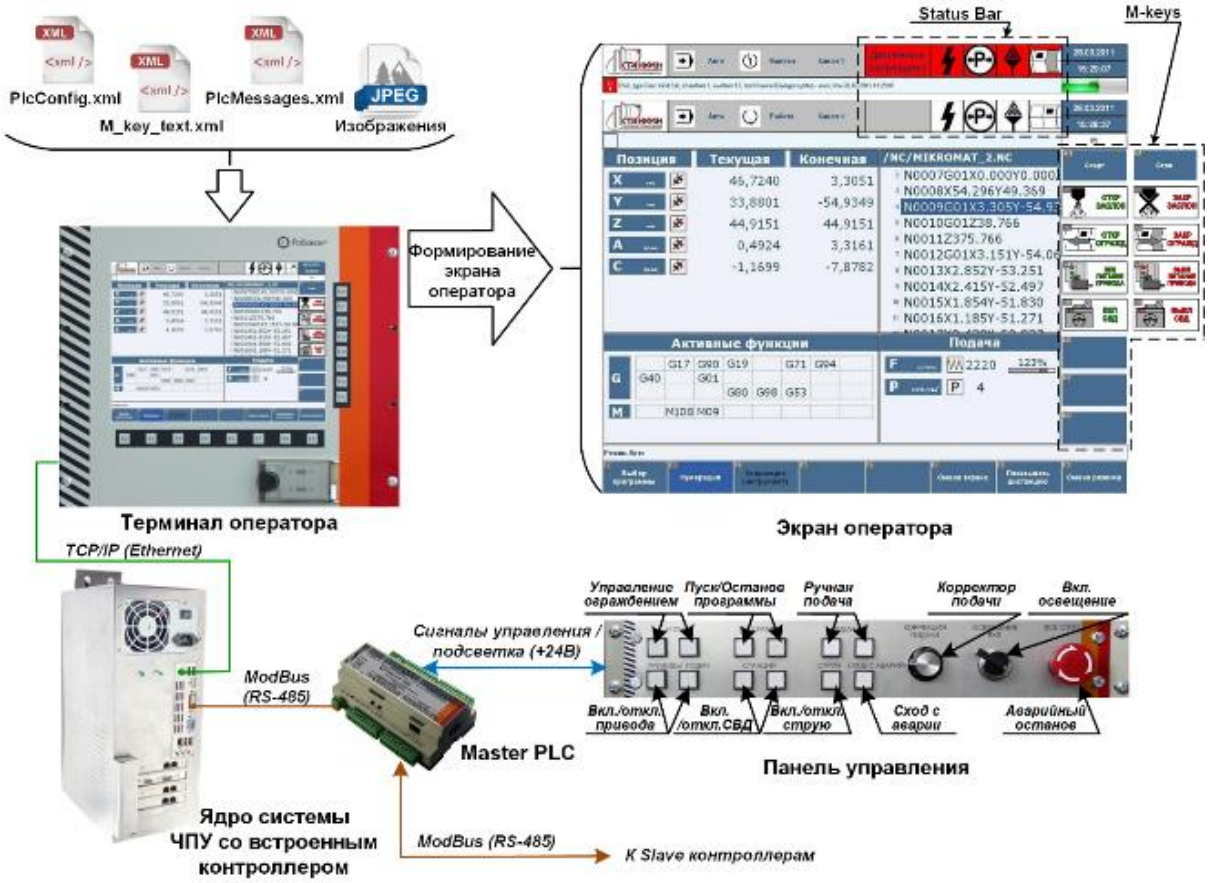


Рисунок 5.10 Схема формирования экрана оператора УГСР

Разработка специализированных библиотек и программы логического управления. Согласно методике построения, на шестом шаге разрабатываются специализированные библиотеки функциональных блоков, которые будут в дальнейшем интегрированы в программу логического управления. В качестве примера рассмотрим управление гидроабразивной струей в установке УГСР, схематичное изображение которой представлено на рисунке 5.11.

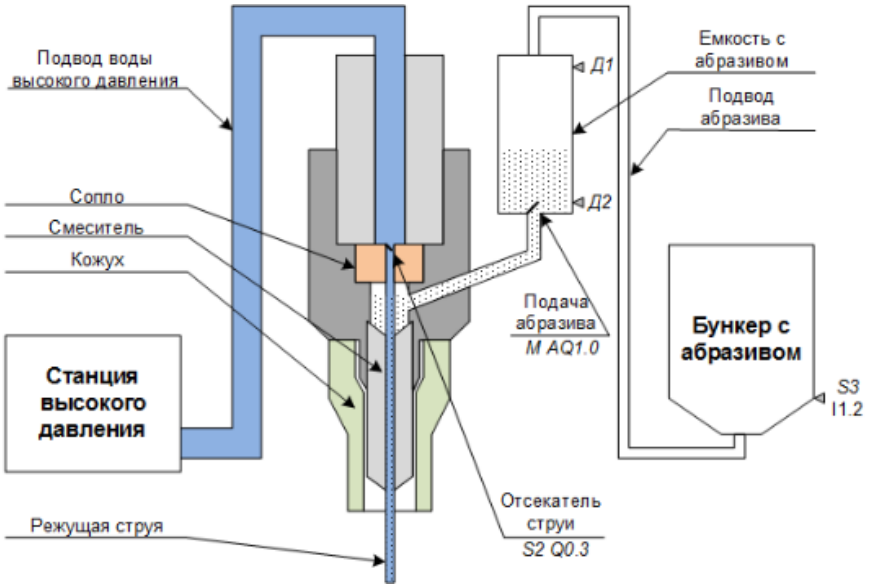


Рисунок 5.11 Схема управления гидроабразивной струей УГСР

В режущей головке осуществляется процесс смешивания воды под высоким давлением (до 5 000 атмосфер) и абразивного материала (гранатовый песок). За формирование струи воды под высоким давлением отвечает СВД, имеющая автономный контроллер, обозначенный на рисунке 5.9 «PLC_Slave_4». Электроавтоматика СВД осуществляет контроль: входного и выходного давления в гидравлической системе; уровень рабочих жидкостей и загрязнение фильтрующих элементов; общего времени работы установки (из-за работы с жидкостью сверх высокого давления установка имеет ограниченный ресурс работы отдельных механических узлов и элементов); сигналов, поступающих с элементов управления, расположенных на панели управления установкой (аварийная кнопка, пуск/останов, сброс ошибки и т.д.). Струя воды под высоким давлением проходит через сопло в режущей головке и при открытой заслонке (управляется контроллером PLC_Slave2, сигнал Q3) попадает в смеситель, где происходит её смешение с абразивным материалом. Абразивный материал поступает из накопительной емкости, расположенной непосредственно на УГСР. Накопительная емкость имеет ограниченный объем и свою систему управления, которая поддерживает уровень абразивного материала в заданном диапазоне (между датчиками Д1 и Д2). Основной объем абразивного материала находится в бункере рядом с УГСР, который оснащен датчиком (контроллер PLC_Slave2, вход I1.2), при срабатывании которого система управления оповещает оператора о необходимости пополнения абразивного материала в бункере. Количество подаваемого абразивного материала в смеситель определяется уровнем открытости заслонки накопительной емкости (от 0 до 100%), заслонка управляется аналоговым выходом master-контроллера (AQ 1.0)

Разработанная система логического управления позволяет использовать при программировании алгоритмов управления как встроенную среду разработки, так и среду разработки, поставляемую с контроллерами. При управлении УГСР использована комбинация подходов: программирование алгоритмов управления формированием гидроабразивной струи (рисунок 5.12) и управление СВД запрограммированы непосредственно на контроллерах, а алгоритмы контроля безопасности, управления питанием приводов, управление пневмосистемой и панелью управления - на базе программно реализованного контроллера, встроенного в систему управления.

Включение/отключение отсекателя гидравлической струи определяется командами M108 и M109, а также сигналом с панели управления. При наличии команды включения отсекателя и отсутствии нерегулярных ситуаций (аварийный останов, СВД отключена и др.) осуществляется подача гидравлической струи.

Применение системы логического управления установкой УГСР совместно со специализированной технологией обработки позволило контролировать и значительно экономить расход

абразивного материала. Экономия не приводит к ухудшению качественных характеристик поверхности реза и составляет от 25 до 30 % по сравнению с установками аналогичного класса и зависит от типа обрабатываемой детали и материала.

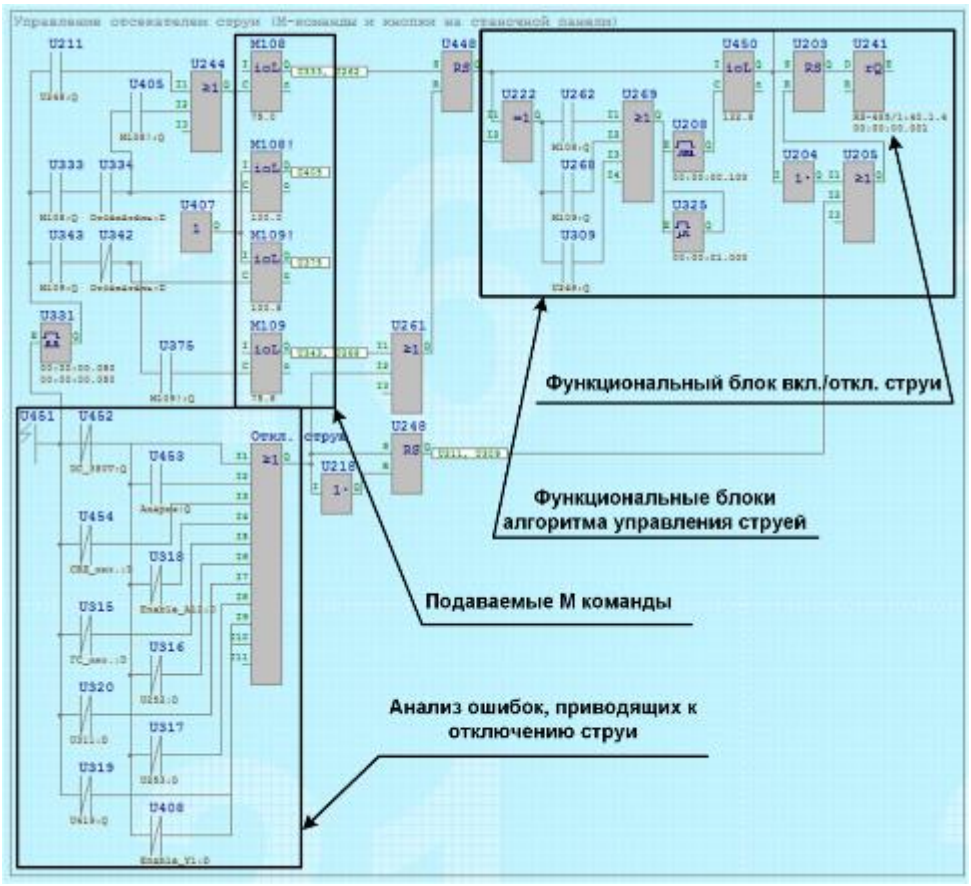


Рисунок 5.12 Программная реализация управления отсекателем струи

5.2.3 Реализация системы логического управления электроавтоматикой гаммы экспериментальных токарно-фрезерных обрабатывающих центров наклонной компоновки

Расширение номенклатуры и функциональных возможностей станков ведет к количественному и качественному увеличению узлов электроавтоматики, основного и вспомогательного технологического оборудования. [170-171]. При этом архитектура системы логического управления технологическим оборудованием значительно усложняется, а программа электроавтоматики разрабатывается индивидуально, исходя из особенностей реализации конкретных узлов. В этом случае применение разработанных принципов организации дает наибольший эффект. [173-175]

В качестве примера практического решения логической задачи управления с применением новых организационных принципов рассмотрим обрабатывающий центр наклонной компоновки «СА 535» производства ОАО «Саста» (г. Сасово, Рязанская область). «СА 535» входит в гамму обрабатывающих центров наклонной компоновки и является центром с наименьшими габаритными размерами. Обрабатывающий центр «СА 535» (рисунок 5.13) оснащен: продольными (X1, X2) и поперечными (Z1, Z2) осями верхнего и нижнего суппортов, основным шпинделем и противошпинеделем (с возможностью продольного перемещения - ось Z) для обеспечения параллельной обработки. При этом две детали, закрепленные в шпинделе и противошпинделе, могут одновременно обрабатываться инструментами, установленными в нижнем и верхнем суппортах. [176-177]

Рисунок 5.13 Обрабатывающий центр наклонной компоновки «СА 535»

Оба шпинделя, установленные на токарно-фрезерном центре, являются интерполируемыми. Схема одновременной обработки двумя инструментами деталей типа тел вращения как с одной стороны, так и с обеих сторон увеличивает производительность оборудования и срок эксплуатации резцов за счет компенсации радиальных составляющих сил резания. Кинематическая схема обрабатывающих центров наклонной компоновки требует от системы ЧПУ реализации двухканального управления. Первый канал управляет верхним суппортом (оси X1, Y1),

продольной осью Z1 и шпиндельным узлом W, второй канал управляет: нижним суппортом (оси X2, Z2), продольной осью Z и противошпинделем W1.

Разработка сетевой структуры системы управления. Исходя из технического задания и предъявляемых требований, была реализована сетевая структура системы (рисунок 5.14).

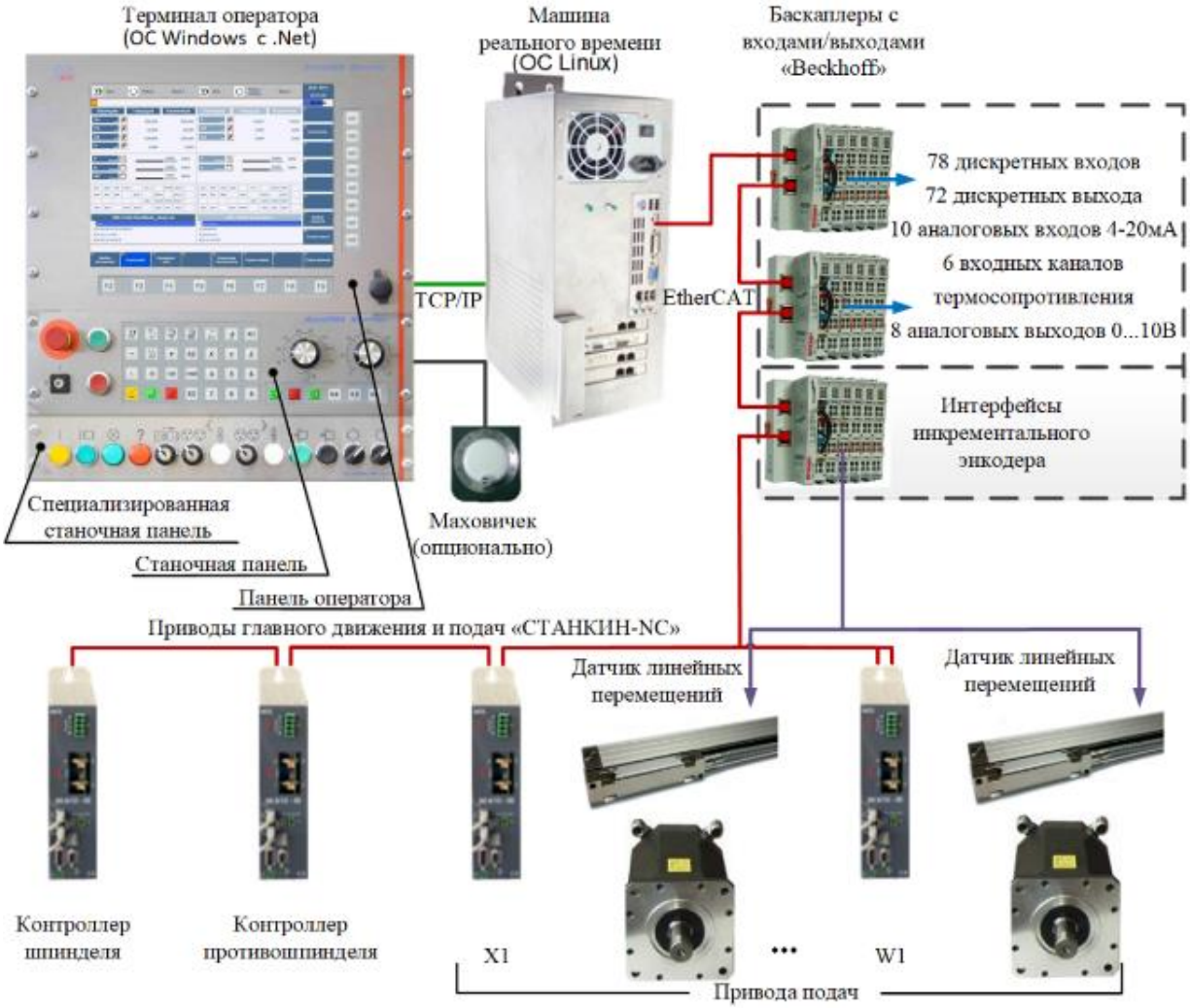


Рисунок 5.14 Сетевая структура системы управления обрабатывающим центром «СА 535»

В состав системы управления обрабатывающим центром входят: машина реального времени (функционирующая в ОС Linux) с ядром системы управления и интегрированным в неё программно-реализованным контроллером электроавтоматики типа Soft PLC; терминал оператора (состоящий из панели оператора на платформе .NET, стандартной станочной панели с опциональным подключением к ней маховичка и специализированной станочной панелью), подключенный к ядру по протоколу TCP/IP; модули аппаратных входов/выходов для подключения электроавтоматики и линейных измерительных устройств; привода главного движения и подач; контроллеры шпинделя и противошпинделя. [178] Сбор и обмен данными в сети между вычислительными устройствами осуществляется на базе открытого высокоскоростного протокола EtherCAT. Большое количество подключаемого к системе управления электрооборудования в

проекте потребовало использования трех головных модулей ввода/вывода (баскаплеры), что позволило распределить электрическую нагрузку между ними равномерно. [179]

Выделение специализированных функций и команд. Обрабатывающие центры содержат большое число технологического оборудования: револьверные головки верхнего и нижнего суппортов, зажимные патроны, система охлаждения инструмента и станка, станция охлаждения и гидростанция, транспортер стружки, защитное ограждение, система подачи воздуха, инструментальный магазин и система автоматической смены инструмента и др. Управление перечисленным набором технологического оборудования требует реализации ряда вспомогательных М-функций (таблица 5-3).

Таблица 5-3 Систематизация вспомогательных М-функций обрабатывающего центра «СА 535»

М-функция	**Назначение**
M50	Включить перемотку ленты транспортера стружки.
M51	Отключить перемотку ленты транспортера стружки.
M21	Осуществить зажим детали в гидравлическом патроне.
M22	Осуществить разжим детали в гидравлическом патроне.
M54	Закрыть защитное ограждение обрабатывающего центра.
M55	Открыть защитное ограждение обрабатывающего центра.

Составление таблицы привязки входов/выходов. Перенос выполнения управляющих программ с уровня автономных ПЛК на уровень системы ЧПУ позволил абстрагироваться от типа применяемых аппаратных модулей ввода/вывода и управляющих сигналов. В результате привязка машинного кода осуществляется не во время компиляции под конкретную платформу, а за счет конфигурирования аппаратного обеспечения в ядре системы управления на основе таблицы привязки входов/выходов (таблица 5-4).

Таблица 5-4 Привязки аппаратных входов/выходов

10 модулей EL1008 дискретных входов на 8 каналов			
1.I1	Запрос на выключение станка	2.I1	Авария оси Z1
1.I2	Станок выключен	2.I2	Авария оси Z2
1.I3	Контроль фаз	2.I3	Ограничение «+» оси X1
1.I4	Питание к шпинделю, противошпинделю и к револьверным гол.	2.I4	Ограничение «-» оси X1
1.I5	Питание к приводам подач	2.I5	«Грубый ноль» оси X1
1.I6	Авария оси X1	2.I6	Ограничение «+» оси X2
1.I7	Авария оси X2	2.I7	Ограничение «-» оси X2
1.I8	Авария оси Z	2.I8	«Грубый ноль» оси X2
3.I1	Ограничение «+» оси Z	4.I1	«Грубый ноль» оси Z2
3.I2	Ограничение «-» оси Z	4.I2	«Грубый ноль» C (шпиндель)
3.I3	«Грубый ноль» оси Z	4.I3	«Грубый ноль» C1
3.I4	Ограничение «+» оси Z1	4.I4	Предохранительная муфта
3.I5	Ограничение «-» оси Z1	4.I5	Привод вращения инструмента нижнего суппорта отключен

Продолжение таблицы 5-4.

10 модулей EL1008 дискретных входов на 8 каналов			
3.I6	«Грубый ноль» оси Z1	4.I6	Привод вращения инструмента нижнего суппорта подключен
3.I7	Ограничение «+» оси Z2	4.I7	Диск револьверной головки нижнего суппорта зафиксирован
3.I8	Ограничение «-» оси Z2	4.I8	Диск револьверной головки нижнего суппорта разфиксир.
5.I1	Ноль рев. гол. нижнего суппорта	6.I1	Ограничение «+» оси Y
5.I2	Предохранительная муфта	6.I2	Ограничение «-» оси Y
5.I3	Привод вращения инструмента верхнего суппорта отключен	6.I3	«Грубый ноль» оси Y
5.I4	Привод вращения инструмента верхнего суппорта подключен	6.I4	Ограничение хода кулачков левого патрона от центра
5.I5	Диск револьверной головки верхнего суппорта зафиксирован	6.I5	Ограничение хода кулачков левого патрон к центру
5.I6	Диск револьверной головки верхнего суппорта разфиксир.	6.I6	Контроль давления в гидросистеме левого патрона
5.I7	Нулевая позиция револьверной головки верхнего суппорта	6.I7	Тип детали левого патрона (вал/кольцо)
5.I8	Питание цепей управления	6.I8	Зажим левого патрона
7.I1	Разжим левого патрона	8.I1	Контроль уровня рабочей жидкости станции охлаждения
7.I2	Ограничение хода кулачков правого патрона от центра	8.I2	Контроль фильтра рабочей жидкости станции охлаж.
7.I3	Ограничение хода кулачков правого патрон к центру	8.I3	Контроль минимального уровня масла
7.I4	Контроль давления в гидросистеме правого патрона	8.I4	Контроль включения двигателя насоса
7.I5	Тип детали правого патрона (вал/кольцо)	8.I5	Контроль включения вентилятора теплообменника
7.I6	Зажим правого патрона	8.I6	Термостат теплообменника
7.I7	Разжим правого патрона	8.I7	Контроль включения двигателя насоса подачи СОЖ
7.I8	Контроль включения двигателей насосов станции охлаждения	8.I8	Контроль включения двигателя барабана магнитного сепаратора
9.I1	Контроль включения двигателя перемотки фильтра	10.I1	Контроль макс. уровня СОЖ в транспортере стружки
9.I2	Контроль мин. уровня СОЖ	10.I2	Контроль дверей облицовки
9.I3	Контроль макс. уровня СОЖ	10.I3	Контроль левой двери огражд.
9.I4	Контроль фильтрующего полотна	10.I4	Контроль правой двери огражд.
9.I5	Контроль включения двигателя насоса откачки СОЖ	10.I5	Контроль давления в пневмосистеме станка
9.I6	Контроль включения двигателя транспортёра стружки вперед	10.I6	Контроль включения вентиляторов шкафа управления
9.I7	Контроль включения двигателя транспортёра стружки назад	10.I7	Не используется
9.I8	Контроль минимального уровня СОЖ в транспортере стружки	10.I8	Не используется

Продолжение таблицы 5-4.

5 модулей EL2008 дискретных выходов на 8 каналов			
1.Q1	Готовность системы ЧПУ	2.Q1	Свести кулачки левого патрона
1.Q2	Отключить станок	2.Q2	Развести кулачки левого патрона
1.Q3	Вкл. зажим диска револьверной головки нижнего суппорта	2.Q3	Вкл. лампу индикации зажима левого патрона
1.Q4	Откл. зажим диска ревльв. головки нижнего суппорта	2.Q4	Свести кулачки правого патрона
1.Q5	Вкл. зажим диска револьверной головки верхнего суппорта	2.Q5	Развести кулачки правого патрона
1.Q6	Откл. зажим диска револьвер. головки верхнего суппорта	2.Q6	Вкл. лампу индикации зажима правого патрона
1.Q7	Вкл. гидротормоз оси Y	2.Q7	Вкл. электродвигатель насосов станции охлаждения
1.Q8	Откл. гидротормоз оси Y	2.Q8	Вкл. напорный клапан станции охлаждения
3.Q1	Вкл. вентилятор гидростанции	4.Q1	Включить замок левой двери
3.Q2	Вкл. двигатель насоса СОЖ	4.Q2	Включить замок правой двери
3.Q3	Вкл. двигатель перемотки фильтра	4.Q3	Управление отсечным пневмоклапаном
3.I4	Вкл. СОЖ к инструменту рев. головки верхнего суппорта	4.Q4	Управление пневмораспределителем
3.I5	Вкл. СОЖ к инструменту рев. головки нижнего суппорта	4.Q5	Отключить тормоз оси X1
3.I6	Вкл. двигатель откачки СОЖ	4.Q6	Отключить тормоз оси X2
3.I7	Вкл. транспорт. стружки вперед	4.Q7	Отключить тормоз оси Z
3.I8	Вкл. транспорт. стружки назад	4.Q8	Отключить тормоз оси Z1
5.Q1	Отключить тормоз оси Z2		
5.Q2	Отключить тормоз оси Y		
5.Q3 - 5.Q8	Не используются		
2 модуля EL3054 аналоговых входов на 4 канала (4-20 мА)			
1.AI1	Датчик температуры канал 1	2.AI1	Датчик температуры канал 5
1.AI2	Датчик температуры канал 2	2.AI2	Датчик температуры канал 6
1.AI3	Датчик температуры канал 3	2.AI3	Датчик температуры канал 7
1.AI4	Датчик температуры канал 4	2.AI4	Датчик температуры канал 8
модуль EL3052 аналоговых входов на 2 канала (4-20 мА)			
3.AI1	Датчик температуры охлаждающей жидкости в баке		
3.AI2	Не используется		
3 модуля EL3202 входных терминала для термосопротивлений на 2 канала			
1.TI1	Терморезистор шпинделя	2.TI1	Терморезистор «передних» подшипников шпинделя
1.TI2	Терморезистор противошпинделя	2.TI2	Терморезистор «задних» подшипников шпинделя
3.TI3	Терморезистор «передних» подшипников противошпинделя		
3.TI4	Терморезистор «задних» подшипников противошпинделя		
2 модуля EL4004 аналоговых выходов на 4 канала			
1.AQ1	Регулятор расхода канала 1	2.AQ1	Регулятор расхода канала 5
1.AQ2	Регулятор расхода канала 2	2.AQ2	Регулятор расхода канала 6

Продолжение таблицы 5-4.

1.AQ3	Регулятор расхода канала 3	2.AQ3	Регулятор расхода канала 7
1.AQ4	Регулятор расхода канала 4	2.AQ4	Регулятор расхода канала 8
6 модулей EL5101 интерфейсов инкрементальных энкодеров			
Подключение оптической линейки оси X1			
Подключение оптической линейки оси X2			
Подключение оптической линейки оси Z			
Подключение оптической линейки оси Z1			
Подключение оптической линейки оси Z2			
Подключение оптической линейки оси Y			
2 модуля EL5021 с интерфейсом SinCos (1Vpp) на 1 канал			
Подключение оптического датчика оси C (шпиндель)			
Подключение оптического датчика оси C1 (противошпиндель)			

Анализ таблицы привязки входов/выходов позволил установить, что баскаплеры необходимо расширить пассивными аппаратными модулями ввода/вывода данных на: 80 дискретных входов (десять 8-ми канальных модулей), 40 дискретных выходов (пять 8-ми канальных модулей), 10 аналоговых входов (два 4-х канальных модуля и один 2-х канальный модуль, 4…20 мА), 6 входных терминалов термосопротивлений (три 2-х канальных модуля), 8 аналоговых выходов (два 4-х канальных модуля, 0…10 В), 6 модулей интерфейса инкрементального энкодера, 2 модуля интерфейса SinCos (1Vpp).

Применение протокола EtherCAT для коммуникации в рамках системы управления обеспечило объединение в единую информационную сеть разнородных управляющих элементов (контроллеров приводов и компонент системы управления электроавтоматикой). Независимо от фирмы-производителя аппаратуры, поддерживающей этот международный стандарт (Beckhoff, NCT, Yaskawa и др.), программный код функционального блока остается неизменным, так же как и математическое обеспечение ядра системы управления. Изменению подвергается только диапазон разделяемой памяти, являющейся проекцией состояния физических входов/выходов и служащий буфером обмена данными между программно-реализованным контроллером и аппаратными средствами управления. Такой подход обеспечивает независимость проектирования системы управления электроавтоматикой станков от типа применяемых аппаратных средств автоматизации, что позволяет повторно использовать разработанные ранее функциональные блоки.

Настройка режимов отображения экрана оператора. Схема формирования экрана оператора в областях «Строка состояний» и «М-клавиши» представлена на рисунке 5.15. М клавиши экрана оператора используются для управления следующими командами: М2 - транспортер вперед/стоп, М3 - транспортер назад/стоп, М4 - блокировка/разблокировка нижнего ограждения, М5 - включить/отключить СОЖ, М6 - реферирование револьверной головки верхнего суппорта, М7 - реферирование револьверной головки нижнего суппорта, М8 – сброс ошибки. В

области «Строка состояний» отображаются следующие статусы узлов обрабатывающего центра: наличие/отсутствие питания на двигателях, уровень рабочих жидкостей, наличие давления в гидросистеме, наличие давления в пневмосистеме, ограждение открыто/закрыто, ошибка зажима правого и левого патронов. При этом статус нерегулярных ситуаций (приводящих к остановке работы обрабатывающего центр), отображается со знаком «?» в пиктограмме.

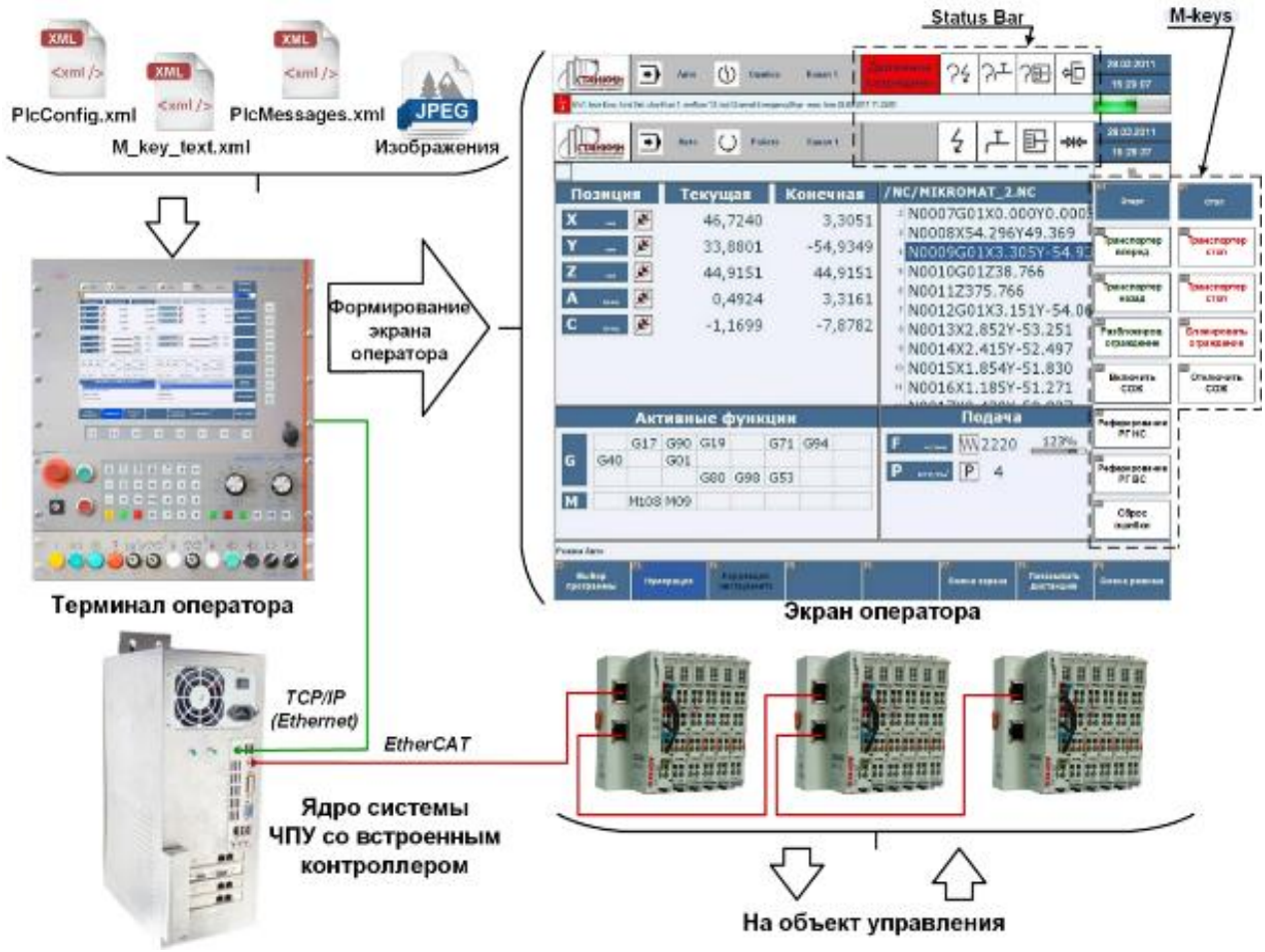


Рисунок 5.15 Схема формирования экрана оператора обрабатывающего центра «СА 535»

Разработка специализированных библиотек управления. Процесс разработки специализированных библиотек управления для обрабатывающего центра наклонной компоновки «СА 535» описан в разделе 4.4.2, в которой в качестве примера приведено программирование цикловой электроавтоматики с использованием математического аппарата временных булевых функций. [180]

5.2.4 Реализация системы логического управления электроавтоматикой вертикально-фрезерного обрабатывающего центра Quaser MV184P

Вертикально-фрезерный обрабатывающий центр с ЧПУ являются наиболее распространённым фрезерным оборудованием, используемым на современных машиностроительных предприятиях. За счет сочетания таких производственно-экономических характеристик, как: надёжность, универсальность, невысокая стоимость, простота и гибкость в управлении, - обрабатывающие центры применяются для решения широкого круга задач. Помимо этого, вертикально фрезерные обрабатывающие центры легко интегрируются в производственные линии и системы. В связи с острой необходимостью в станках и малой номенклатурой производимого в Российской Федерации оборудования многие предприятия закупали станки зарубежного производства. В Российской Федерации активно продвигается политика импортозамещения, в связи с этим предприятием ОАО «Ковровский электромеханический завод» (ОАО «КЭМЗ») была приобретена лицензия на сборку обрабатывающих центров Quaser MV184P тайваньского производства (рисунок 5.16). Приобретенная лицензия не распространяется на высокотехнологичные узлы, к которым относится система управления и электроавтоматика. В рамках совместных опытно конструкторских работ с МГТУ «СТАНКИН» была разработана специализированная система управления, в которой логическая задача решалась с применением описанных в диссертации принципов организации систем логического управления. [181-183]

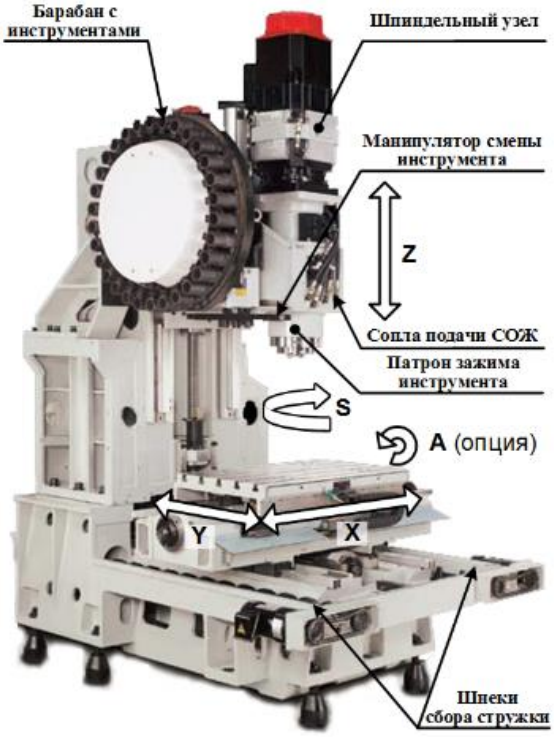


Рисунок 5.16 Вертикально фрезерный обрабатывающий центр Quaser 184P

Разработка сетевой структуры системы управления. Кинематическая схема вертикально фрезерного обрабатывающего центра Quaser 184P предполагает наличие шпинделя, трех интерполируемых осей (X, Y, Z) и оси A (опционально), отвечающей за поворотные движения рабочего стола (рисунок 5.16). Исходя из технического задания и предъявляемых требований, была разработана сетевая структура системы управления обрабатывающим центром Quaser 184P (рисунок 5.17). [184]

В состав системы управления входят: машина реального времени (функционирующая в ОС Linux) с ядром системы управления и интегрированным в неё программно-реализованным контроллером электроавтоматики типа Soft PLC; терминал оператора (состоящий из панели оператора на платформе .NET, стандартной станочной панели), подключенный к ядру по протоколу TCP/IP; модули аппаратных входов/выходов для подключения электроавтоматики и линейных измерительных устройств (баскаплеры); двигатели главного движения и подач; частотный преобразователь шпинделя.

Сбор и обмен данными в сети между вычислительными устройствами осуществляется на базе открытого высокоскоростного протокола EtherCAT, а шпиндельный узел управляется по протоколу SERCOS. Контроллеры приводов и пассивные модули входов/выходов объединены в единое EtherCAT кольцо. [185]

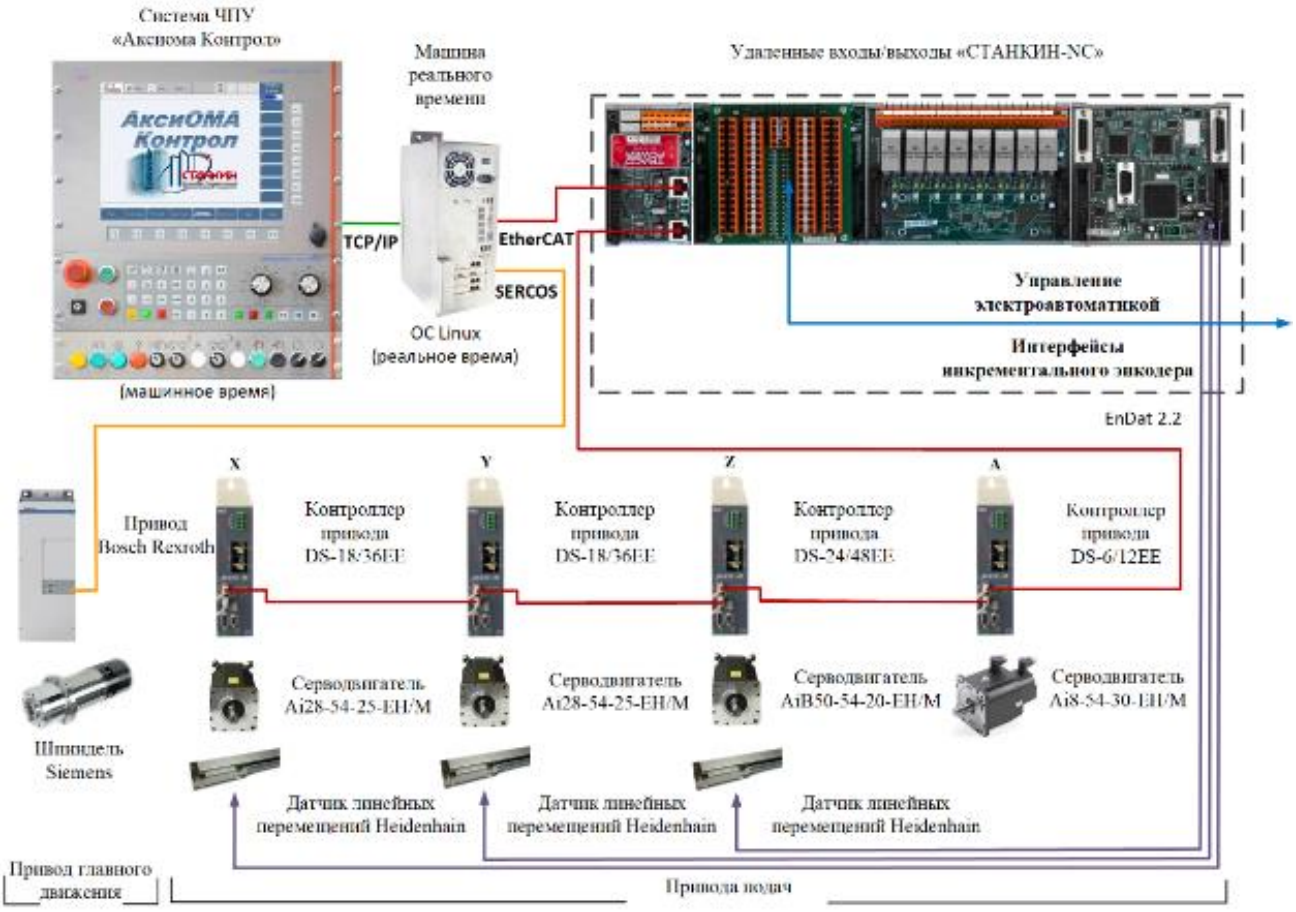


Рисунок 5.17 Сетевая структура системы управления обрабатывающим центром Quaser 184P

Выделение специализированных функций и команд. Встроенный в систему ЧПУ программно-реализованный контроллер реализует следующий функционал: управление цепями и источниками питания, обработка машинных клавиш (М-клавиш), подача смазочно-охлаждающей жидкости, автоматическая смазка направляющих и шпиндельного узла, управление механизмами удаления стружки, управление защитными ограждениями, автоматическая смена инструмента. Помимо этого для управления технологическим оборудованием необходимо реализовать целый ряд вспомогательных М-функций (таблица 5-5).

Таблица 5-5 Систематизация вспомогательных М-функций обрабатывающего центра Quaser 184P

М-функция	**Предназначение**
М3, М4, М5, М13, М14, М19	Функции запуска шпинделя по/против часовой стрелки, с и без включения СОЖ.
М7	Подача стандартной охлаждающей жидкости *(Включение режима 1 охлаждения.) помпа В.*
М8	Подача охлаждающей жидкости через шпиндель *(Включение режима 2 охлаждения) помпа С.*
М9	Прекращение подачи всех охлаждающих жидкостей. Выключает М7, М8, М10.
М10	Подача промывочной охлаждающей жидкости *помпа А.*
М708	М7+М8.
М709	М7 +М10.
М710	М8+М10.
М711	М7+М8+М10.
М16	Включение подачи воздуха через шпиндель (опция).
М17	Отключение подачи воздуха через шпиндель (опция).
М21	Фиксация инструмента на шпинделе.
М22	Освобождение инструмента на шпинделе.
М25	Поворот руки на позицию 0 градусов.
М26	Поворот руки на позицию 60 градусов.
М27	Поворот руки с позиции 0 на позицию 180 градусов (по часовой стрелке).
М28	Поворот руки с позиции 180 на позицию 0 градусов (против часовой стрелки).
М56	Включение обдувки заготовки.
М57	Отключение воздушной обдувки заготовки.

Используя М-функции можно в автоматическом режиме производить пуск и останов узлов электроавтоматики в управляющей программе [15].

Таблица привязки аппаратных входов/выходов. Головное устройство организации аппаратных входов/выходов расширено пассивными электронными модулями ввода/вывода данных, из которых: 64 дискретных входа (два 32-х канальных модуля), 40 дискретных релейных выходов (пять 8-ми канальных модуля), два модуля интерфейса инкрементального энкодера для подключения линейных датчиков. Для дальней шей разработки была сформирована таблица привязки аппартных входов/выходов(таблица 5-6).

Таблица 5-6 Привязки аппаратных входов/выходов

2 модуля дискретных входов на 32 канала			
1.I1	Авария позиции 0D	2.I1	Авария позиции 180D
1.I2	Авария позиции 60D	2.I2	Магазин инстр. в «IN»
1.I3	Авария рычага тормоза	2.I3	Магазин инстр. в «OUT»
1.I4	Датчик магазина инструментов	2.I4	Магазин инстр. в позиции смены
1.I5	Счетчик магазина инструментов	2.I5	Авария мотора конвейера стружки
1.I6	«Стакан» с инстр. внизу	2.I6	Зафиксировать ось «A»
1.I7	«Стакан» с инстр. вверху	2.I7	Разфиксировать ось «A»
1.I8	Смазка lubcon 1 включена	2.I8	Уровня масла смазки направл.
1.I9	Смазка lubcon 2 включена	2.I9	Давление системы смазки направляющих в норме
1.I10	Конц. выкл. сервисной двери	2.I10	Ограничение «-» оси X
1.I11	Ручная смена инструмента	2.I11	Ограничение «+» оси X
1.I12	Разжать инструмент	2.I12	Ограничение «-» оси Y
1.I13	Зажать инструмент	2.I13	Ограничение «+» оси Y
1.I14	Датчик нулевой позиции шпинделя	2.I14	Ограничение «-» оси Z
1.I15	Ориентация шпинделя по энкодеру	2.I15	Ограничение «+» оси Z
1.I16	Защитное ограждение открыто	2.I16	Аварийный грибок сервисного пульта нажат
1.I17	Защитное ограждение закрыто	2.I17	Нажата кнопка вращения магазина инстр. по ЧС
1.I18	Авария рычага магазина инструментов	2.I18	Нажата кнопка вращения магазина инстр. против ЧС
1.I19	Авария мотора магазина инструментов	2.I19	Нажата кнопка «открыть дверь ручной загрузки инструмента»
1.I20	Датчик уровня СОЖ	2.I20	Кнопка «открыть/закрыть фронтальную дверь»
1.I21	Контроль фильтра СОЖ	2.I21	Нажата кнопка «открыть дверь магазина смены инструментов»
1.I22	Авария левого шнека	2.I22	«Инструменты отсутствуют»
1.I23	Авария правого шнека	2.I23	Двери электрошкафа открыты
1.I24	Авария помпы A	2.I24	Не используется
1.I25	Авария помпы B	2.I25	Не используется
1.I26	Авария помпы C	2.I26	Не используется
1.I27	Уровень масла гидросистемы	2.I27	Не используется
1.I28	Вентилятор шпинделя включен	2.I28	Не используется
1.I28	БП шпинделя включен	2.I28	Не используется
1.I30	БП приводов включен	2.I30	Не используется
1.I31	Авария мотора откачки СОЖ	2.I31	Не используется
1.I32	Датчик уровня масла	2.I32	Не используется
5 модулей дискретных выходов с реле на 8 каналов			
1.Q1	Вкл./выкл. вентилятор 1 электрошкафа	2.Q1	Вкл. освещение станка
1.Q2	Вкл./выкл. вентилятор 2 электрошкафа	2.Q2	Вкл. мотор станции смазки направляющих
1.Q3	Вкл. мотор станции охлаждения шпинделя	2.Q3	Вкл. мотор конвейера стружки
1.Q4	Вкл. левый шнек	2.Q4	Вкл. мотор откачки СОЖ

Продолжение таблицы 5-6.

1.Q5	Вкл. правый шнек	2.Q5	Вкл. поворот мотора рычага инструментов по ЧС
1.Q6	Вкл. мотор помпы «A»	2.Q6	Вкл. поворот мотора рычага инструментов против ЧС
1.Q7	Вкл. мотор помпы «B»	2.Q7	Повернуть мотор магазина смены инструментов по ЧС
1.Q8	Вкл. мотор помпы «C»	2.Q8	Повернуть мотор магазина смены инструментов против ЧС
3.Q1	Открыть дверь облицовки	4.Q1	Вкл. индикатор «ЗЕЛЕНЫЙ»
3.Q2	Повернуть «стакан» с инструментом вверх	4.Q2	Вкл. индикатор «ЖЕЛТЫЙ»
3.Q3	Повернуть «стакан» с инструментом вниз	4.Q3	Вкл. индикатор «КРАСНЫЙ»
3.Q4	Вкл. электромагнит фиксации магазина инструментов	4.Q4	Зажать ось «A»
3.Q5	Вкл./выкл. тормоз оси «Z»	4.Q5	Вкл. смазку направляющих
3.Q6	Вкл. продувку инструмента	4.Q6	Разжать инструмент
3.Q7	Отключить общее питание	4.Q7	Сдув стружки в рабочей зоне
3.Q8	Не используется	4.Q8	Не используется
5.Q1	Открыть дверь магазина инструментов		
5.Q2	Открыть фронтальную дверь станка		
5.Q3	Продувка инструмента		
5.Q4	Не используется		
5.Q5	Не используется		
5.Q6	Не используется		
5.Q7	Не используется		
5.Q8	Не используется		
2 модуля интерфейса EnDat 2.2 на 2 канала			
1.E1	Ввод данных с датчика линейных перемещений оси X		
1.E2	Ввод данных с датчика линейных перемещений оси Y		
2.E1	Ввод данных с датчика линейных перемещений оси Z		
2.E1	Не используется		

Настройка режимов отображения экрана оператора. Изображение в областях «Строка состояний» и «М-клавиш» экрана оператора формируются динамически (рисунок 5.18). Вертикальная свободно программируемая панель М-клавиш предназначена для ручного запуска узлов электроавтоматики оператором. Для управления обрабатывающим центром был реализован следующий набор М-клавиш: подача промывочной охлаждающей жидкости (помпа A), подача стандартной охлаждающей жидкости (помпа B), подача охлаждающей жидкости через шпиндель (помпа C), включение шнеков удаления стружки, управление освещением рабочей зоны, включение обдува заготовки, управление разблокировкой дверей, включение реверса шнеков сбора стружки. Пиктограммы в области «Строка состояний» отображают статусы узлов обрабатывающего центра, при этом нерегулярные ситуации обозначены знаком «?». [186]

Разработка специализированных библиотек управления. Для каждого из выделенных узлов в среде разработки программ электроавтоматики Soft PLC контроллера на языке функциональных блоков были разработаны специализированные пользовательские блоки – библиотеки [14]. Базовый блок управления работой станка - блок управления цепями и источниками питания. Основная задача блока - контроль за работой электротехнических элементов станка, управление подачей силового и низковольтного питания и обработка сигналов с клавиш и кнопок панели оператора. В блоке «PowerControl» производится обработка сигналов от элементов управления и считывание сигналов об аварийных ситуациях. На выходы блока выдаются номера предупреждений и ошибок, передаваемые в систему ЧПУ для отображения их оператору, а также сигналы на запрет/разрешение работы канала системы ЧПУ.

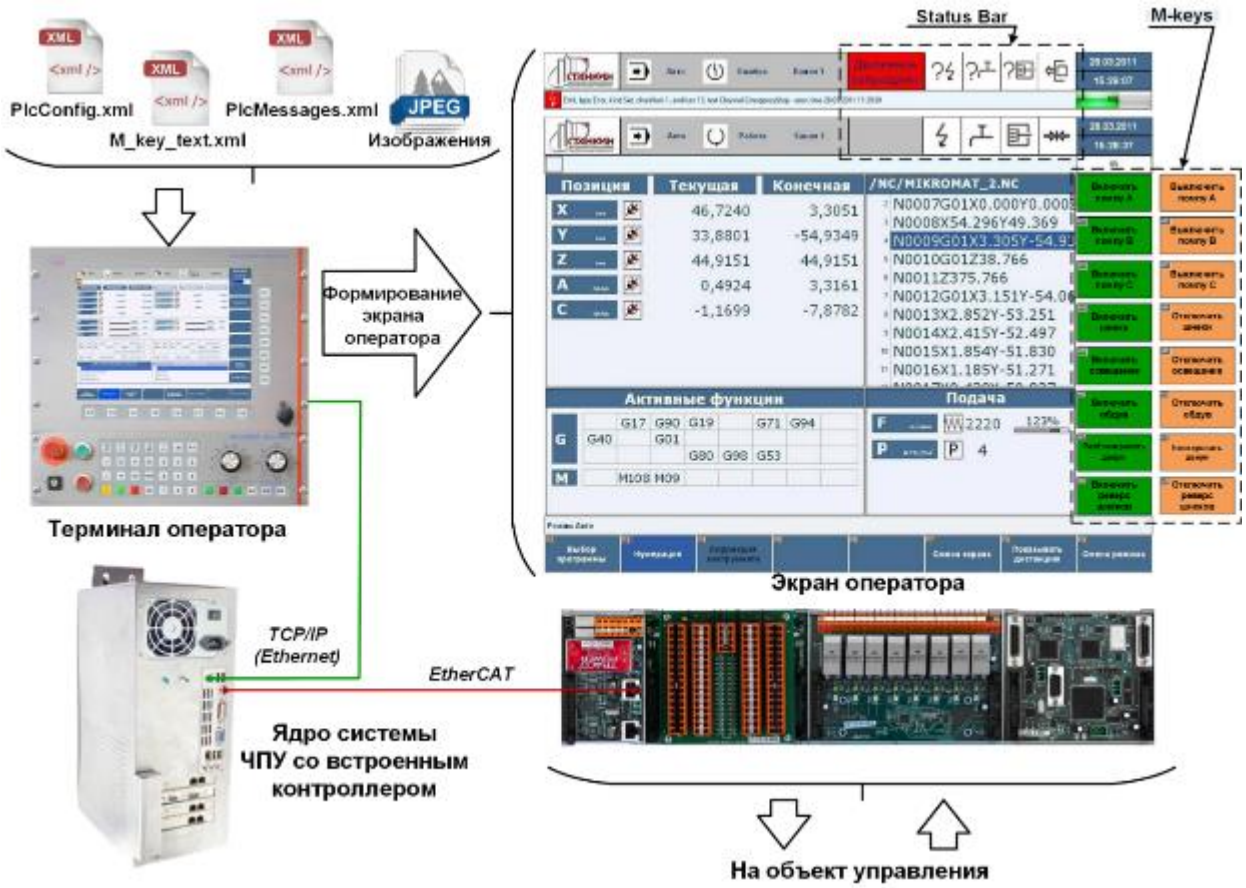


Рисунок 5.18 Схема формирования экрана оператора обрабатывающего центра Quaser 184P

Для обеспечения смазочных функций, охлаждения заготовки и режущего инструмента в фрезерном центре предусмотрена возможность подачи трех видов СОЖ: стандартная, промывочная, и подаваемая через шпиндель, для чего были созданы пользовательские библиотеки «CoolantSpindle» и «PompAB» (рисунок 5.19). Промывочная СОЖ подается через специальные сопла, установленные на шпинделе и направленные в зону резания. Стандартная СОЖ подается через дополнительное сопло, которое оператор может направлять в различные зоны по своему усмотрению. СОЖ через шпиндель предназначена для использования со

специальными инструментами, имеющими сквозное отверстие, проходящее вдоль всего инструмента. Для подачи каждого вида СОЖ имеется независимая помпа, осуществляющая накачку охлаждающей жидкости из общего бака.

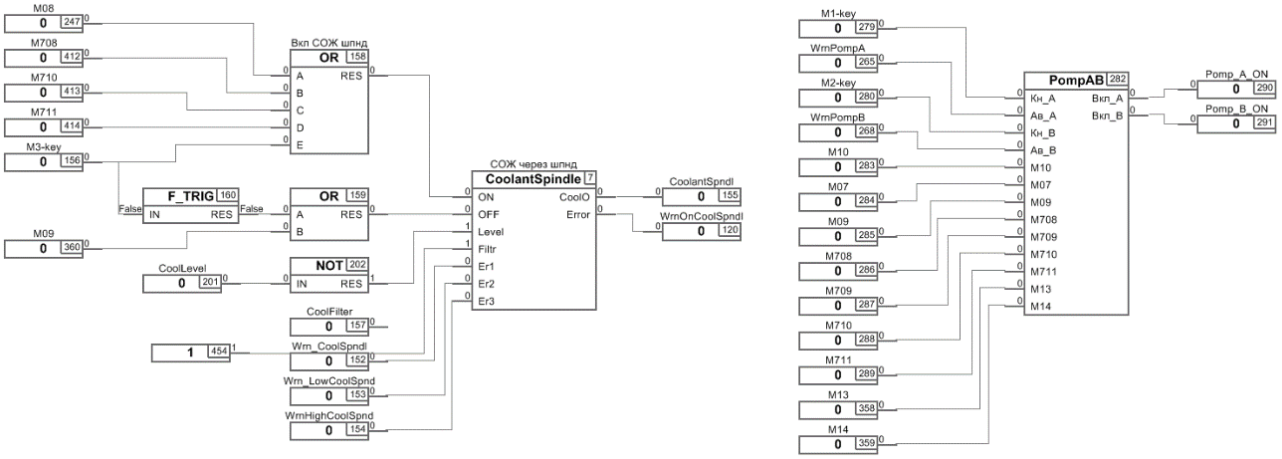


Рисунок 5.19. Функциональный блок управления подачей СОЖ

Подача смазки на направляющие осей станка производится непрерывно, после получения сигнала о готовности к работе. Подача смазки в шпиндель производится при получении сигнала от интерфейса ЧПУ-ПЛК о выходе шпинделя в рабочее состояние. [17] Механизм удаления стружки в фрезерном центре Quaser MV 184P включает два основных компонента, управляемых с помощью электроавтоматики: шнеки удаления стружки и конвейер стружки. Шнеки необходимы для отвода стружки непосредственно из зоны резания, конвейер - для отвода стружки от станка в резервуар. Шнеки удаления стружки могут быть включены как автоматически в управляющей программе ЧПУ, так и вручную, с помощью М-клавиши (рисунок 5.20).

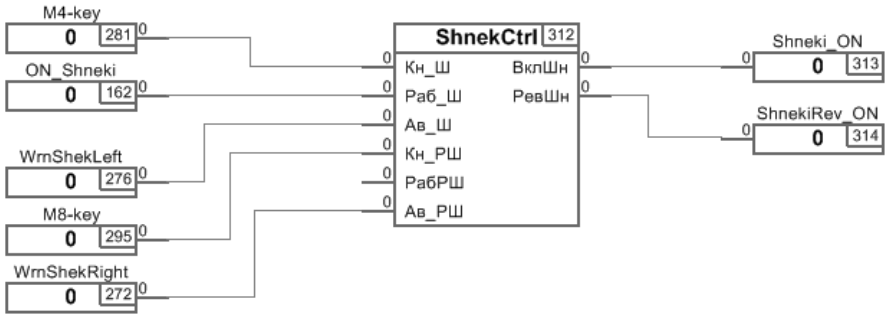


Рисунок 5.20. Функциональный блок управления шнеками стружки

Для предотвращения засоров каналов отвода стружки имеется возможность запустить реверс шнеков. Во время работы шнеков производится контроль за корректной работой электродвигателей, осуществляющих вращение. Конвейер стружки запускается вместе со шнеками удаления стружки.

Управление защитными ограждениями включает в себя контроль открытия и управление электромагнитными замками двух основных дверей станка: фронтальной двери (закрывает зону обработки) и двери доступа к инструментальному магазину. Для реализации логики управления ограждениями был разработан пользовательский блок «DoorsControl» (рисунок 5.21). В соответствии с техникой безопасности, разблокировка дверей возможна в двух ситуациях: при ручной смене инструмента (вызове команды M66) или в режиме ручного управления (Jog), при условии, что шпиндель остановлен. Разблокировка дверей производится по нажатию на M-клавишу.

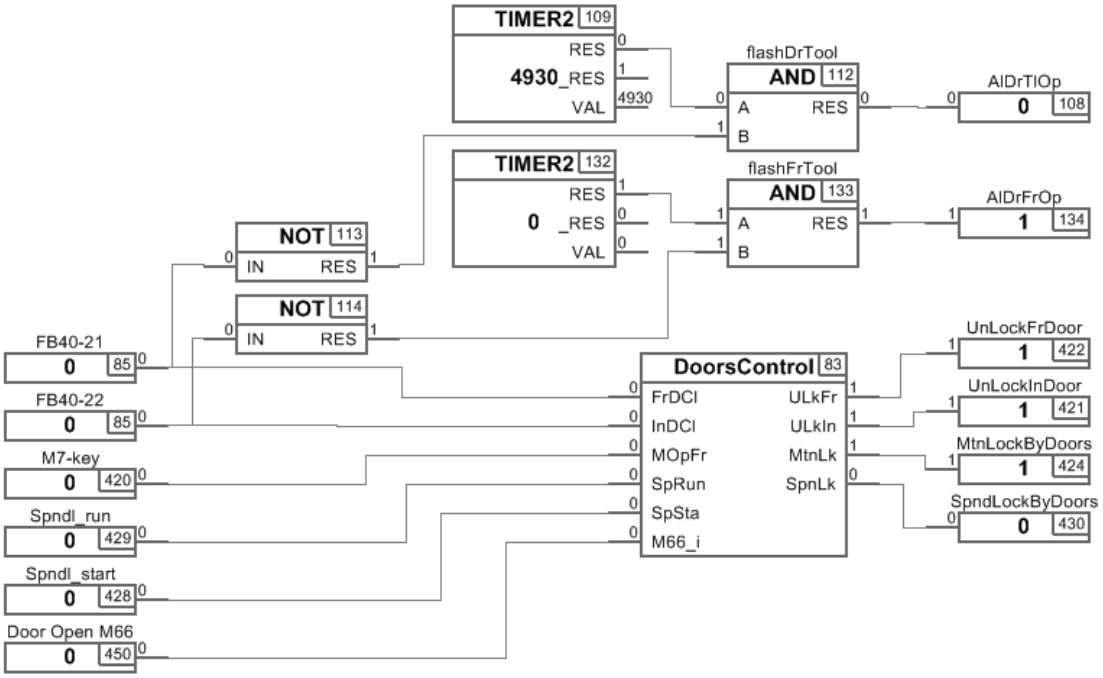


Рисунок 5.21. Функциональный блок управления защитными ограждениями

Важным условием является запрет работы при переходе в другие режимы до закрытия обоих дверей. Управление механизмом автоматической смены инструмента является наиболее сложным логическим блоком из всех органов управления фрезерным центром. Смена инструмента осуществляется при помощи механизма автоматической смены (рисунок 5.22). [187] Механизм состоит из следующих элементов:

- Инструментальный магазин – барабан с 32-мя инструментальными карманами. Смена инструмента происходит между патроном шпинделя и нижним карманом барабана. Вращение барабана осуществляется при помощи электродвигателя, управляемого сигналами O1.1 (вращение по часовой стрелке) и O1.2 (вращение против часовой стрелки). Выход нужного кармана в нижнее положение для смены инструмента контролируется датчиком I2.1. Опускание и подъем кармана осуществляется пневмоцилиндром. Если сигнал O1.5 установлен, происходит

опускание кармана, при снятии сигнала карман поднимается. Положение кармана контролируется датчиком (сигнал I2.2 - карман опущен, I2.3 - карман поднят).

- Рычаг смены инструмента – осуществляет захват инструмента в патроне шпинделя и стакане барабана и меняет их местами. Управление рычагом осуществляется подачей сигнала O1.3 на привод электродвигателя рычага. Механический привод рычага обеспечивает следующий цикл движения: выход в положение «90 градусов» для захвата инструмента (сигналом I2.4), выдвижение рычага с разворотом на 180 градусов и его отвод для перестановки инструментов местами (сигналом I2.5), поворот в исходное положение (сигналом I2.6).
- Патрон шпинделя – зажимает инструмент. Разжим патрона осуществляется подачей сигнала O0.1 на пневмоцилиндр. Инструмент может быть вставлен в патрон только при определенном угле поворота шпинделя. Сигнал датчика I2.0 контролирует поворот патрона в позицию смены инструмента, сигнал I2.7 показывает, что патрон разжат.

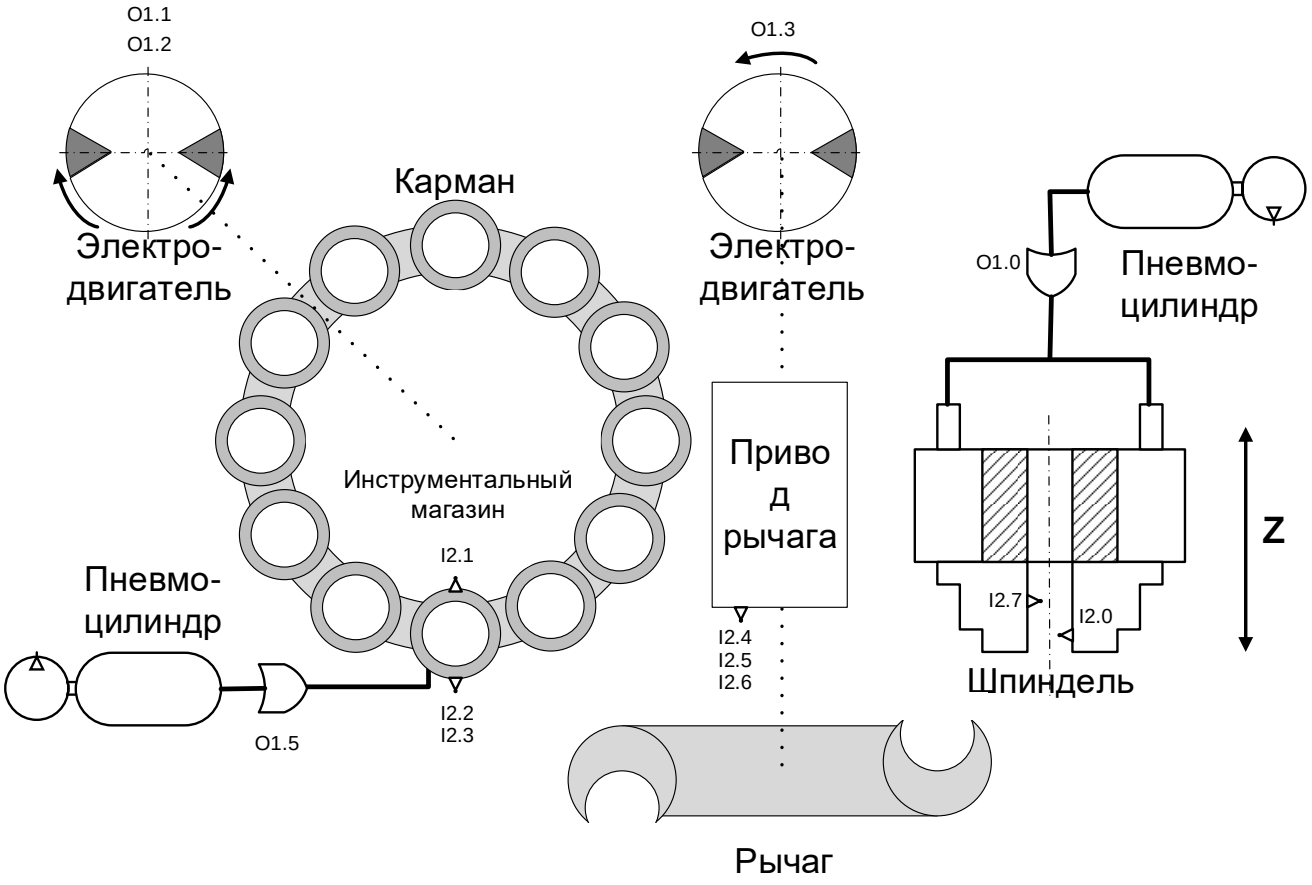


Рисунок 5.22. Схема механизма автоматической смены инструмента

5.3 Выводы

1. Модульный принцип построения систем логического управления позволяет реализовывать как автономные решения, используемые для управления отдельными технологическими узлами или процессами, так и решения, входящие в составе комплекса систем управления (например, в рамках системы ЧПУ).
2. Предложенные в работе теоретические аспекты и разработанные принципы построения систем логического управления были проверены в ходе экспериментальных стендовых испытаний с подключением различных типов объектов управления, в ходе которых удалось установить, что система управления работоспособна как автономная единица и способна самостоятельно управлять технологическими объектами без привлечения дополнительных аппаратных и программных ресурсов.
3. Для организации работы технологических объектов, содержащих в своем составе несколько узлов, с автономными системами логического управления предложен вариант их интеграции в единую программно-аппаратную среду управления в рамках концепции «ведущий-ведомый», когда одна из систем управления выступает «ведущей» и формирует набор команд для «ведомой» системы.
4. Применение системы логического управления установкой УГСР совместно со специализированной технологией обработки позволило контролировать и экономить расход абразивного материала (от 25 до 30 %), без ухудшения качественных характеристик поверхности реза.
5. Предложен механизм, позволяющий распределить электрическую нагрузку в цепи ввода/вывода за счет увеличения количества головных аппаратных модулей. Возрастание электрической нагрузки связано со значительным увеличением количества и сложности объектов управления и, как следствие, обрабатываемых СЛУ сигналов.
6. Выделение в системе управления отдельного уровня поддержки промышленных протоколов связи позволило реализовать мультипротокольную систему управления с поддержкой ряда высокоскоростных протоколов на базе технологии Ethernet, в которой реализована возможность интеграции комплектующих, использующих разнородные промышленные сети, в единую распределенную систему логического управления.

Заключение

Общие выводы и результаты

1. В работе решена актуальная научная проблема, имеющая важное значение для развития промышленности РФ - создание принципов и методологических основ построения кроссплатформенных, мультипротокольных систем логического управления технологическим оборудованием, реализованных по модульному принципу, имеющих открытую архитектуру и поддерживающих работу с широким классом аппаратных устройств.
2. Установлены взаимосвязи между характеристиками технологического оборудования и задачами, функциями, параметрами систем логического управления технологическим оборудованием, влияющими на структуру системы и определяющими состав программно-аппаратных модулей логических контроллеров как основного инструментария автоматизации.
3. Разработаны формальные модели проектирования систем управления, объединенные в логическую схему последовательного их создания и трансформации. Модели каждого уровня описывают определенные аспекты работы системы логического управления и позволяют: определить основные функции системы, выявить зависимость между платформой и прикладной компонентой, систематизировать основные потоки данных, определить структуру системы управления, формализовать этапы проектирования программ логического управления, определить схему функционирования системы в условиях распределения вычислительных ресурсов.
4. Подход к модульной структуре системы логического управления, предложенный в работе, предполагает выделение модулей, имеющих структурное единообразие и общий механизм связей, что позволяет: организовать систему управления как в рамках единой платформы, так и на нескольких платформах, путем разделения группы модулей по единому функциональному признаку; комплектовать систему в зависимости от реализуемой задачи; использовать программные и аппаратные компоненты различных производителей.
5. Предложенная модульная структура позволяет разделить систему логического управления на две подсистемы: программирования, работающая в режиме машинного времени, и исполнительное ядро, работающее в режиме реального времени. Взаимодействие между двумя подсистемами на физическом уровне реализовано на базе Ethernet, на программном уровне - на базе технологии сокетов.
6. Разработанные методологические аспекты позволяют предложить комплексное решение проблемы проектирования и реализации систем логического управления. Каждая фаза процесса разработки формирует конкретное решение, набор которых охватывает весь последовательный процесс проектирования системы.

7. Сформирован набор формализованных математических методов, который позволяет производить проектирование программ логического управления любой сложности с использованием реализованной в работе среды проектирования.

8. Проведено комплексное тестирование систем логического управления, которое включает: нагрузочное тестирование, тестирование на отказ и проведение приемо-сдаточных испытаний. Указанные виды тестовых испытаний позволяют сделать выводы о возможности эксплуатации разрабатываемой системы управления.

9. Реализованы системы логического управления реальными технологическими объектами с использованием разработанной методики проектирования, позволившие проверить возможности, заложенные в новых принципах построения систем управления.

10. Полученные результаты рекомендуется использовать при создании систем управления технологическим оборудованием для машиностроительных предприятий и в учебном процессе при подготовке инженерно-технических и научно-педагогических кадров по направлению «Автоматизация и управление технологическими процессами и производствами».

Список сокращений и условных обозначений

АСТПП - автоматизированная система технической подготовкой производства;

АСУ – автоматизированная система управления;

АСУП - автоматизированная система управления производствами;

АСУТП - автоматизированная система управления технологическими процессами;

ВБФ - временные Булевы функции;

МЭК / IEC – международная электротехническая комиссия / International Electrotechnical Commission;

ОЗУ – оперативное запоминающее устройство;

ОС – операционная система;

ОСРВ – операционная система реального времени;

ПАЗ - противоаварийная защита;

ПК – персональный компьютера;

ПЛК / PLC – программируемый логический контроллер / programmable logical controller;

ПО - программное обеспечение;

РКС/LD - релейно-контактные схемы / ladder diagram;

СВД - станция высокого давления.

СОЖ - смазочно-охлаждающей жидкостью;

УГСР - Установка Гидро-Струйной Резки;

ФЦП – федеральная целевая программа;

ЧПУ – числовое программное управление;

ШИМ – широтно-импульсный модулятор;

ЭВМ – электронно-вычислительная машина;

API – Application Program Interface;

CFC - Continuous Function Chart;

COM - Component Object Model;

DCOM - Distributed COM;

DFD - Data Flow Diagrams;

DLL - Dynamic Link Library;

FBD – functional block diagram;

GUI - Graphical user interface;

IDEF – ICAM Definition (Integrated Computer Aided Manufacturing – интегрированное автоматизированное производство);

IL – instruction list;

IoT - Internet of Things;

ISO - International Organization for Standardization;

IT – information technology;

MFC - Microsoft Foundation Classes;

MVC - Model View Controller;

OEM - original equipment manufacturer;

OPC - Open Platform Communications (ранее OLE for Process Control);

OPC UA - OPC Unified Architecture;

PAC - Programmable Automation Controller;

PCNC - Personal Computer Numerical Control;

POSIX - Portable Operating System Interface;

RTX - Real Time eXtension;

SCADA - Supervisory Control And Data Acquisition

SQL - Structured Query Language;

ST – structured text;

UML - Unified Modeling Language;

W3C - World Wide Web Consortium;

XML - eXtensible Markup Language;

XSD - XML Schema Definition Language.

Список литературы

1. Официальный сайт маркетинговой компании HIS Markits [Электронный ресурс]. - Исследования рынка программируемых логических контроллеров - Режим доступа: https://technology.ihs.com/599840/ (дата обращения 15.01.2019)

2. Сосонкин, В.Л. Принцип подчиненного управления в логических системах управления / В.Л. Сосонкин, В.И. Клепиков // Приборы и системы управления. – 1995.- №12. - с.16-18.

3. Сосонкин, В.Л. Концепция управления участком механообработки с различным по уровню автоматизации оборудованием / В.Л. Сосонкин // Вестник машиностроения. - 1991. - №9. - С. 28-30.

4. Сосонкин, В.Л. Концепция числового программного управления мехатронными системами: реализация логической задачи / В.Л. Сосонкин, Г.М. Мартинов // Мехатроника. – 2001.- №2.- С. 3-7.

5. Сосонкин, В.Л. Программное управление технологическим оборудованием: учебник для вузов / В.Л. Сосонкин. – М.: Машиностроение, 1991. - 509 с.

6. Сосонкин, В.Л. Концепция числового программного управления мехатронными системами: проблемы управления электроавтоматикой / В.Л. Сосонкин, Г.М. Мартинов // Автотракторное электрооборудование. - 2002. - №4. - с. 33-42.

7. Сосонкин, В.Л. Концепция числового программного управления мехатронными системами: управление электроавтоматикой станков с ЧПУ по типу виртуальных контроллеров SoftPLC / В.Л. Сосонкин, Г.М. Мартинов, М.М. Перепелкина // Приборы и системы. Управление, контроль, диагностика. - 2003. - №7. - с. 5-10.

8. Сосонкин, В.Л. Программирование систем числового программного управления: учебное пособие / В.Л. Сосонкин, Г.М. Мартинов. – М. Логос, 2008. – 344 с.

9. Сосонкин, В.Л. Системы числового программного управления: учебное пособие / В.Л. Сосонкин, Г.М. Мартинов. – М. Логос, 2005. – 296 с.

10. Соломенцев, Ю.М. Построение персональных систем ЧПУ (PCNC) по типу открытых систем управления / Ю.М. Соломенцев, В.Л. Сосонкин, Г.М. Мартинов // Информационные технологии и вычислительные системы. - 1997. - №3. - с.68-76.

11. Соломенцев, Ю.М. Управление гибкими производственными системами: учебник / Ю.М. Соломенцев, В.Л. Сосонкин // М.: Машиностроение, 1988. - с.352.

12. Соломенцев, Ю.М. Проблема создания компьютеризированных интегрированных производств / Ю.М. Соломенцев // Автоматизация проектирования. - 1997. - №1. - с.10-14.

13. Мартинов, Г. М. Принципы интеграции компонентов электроавтоматики на примере 3D-сцены визуализации / Г. М. Мартинов, Н. В. Козак // Системы управления и информационные технологии. - 2007. - №2 (28). - с. 88-92.

14. Мартинов, Г.М. Программируемые контроллеры автоматизации PAC (Programmable Automation Controller). Эволюция, проблемы, тенденции развития / Г.М. Мартинов, Л.И. Мартинова // Стружка. - 2007. - №4. - с. 22–25.

15. Юдицкий, С. А. Логическое управление дискретными процессами / С.А. Юдицкий, В. З. Магергут – Москва: Машиностроение, 1987.

16. Юдицкий, С. А. Операционно-целевое моделирование динамики развития организационных систем средствами сетей Петри / С. А. Юдицкий // Автоматика и телемеханика. – 2008. - № 1. - с. 114–123.

17. Клуг, А. Ю. Модель взаимодействия параллельных асинхронных дискретных процессов / А. Ю. Клуг, С. А. Юдицкий, Автоматика и телемеханика. – 1991. - № 3. - с. 133–142.

18. Авакян, В. В. Описание и анализ параллельных алгоритмов логического управления с применением графов операции / В. В. Авакян, С. А. Юдицкий // Автоматика и телемеханика, - 1990, - № 1, - с. 100–108.

19. Юдицкий, С. А. Конвейерные дискретные процессы и сети Петри / С. А. Юдицкий // Автоматика и телемеханика, - 1983, - № 6, - с. 141–147.

20. Бунько, Е. Б. Программная реализация сетей Петри в асинхронных устройствах логического управления / Е. Б. Бунько, С. А. Юдицкий // Автоматика и телемеханика, - 1983, - № 3, - с. 109–119.

21. Гаврилов, М. А. Теория релейно-контактных схем / М.А. Гаврилов // М. -Л.: Издательство АН СССР, 1950.

22. Гаврилов, М. А. Абстрактная и структурная теория релейных устройств / М. А. Гаврилов, В.М. Копыленко // Москва: Наука, 1972.

23. Грейнер, Г. Р. Проектирование бесконтактных управляющих логических устройств промышленной автоматики // Г. Р. Грейнер, В. П. Ильяшенко, В. П. Май и др. – Москва: Энергия, 1977.

24. Петров, И.В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования // И.В. Петров — Москва: СОЛОН-Пресс, 2004. — 256 с.

25. Петров, И.В. Программируемые контроллеры. Практическое применение языков стандарта МЭК 61131-3 / И.В. Петров // М.: СОЛОН-Пресс, 2003. 256 с.

26. Аршанский, М.М. Объектно-ориентированный подход к привязкам как основа иерархического управления мехатронными объектами / М.М. Аршанский, Ганюшин Р.С. // Мехатроника, автоматизация, управление. – 2007. - № 7. - с. 24-28.

27. Титаренко, Ю.И. Индустриальное проектирование автоматизированных систем управления технологическими процессами на базе семейства виртуальных контроллеров: автореф. дис. … д-ра тех. наук: 05.13.07 / Титаренко Юрий Иванович - Томск, 1995. - 43 с.

28. Модели, методы и средства проектирования распределенных компонентно-базированных информационно-управляющих систем промышленной автоматики: автореферат дис. … д- ра тех. наук: 05.13.17, 05.13.05 / Дубинин Виктор Николаевич. - Пенза, 2014. - 34 с.

29. Олссон, Г. Цифровые системы автоматизации и управления / Олссон Г., Пиани Д. - 3-е издание перераб. и доп. — СПб.: Невский Диалект, 2001. — 557 с.: ил.

30. Agustin Rullan, Programmable Logical Controllers versus Personal Computers for Process Control, Computers ind. Engng Vol. 33, Nos 1-2, pp. 421-424, 1997, Elsevier Science Ltd, GB

31. Parr, E. A. Programmable Controllers: An Engineer's Guide // E. A. Parr - Third Edition, Publisher: Newnes, Oxford, 2003, p. 448.

32. Frank, D. Petruzella: Programmable Logic Controllers, 3rd edition 2005 copyright ISBN: 0-07-829852-0

33. Loose Leaf for Electric Motors and Control Systems (2nd Edition), by Frank Petruzella, Loose Leaf, 320 Pages, Published 2015

34. Liang, Q. The Development of Soft PLC Running and Simulation System Based on RTX / Liang, Quan, Su Donghai, Wang, Jie, Wang Wang Yemu // Manufacturing process and equipment, pts 1-4, - 4th International Conference on Manufacturing Science and Engineering (ICMSE 2013). - Том: 694-697. - стр.: 2660-2666

35. Liang, Q. The Study of Soft PLC Running System / Liang Quan, Li Li // International Conference on Advanced in Control Engineering and Information Science (CEIS). - Volume 15. – 2011.- p. 1234-1238.

36. Чернецкий, В. М. Процессно-ориентированная концепция системного моделирования АСУ: автореф. дис. … д-ра тех. наук: 05.13.06 / Чернецкий Валерий Михайлович. - Москва, 2000. - 34 с.

37. Зотов, И. В. Теоретические основы синтеза схем быстродействующих устройств распределенной децентрализованной координации параллельных микропрограмм в мультиконтроллерах: дис. … д-ра тех. наук: 05.13.05 / Зотов Игорь Валерьевич. - Курск, 2006. - 383 с.

38. Чебурахин, И. Ф. Математическое моделирование и синтез вычислительных и управляющих логических устройств: автореф. дис. … д-ра тех. наук: 05.13.18 / Чебурахин Игорь Федорович. - Москва, 2004. - 34 с.

39. Шалыто, А. А. Методы аппаратной и программной реализации алгоритмов логического управления технологическими процессами: автореф. дис. д-ра … тех. наук: 05.13.05: Шалыто Анатолий Абрамович. - Санкт-Петербург, 1999. - 32 с.

40. Фокин, А. Л. Синтез систем автоматического управления технологическими процессами по расширенной модели динамики объекта: автореф. дис. д-ра … тех. наук: 05.13.06: Фокин, Александр Леонидович / С.-Петерб. гос. технол. ин-т. - Санкт-Петербург, 2002. - 40 с.

41. Харазов, В.Г. Интегрированные системы управления технологическими процессами / В.Г. Харазов // СпБ.: Профессия, 2009. - 592 с.

42. Лескин, А.А. Алгебраические модели гибких производственных систем / А.А. Лескин, П.А.Мальцев, А.М. Спиридонов // Л.: Наука, 1986. – 150 с.

43. Шилин, А.А. Моделирование нелинейных систем на FBD-блоках с ограниченным базисом / А.А. Шилин, В.Г. Букреев, Е.И. Гладышева // Доклады Томского государственного университета систем управления и радиоэлектроники. - №1 (27). - 2013. – с. 107-113.

44. Kandray, D. Programmable Automation Technologies, New York:Industrial Press,2010.-405 p.

45. Bolton, W. Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering (Paperback). - London: 2003. 592p.

46. Денисенко, В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием / В.В. Денисенко, М.: Горячая линия-Телеком, 2009. —608 с.

47. Денисенко, В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием / В.В. Денисенко, Учебное пособие. - Ставрополь: АГРУС, 2009. - 100 с.

48. Минаев, И. Г. Программируемые логические контроллеры. Практическое руководство для начинающего инженера. / И. Г. Минаев, В. В. Самойленко — Ставрополь: АГРУС, 2009. — 100 с.

49. Минаев, И. Г. Программируемые логические контроллеры в автоматизированных системах управления / И. Г. Минаев, В. М. Шарапов, В. В. Самойленко, Д. Г. Ушкур. 2-е изд., перераб. и доп. — Ставрополь: АГРУС, 2010. — 128 с.

50. Минаев, И.Г. Свободно программируемые устройства в автоматизированных системах управления / И.Г. Минаев, В.В. Самойленко, Д.Г. Ушкур, И.В. Федоренко - Ставрополь: АГРУС. 2016. - 168 с.

51. Официальный сайт русскоязычной версии журнала «Control Engineering» [электронный ресурс]: Официальный сайт русскоязычной версии журнала «Control Engineering» - Режим доступа: http://www.controlengrussia.com/ (дата обращения 16.10.2017).

52. Официальный сайт журнала «Control Engineering» [электронный ресурс]: Официальный сайт журнала «Control Engineering» - Режим доступа: http://www.controleng.com (дата обращения 16.10.2017).

53. ГОСТ Р МЭК 61508-1-2007 Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. – М. : Стандартинформ, 2008. – 58 с.

54. Официальный ресурс торговой ассоциации SERCOS [электронный ресурс]: Официальный ресурс торговой ассоциации SERCOS – Режим доступа: http://www.sercos.com/ (дата обращения 16.10.2017).

55. Официальная web-страница коммуникационного интерфейса SERCOS [электронный ресурс]: Официальный ресурс компании Bosch-Rexroth – Режим доступа: https://www.boschrexroth.com/en/xc/products/engineering/communication-interfaces/sercos/index (дата обращения 16.10.2017).

56. Официальный портал ассоциации EtherCAT-разработчиков [электронный ресурс]: Официальный портал ассоциации EtherCAT-разработчиков – Режим доступа: https://www.ethercat.org/default.htm (дата обращения 16.10.2017).

57. Официальная web-страница коммуникационного интерфейса EtherCAT [электронный ресурс]: Официальный ресурс компании Beckhoff – Режим доступа: http://www.beckhoff.ru/ru/default.htm?ethercat/default.htm (дата обращения 16.10.2017).

58. Официальный портал ассоциации Modbus-разработчиков [электронный ресурс]: Официальный портал ассоциации Modbus-разработчиков – Режим доступа: http://www.modbus.org/ (дата обращения 16.10.2017).

59. Официальный сайт компании «Wind River Systems, Inc.» [электронный ресурс]: официальный сайт компании «Wind River Systems, Inc.» - Режим доступа: http://www.rtlinuxfree.com/ (дата обращения 16.10.2017).

60. Web-страница проекта «ART-Linux» [электронный ресурс]: web-страница проекта «ART-Linux» - Режим доступа: http://art-linux.sourceforge.net/ (дата обращения 16.10.2017).

61. Официальная страница операционной системы Windows Embedded [электронный ресурс]: официальный сайт корпорации Microsoft – Режим доступа: https://www.microsoft.com/windowsembedded/en-us/windows-embedded.aspx (дата обращения 16.10.2017).

62. Официальная страница интегрированной среды разработки Visual Studio [электронный ресурс]: официальный сайт корпорации Microsoft – Режим доступа: https://www.visualstudio.com (дата обращения 16.10.2017).

63. Официальный сайт разработчиков интегрированной среды разработки Eclipse [электронный ресурс]: официальный сайт компании «The Eclipse Foundation» – Режим доступа: http://www.eclipse.org/ (дата обращения 16.10.2017).

64. Официальный ресурс свободно распространяемой интегрированной среды разработки «Code Blocks» [электронный ресурс]: Официальный ресурс свободно распространяемой интегрированной среды разработки «Code Blocks» - Режим доступа: http://www.codeblocks.org/ (дата обращения 16.10.2017).

65. Официальный сайт консорциума «OPC Foundation» [электронный ресурс]: официальная сайт консорциума «OPC Foundation» - Режим доступа: https://opcfoundation.org/ (дата обращения 15.01.2019).

66. Basset, E.W. Real-Time PC Control / E.W. Basset // Industrial Computing, - vol. 15. - no. 2. – 1995. - p. 14-17.

67. Gee, D. The Benefits of Personal Computer Based Control Systems / Gee D. //Proceedings of IPC'95 Conference. - May 8-11 – 1995. - pp. 111-120.

68. Gee, D. PC or PLC, What's in a Name / Gee D. // Industrial Computing. - vol.14. - no.7. – 1995. - pp. 24-26.

69. Hohmann, T. Why PCs Won't Kill PLCs / T. Hohmann // Industrial Computing. - vol. 15. - no. 10. – 1996. - pp. 6-11.

70. Аналитический консультационный центр в области промышленности «ARC Advisory Group» [электронный ресурс]: аналитический консультационный центр в области промышленности «ARC Advisory Group» - Режим доступа: https://www.arcweb.com/ (дата обращения 16.10.2017).

71. Официальный сайт компании «National Instruments» [электронный ресурс]: Официальный сайт компании «National Instruments» - Режим доступа: http://www.ni.com/ru-ru.html (дата обращения 16.10.2017).

72. Официальный сайт компании «Siemens» [электронный ресурс]: Официальный сайт компании «Siemens» - Режим доступа: https://www.siemens.com/global/en/home.html (дата обращения 16.10.2017).

73. Митин, Г.П. Системы автоматизации с использованием программируемых логических контроллеров: учеб. пособие / Г.П. Митин, О.В. Хазанова // – М.: ИЦ МГТУ «СТАНКИН», 2005. – 136 с.

74. Официальный сайт компании Rockwell Automation [электронный ресурс]: официальный сайт компании Mitsubishi Electric - Режим доступа: https://www.rockwellautomation.com/ (дата обращения 15.01.2019)

75. Официальный сайт компании Mitsubishi Electric [электронный ресурс]: официальный сайт компании Mitsubishi Electric - Режим доступа: https://opcfoundation.org/ (дата обращения 15.01.2019)

76. Официальный сайт компании «Bosch-Rexroth» [электронный ресурс]: Официальный сайт компании «Bosch-Rexroth» - Режим доступа: https://www.boschrexroth.com/ru/ru/ (дата обращения 16.10.2017).

77. Официальный сайт компании «B&R» [электронный ресурс]: Официальный сайт компании «B&R» - Режим доступа: https://www.br-automation.com/ru/soviershienstvo-v-avtomatizatsii/ (дата обращения 16.10.2017).

78. Официальный сайт компании «ОВЕН» [электронный ресурс]: Официальный сайт компании «ОВЕН» - Режим доступа: http://www.owen.ru/ (дата обращения 16.10.2017).

79. Официальный сайт компании «Fastwel» [электронный ресурс]: Официальный сайт компании «Fastwel» - Режим доступа: http://www.fastwel.ru/ (дата обращения 16.10.2017).

80. Официальный русскоязычный сайт компании «Opto22» [электронный ресурс]: Официальный русскоязычный сайт компании «Opto22» - Режим доступа: http://www.opto22.ru/ (дата обращения 16.10.2017).

81. Официальный сайт компании «Opto22» [электронный ресурс]: Официальный сайт компании «Opto22» - Режим доступа: http://www.opto22.com/ (дата обращения 16.10.2017).

82. Официальный русскоязычный сайт компании «CoDeSys» [электронный ресурс]: Официальный русскоязычный сайт компании «CoDeSys» - Режим доступа: http://www.codesys.ru/ (дата обращения 16.10.2017).

83. Официальный сайт программного продукта CoDeSys компании 3S - Smart Software Solution GmbH [электронный ресурс]: официальный сайт программного продукта CoDeSys компании 3S - Smart Software Solution GmbH - Режим доступа: https://www.codesys.com/ (дата обращения 01.02.2018).

84. Модели и алгоритмы синтеза кроссплатформенного контроллера автоматизации технологических процессов: автореферат дис. … канд. тех. наук: 05.13.06 / Ковалев Илья Александрович; [Место защиты: МГТУ "СТАНКИН"]. - Москва, 2017. - 25 с.

85. Официальный сайт программного продукта LabView компании National Instruments [электронный ресурс]: Официальный сайт программного продукта LabView компании National Instruments - Режим доступа: http://www.labview.ru/ (дата обращения 01.02.2018).

86. Официальный русскоязычный сайт программного продукта ISaGRAF компании Rockwell Automation [электронный ресурс]: Официальный русскоязычный сайт программного продукта ISaGRAF компании Rockwell Automation - Режим доступа: http://www.isagraf.ru/ (дата обращения 01.02.2018).

87. Официальный сайт программного продукта ISaGRAF компании Rockwell Automation [электронный ресурс]: Официальный сайт программного продукта ISaGRAF компании Rockwell Automation - Режим доступа: http://www.isagraf.com/index.htm (дата обращения 01.02.2018).

88. Официальный сайт компании Fiord [электронный ресурс]: Официальный сайт компании Fiord - Режим доступа: http://www.fiord.com/ (дата обращения 01.02.2018).

89. Официальный сайт компании KW software [электронный ресурс]: Официальный сайт компании KW software - Режим доступа: http://www.kw-software.com/ (дата обращения 01.02.2018).

90. Официальный сайт компании Infoteam software AG [электронный ресурс]: Официальный сайт компании Infoteam software AG - Режим доступа: https://www.infoteam.de/ (дата обращения 01.02.2018).

91. Официальный сайт компании Pro Sign [электронный ресурс]: Официальный сайт компании Pro Sign - Режим доступа: https://www.infoteam.de/ (дата обращения 01.02.2018).

92. ГОСТ Р МЭК 61131-6-2016 Контроллеры программируемые. Часть 6. Безопасность функциональная (IEC 61131-6: 2012, IDT).– М.: Стандартинформ, 2016. – 86 с.

93. Стандарт IEEE 1016-1998 Recommended Practice for Software Design Descriptions (методические рекомендации для описания программных решений). - The Institute of Electrical and Electronics Engineers, Inc. - 1998. - 23 с.

94. Стандарт IEEE 1471-2000 Recommended Practice for Architectural Description of Software-Intensive Systems (методические рекомендации для описания архитектуры программных систем). The Institute of Electrical and Electronics Engineers Inc. - 2000. - 29 с.

95. Липаев В. В. Формирование и применение профилей открытых информационных систем / В. В. Липаев, Е. А. Филинов - Открытые системы, 1997. № 5. с. 23

96. ГОСТ Р ИСО/МЭК ТО 10000-1-99. Информационная технология. Основы и таксономия международных функциональных стандартов. – М.: Стандартинформ, 2016. – 20 с.

97. ГОСТ Р МЭК 61131-1-2016 Контроллеры программируемые. Часть 1. Общая информация (IEC 61131-1: 2003, IDT). – М. : Стандартинформ, 2016. – 16 с.

98. ГОСТ Р МЭК 61131-2-2012 Контроллеры программируемые. Часть 2. Требования к оборудованию и испытания (IEC 61131-2: 2007, IDT). – М. : Стандартинформ, 2016. – 111 с.

99. ГОСТ Р МЭК 61131-3-2016 Контроллеры программируемые. Часть 3. Языки программирования (IEC 61131-3: 2013, IDT). – М. : Стандартинформ, 2016. – 230 с.

100. ГОСТ Р МЭК 61131-4:2004 Контроллеры программируемые. Часть 4. Руководство для пользователя (IEC 61131-4:2004, Programmable controllers. Part 4. User guidelines). – М. : Стандартинформ, 2016. – 111 с.

101. МЭК 61131-5-2000. Программируемые контроллеры. Спецификация сообщений (Programmable controllers. Part 5. Communications). The Institute of Electrical and Electronics Engineers Inc. - 2000. - 102 с.

102. МЭК 61131-7-2000. Программируемые контроллеры. Программирование с нечеткой логикой (IEC 61131-7-2000. Programmable controllers. Part 7. Fuzzy control programming). The Institute of Electrical and Electronics Engineers Inc. - 2000. - 116 с.

103. МЭК 61131-8-2003. Программируемые контроллеры. Руководящие принципы применения и реализации языков ПЛК (IEC 61131-8-2003. Programmable controllers. Part 8. Guidelines for the application and implementation of programming languages). The Institute of Electrical and Electronics Engineers Inc. - 2000. - 108 с.

104. ГОСТ Р 51840-2001. Программируемые контроллеры. Общие положения и функциональные характеристики. – М.: Стандартинформ, 2001. – 16 с.

105. ГОСТ 30607-98. Контроллеры программируемые станочные. Общие технические требования. – М.: Стандартинформ, 1998. – 15 с.

106. ГОСТ 26.013-81. Средства измерения и автоматизации. Сигналы электрические с дискретным изменением параметров входные и выходные. – М.: Стандартинформ, –7 с.

107. Официальная спецификация стандарта XML [электронный ресурс]: Официальная спецификация стандарта XML - Режим доступа: https://www.w3.org/TR/xml11/ (дата обращения 01.02.2018).

108. Стандарт ISO/IEC 9075:2008, Information technology—Database languages—SQL—Part 1 – 14.

109. Официальный сайт компании Microsoft [электронный ресурс]: Раздел по COM технологиям в библиотеке MSDN - Режим доступа: https://msdn.microsoft.com/en-us/library/ms680573.aspx (дата обращения 01.02.2018).

110. Стандарт ISO/IEC 9945:2003. Information technology - Portable Operating System Interface (POSIX)

111. Стандарт ГОСТ Р ИСО 9001 - 2008 Системы менеджмента качества. Требования. – М.: Стандартинформ, 2008.–34 с.

112. EIA / TIA RS-232 - Electronic Industries Accociation Recommended Standart 232 (Рекомендуемый стандарт 232 Ассоциации электронной промышленности и Ассоциации телесвязи)

113. ANSI/TIA-485-A - Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems (Электрические характеристики генераторов и приёмников для использования в балансных многоточечных системах)

114. Мартинова, Л.И. Реализация открытости управления электроавтоматикой станков в системе ЧПУ класса PCNC / Л.И. Мартинова, Н.В. Козак, Р.А. Нежметдинов, Р.Л. Пушков // Приборы и системы. Управление, контроль, диагностика. - 2011. - №02. - с. 11-16.

115. Мартинов, Г.М. Прикладные решения в области управления электроавтоматикой станков с ЧПУ класса PCNC / Г.М. Мартинов, Р.А. Нежметдинов, Н.В. Козак, Р.Л. Пушков // Промышленные АСУ и контроллеры, 2011. - №4. - с. 48-53

116. Мартинов, Г.М. Новый подход к построению программно реализованного контроллера электроавтоматики технологического оборудования с ЧПУ / Г.М. Мартинов, Р.А. Нежметдинов, А.С. Емельянов // Известия Юго-Западного государственного университета. - 2013. - №3 (48). - с. 156-166.

117. Мартинов, Г.М. Принципы построения кроссплатформенного программно реализованного контроллера электроавтоматики систем ЧПУ высокотехнологичными производственными комплексами / Г.М. Мартинов, Р.А. Нежметдинов, А.С. Емельянов // Вестник МГТУ Станкин. - 2013. - № 1 (24). - с. 42-51.

118. Мартинов, Г.М. Специфика построения редактора управляющих программ электроавтоматики стандарта МЭК 61131 / Г.М. Мартинов, Р.А. Нежметдинов, П.А. Никишечкин // Вестник МГТУ Станкин. - 2014. - № 4 (31). - с. 127-132

119. Мартинов, Г.М. Разработка средств визуализации и отладки управляющих программ для электроавтоматики, интегрированных в систему ЧПУ / Г.М. Мартинов, Р.А. Нежметдинов, П.А. Никишечкин // Вестник МГТУ Станкин. - 2012. - № 4 (23). - с. 134-138.

120. Martinov, G.M. Trends in the numerical control of machine-tool systems / G.M. Martinov, L.I. Martinova // Russian Engineering Research. - 2010. - Т. 30. - № 10. - с. 1041-1045.

121. Григорьев, С.Н. Подход к построению информационно-вычислительных сред виртуальных производственных корпораций / С.Н. Григорьев, Л.И. Мартинова // Межотраслевая информационная служба. - 2012. - № 4. - с. 31-37.

122. Мартинов, Г.М. Метод декомпозиции и синтеза специализированных систем ЧПУ / Г.М. Мартинов, Н.В. Козак, Р.А. Нежметдинов, А.С. Григорьев, А.И. Обухов, Л.И. Мартинова// Автоматизация в промышленности. - 2013. - № 5. - с. 8.

123. Мартинов, Г.М. Принципы построения распределенной системы ЧПУ технологическими машинами с использованием открытой модульной архитектуры / Г.М. Мартинов, Л.И. Мартинова, Н.В. Козак, Р.А. Нежметдинов, Р.Л. Пушков // Справочник. Инженерный журнал с приложением. - 2011. - № 12. – с. 45-51.

124. Мартинов, Г.М. Анализ систем ЧПУ, представленных на международной выставке "Металлообработка-Технофорум-2009", их новизна и особенности / Г.М. Мартинов, Л.И. Мартинова // Автоматизация в промышленности. - 2009. - № 12. - с. 59-65.

125. Martinov, G.M. Multifunction numerical control solution for hybrid mechanic and laser machine tool / G.M. Martinov, A.B. Ljubimov, A.S. Grigoriev, L.I. Martinova // Procedia CIRP. - 2012.- Т. 1. - с. 260-264.

126. Нежметдинов, Р.А. Построение специализированной системы ЧПУ для многокоординатных токарно-фрезерных обрабатывающих центров / Р.А. Нежметдинов, Р.Л. Пушков, С.В. Евстафиева, Л.И. Мартинова // Автоматизация в промышленности. - 2014. - № 6. - с. 25-28.

127. Мартинов, Г.М. Способ построения инструментария систем мониторинга и настройки параметров мехатронного технологического оборудования на основе специализированных программных средств / Г.М. Мартинов, Р.А. Нежметдинов, С.В. Соколов // Мехатроника, автоматизация, управление. 2012. № 7. С. 45-50.

128. Мартинова, Л.И. Организация межмодульного взаимодействия в распределенных системах ЧПУ. Модели и алгоритмы реализации / Л.И. Мартинова, Г.М. Мартинов // Мехатроника, автоматизация, управление. - 2010. - № 11. - с. 50-55.

129. Мартинова, Л.И. Практические аспекты реализации модулей открытой системы ЧПУ / Л.И. Мартинова, Г.М. Мартинов // Автотракторное электрооборудование. - 2002. - № 3. - с. 31- 37.

130. Мартинов, Г.М. Формирование базовой вычислительной платформы ЧПУ для построения специализированных систем управления / Г.М. Мартинов, Л.И. Мартинова // Вестник МГТУ "Станкин". - №1(24). – 2014. - с. 92-97.

131. Мартинов, Г.М. Кроссплатформенный программно-реализованный логический контроллер управления электроавтоматикой станков с ЧПУ / Г.М. Мартинов, Р.А. Нежметдинов // Автоматизация и современные технологии. – 2013. - №1. – с. 12- 16.

132. Разработка способа аппаратно-независимого управления электроавтоматикой для сокращения времени выпуска различных компоновок токарно-фрезерных станков с ЧПУ: автореферат дис. ... канд. тех. наук: 05.13.06 / Кулиев Абай Уангалиевич; [Место защиты: МГТУ "СТАНКИН"]. - Москва, 2016. - 23 с.

133. Расширение функциональных возможностей специализированных систем ЧПУ посредством организации многоцелевого канала взаимодействия их основных компонентов: автореферат дис. … канд. тех. наук: 05.13.06 / Никишечкин Петр Анатольевич; [Место защиты: МГТУ "СТАНКИН"]. - Москва, 2015. - 24 с.

134. ГОСТ 15971-90 Системы обработки информации. Термины и определения.– М.: Издательство стандартов, 1990. – 14 с.

135. Официальный сайт компании «Microsoft» [электронный ресурс]: Официальная страница «Windows 10 для «Интернета вещей» - Режим доступа: https://www.microsoft.com/ru-ru/windowsforbusiness/windows-iot (дата обращения 13.02.2018).

136. Мишель, Ж. Программируемые контроллеры / Ж. Мишель, К. Лоржо, Б. М. Эспьо // М.: Машиностроение. - 1986. – с. 180.

137. Мишель, Ж. Программируемые контроллеры: архитектура и применение / Ж. Мишель // М.: Машиностроение. - 1992. – с. 167.

138. Пушков, Р.Л. Практические аспекты построения многотерминального человеко-машинного интерфейса на примере системы ЧПУ «АксиОМА Контрол» / Р.Л. Пушков, С.В. Евстафиева, С.В. Соколов, Р.А. Абдуллаев, П.А. Никишечкин, А.У. Кулиев, А.Е. Сорокоумов // Автоматизация в промышленности. - 2013. - № 05. - с. 36-40.

139. Нежметдинов, Р.А. Разработка редактора по созданию управляющих программ для электроавтоматики станка с ЧПУ на базе программно-реализованного контроллера. / Р.А. Нежметдинов, П.А. Никишечкин, С.В. Евстафиева, Ю.С. Волкова // В сборнике: Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла промышленного продукта (CAD/CAM/PDM - 2013). - 2013. - С. 230-233

140. Карпов, Ю. Г. Теория автоматов / Ю. Г. Карпов // СПб.: Питер. – 2002. - 224 с.

141. Шалыто, А. А. Логическое управление. Методы аппаратной и программой реализации алгоритмов / А. А. Шалыто // СПб.: Наука. - 2000. — 780 с.

142. Дьяконов, В. П. Компьютерная математика. Теория и практика / В. П. Дьяконов // М.: Нолидж. — 2001. — 1296 с.

143. Филлипс, Ч. Системы управления с обратной связью / Ч. Филлипс, Р. Харбор // М.: Лаборатория Базовых Знаний, — 2001 — 616 с.

144. Каханер, Д. Численные методы и программное обеспечение / Д. Каханер, К. Моулер, С. Нэш // М.: Мир. - 2001. — 575 с.

145. Нежметдинов, Р.А. Моделирование термокомпенсационного регулирования станины станка наклонной компоновки высокой точности / Р.А. Нежметдинов, В.О. Давыденко // Материалы первого тура научно-практической конференции «Автоматизация и информационные технологии (АИТ-2016)». Сборник тезисов докладов. Том 1. Секция «Автоматизация и управление» - М.: ФГБОУ ВО «МГТУ «Станкин». - 2016. – с. 56- 60.

146. Дьяконов, В. П. MATLAB 6/6.1/6.5, Simulink 4/5. Основы применения. Полное руководство пользователя / В. П. Дьяконов // М.: Солон-Пресс. — 2002. — 768 с.

147. Дьяконов, В. П. Simulink 4. Специальный справочник / В. П. Дьяконов // СПб.: Питер. — 2002. — 528 с.

148. ГОСТ 16504-81 Система государственных испытаний продукции. Испытания и контроль качества продукции. Основные термины и определения. – М.: Стандартинформ, 2001 г.– 24 с.

149. ГОСТ 34.603-92 Виды испытаний автоматизированных систем – М.: Стандартинформ, 2004 г.– 24 с.

150. ГОСТ 27.002-89 Надежность в технике основные понятия. термины и определения. – М.: Стандартинформ, 2016 г.– 28 с.

151. ГОСТ 27.203-83. Технологические системы. Общие требования к методам оценки надежности. – М.: Издательство стандартов, 1984 г.– 8 с.

152. ГОСТ 27.410-87. Надежность в технике. Методы контроля показателей надежности и планы контрольных испытаний на надежность. – М.: Издательство стандартов, 1988 г.– 114 с.

153. ГОСТ 27.003-90. Состав и общие правила задания требований по надежности – М.: Стандартинформ, 2008 г.– 20 с.

154. ГОСТ 27.301-95. Надежность в технике. Расчет надежности. Основные положения– М.: Издательство стандартов, 1995 г.– 19 с.

155. Мартинов, Г.М. Принцип построения распределенной системы ЧПУ с открытой модульной архитектурой / Г.М. Мартинов, Н.В. Козак, Р.А. Нежметдинов, Р.Л. Пушков // Вестник МГТУ "Станкин". - 2010. - №4(12). - с. 116-122.

156. Григорьев, С.Н. Перспективы развития кроссплатформенных компьютерных систем числового программного управления высокотехнологичного оборудования / С.Н. Григорьев, А.Г. Андреев, Г.М. Мартинов // Автоматизация в промышленности. - 2011. - №5. - с. 3-8.

157. Нежметдинов, Р.А. Разработка подсистемы защиты информационных потоков для систем ЧПУ технологического оборудования / Р.А. Нежметдинов, А.С. Емельянов, Н.В. Козак // Известия Юго-Западного государственного университета. - 2013. - №5. - с. 61 – 67.

158. Нежметдинов, Р.А., Подход к проведению стендовых экспериментальных исследований систем логического управления технологическим оборудованием / Р.А. Нежметдинов // Вестник МГТУ «СТАНКИН». - №1. – 2017. - с. 28 – 34,

159. Козак, Н.В. Применение программно-реализованных логических контроллеров в системах автоматизации упаковочного оборудования / Н.В. Козак, Р.А. Нежметдинов // Автоматизация в промышленности. - 2012. - №11. - с.23-28.

160. Нежметдинов, Р.А. Программная реализация управления электроавтоматикой технологического оборудования с применением компонентного подхода / Р.А. Нежметдинов // Главный механик. – 2011. - №6. - с. 42-46.

161. Нежметдинов, Р.А Эффективность функционирования электроавтоматики станков с ЧПУ/ Р.А. Нежметдинов // Главный механик. - 2013. - № 9. - с. 33-43.

162. Нежметдинов, Р.А. Программно-реализованный логический контроллер – инновационный продукт для автоматизации технологического оборудования / Р.А. Нежметдинов // Инновации. - №8. – 2016. - с. 99-103.

163. Шемелин, В.К. Повышение качества архитектурных решений систем ЧПУ на основе программно реализованного контроллера типа SoftPLC / В.К. Шемелин, Р.А. Нежметдинов // Автоматизация и современные технологии. - №6. - 2008. - с. 33-35.

164. Шемелин, В.К. Применение технологии клиент-сервер при проектировании контроллера типа SoftPLC для решения логической задачи в рамках систем ЧПУ / В.К. Шемелин, Р.А. Нежметдинов // Автоматизация и современные технологии. - №3. - 2010. - с. 31-37.

165. Нежметдинов, Р.А. Расширение функциональных возможностей систем ЧПУ для управления механо-лазерной обработкой / Р.А. Нежметдинов, С.В. Соколов, А.И. Обухов, А.С. Григорьев // Автоматизация в промышленности. - 2011. - №05. - с. 49-53.

166. Nezhmetdinov, R. A. Extending the functional capabilities of NC systems for control over mechano-laser processing / Nezhmetdinov R. A., Sokolov S. V., Obukhov A. I., Grigor'ev A. S. // Automation and Remote Control. – 2014. - Volume 75. - Issue 5. – pp. 945-952.

167. Мартинов, Г. М. Организация распределенного управления станком гидроабразивной резки с ЧПУ / Г. М. Мартинов, Р. А. Нежметдинов, С. В. Рыбников, А. У. Кулиев // Мехатроника, автоматизация, управление. – 2011. - №11. - с. 35-39.

168. Мартинов, Г.М. Специфика построения панелей управления систем ЧПУ по типу универсальных программно-аппаратных компонентов / Г.М. Мартинов, Н.В. Козак, Р.А. Нежметдинов, А.Б. Любимов // Автоматизация и современные технологии. - 2010. - №7. - с. 34-40

169. Нежметдинов, Р.А. Принципы построения распределенных систем управления электроавтоматикой на базе взаимодействия автономных ПЛК / Р.А. Нежметдинов, А.У. Кулиев, Н.Ю. Червоннова // Труды международной научно-практической конференции "Передовые информационные технологии, средства и автоматизации и их внедрение на российских предприятиях" AITA 2011. - с. 725-728.

170. Mori, Masahiko. 5 axis mill turn and hybrid machining for advanced application / Mori, Masahiko, Fujishima Makoto, Yohei, // Procedia CIRP 1 (2012). - p.p. 22-27.

171. Crichigno Filho, Prediction of cutting forces in mill turning through process simulation using a five-axis machining center / Crichigno Filho, Joel Martins, // International journal of advanced manufacturing technology. - v. 58. - i. 1-4. - p. 71-80. - 2012

172. She Chen-Hua, Development of multi-axis numerical control program for mill-turn machine / She Chen-Hua, Hung Chih-Wei // Proceedings of the institution of mechanical engineers part b-journal of engineering manufacture. - v. 222. - Issue: 6. - p. 741-745. – 2008.

173. Мартинова, Л.И. Практические аспекты применения отечественной многофункциональной системы ЧПУ ""АксиОМАКонтрол" / Л.И. Мартинова, Н.В. Козак, Р.А. Нежметдинов, Р.Л. Пушков, А.И. Обухов // Автоматизация в промышленности. - 2012. - №5. - с.36-40.

174. Мартинов, Г.М. Метод декомпозиции и синтеза современных систем с ЧПУ / Г.М. Мартинов, Н.В. Козак, Р.А. Нежметдинов, А.С. Григорьев, А.И. Обухов, Л.И. Мартинова // Автоматизация в промышленности. - 2013. - № 5. - с. 9-15.

175. Martinov, G.M. Method of decomposition and synthesis of the custom CNC systems / G.M. Martinov, N.V. Kozak, R.A. Nezhmetdinov, A.S. Grigoriev, A.I. Obukhov, L.I. Martinova // Automation and remote Control. – 2017. - №78 - p. 525 - 536.

176. Мартинов, Г.М. Модульный подход к построению специализированной системы ЧПУ для обрабатывающих центров наклонной компоновки / Г.М. Мартинов, Р.А. Нежметдинов // Станки и инструменты. - №11. - 2014. - с. 28-33.

177. Martinov, G. M. Modular design of specialized numerical control systems for inclined machining centers / G. M. Martinov, R. A. Nezhmetdinov // Russian Engineering Research. - May 2015. - Volume 35. - Issue 5. – pp. 389-393.

178. Фомин, Е.И. Практические аспекты применения отечественной комплектной системы ЧПУ СТАНКИН NC 201 / Е.И. Фомин, П.А. Никишечкин // Автоматизация в промышленности. - №6. - 2014. - с.38-41.

179. Козак, Н.В. Концепция построения средств диагностики и управления устройствами электроавтоматики на базе OPC технологии / Н.В. Козак, Р.А. Абдуллаев // Системы управления и информационные технологии. - 2010. - Т. 41. - № 3. - с. 28-32.

180. Нежметдинов, Р.А.Управление электроавтоматикой токарных и токарно-фрезерных станков на базе Soft PLC / Р.А. Нежметдинов, А.У. Кулиев, А.Ю. Николушкин, Н.Ю. Червоннова // Автоматизация в промышленности. - 2014. - № 4. - с. 46-48.

181. Нежметдинов, Р.А. Числовое программное управление фрезерными обрабатывающими центрами с использованием высокоскоростных протоколов связи / Р.А. Нежметдинов, Р.Л. Пушков, А.Б. Любимов А.Б., Л.И. Мартинова Л.И., С.В. Евстафиева // Автоматизация в промышленности. - №5. - 2015. - с.24-26.

182. Мартинов, Г.М. Подход к реализации аппаратно-независимого управления электроавтоматикой токарных и токарно-фрезерных станков с ЧПУ / Г.М. Мартинов, Р.А. Нежметдинов, А.У. Кулиев // Авиационная техника. - 2016. - №2. - с.128-131.

183. Martinov, G.M. Approach to implementing hardware-independent automatic control systems of lathes and lathe-milling CNC machines / G.M. Martinov, R.A. Nezhmetdinov, and A.U. Kuliev // Izv.Vuz. Av. Tekhnika. 2016. - no. 2. - pp. 128–131. [Russian Aeronautics (Engl. Transl.) vol.59, no. 2, pp. 293-296.]

184. Нежметдинов, Р.А. Подход к построению систем логического управления технологическим оборудованием для реализации концепции «Индустрия 4.0» / Р.А. Нежметдинов, П.А. Никишечкин, И.А. Ковалев, Н.Ю. Червоннова // Автоматизация в промышленности. - №5. - 2017. - с.5-9.

185. Нежметдинов, Р.А. Практические аспекты применения программно-реализованного контроллера для управления электраовтоматикой вертикально-фрезерных станков Quaser MV184 /

Р.А. Нежметдинов, П.А. Никишечкин, Р.Л. Пушков, С.В. Евстафиева // Автоматизация в промышленности. - №5. - 2016. - с.15-18.

186. Martinov, G.M. Implementation of Control for Peripheral Machine Equipment Based on the External Soft PLC Integrated with CNC / G.M. Martinov, N.V. Kozak, R.A. Nezhmetdinov // 2017. - International Conference on Industrial Engineering. - Applications and Manufacturing (ICIEAM). - 16-19 May. - 2017. - p.1-4.

187. Нежметдинов, Р.А. Управление автоматической сменой инструмента на многоцелевых обрабатывающих центрах с применением унифицированных программных / Р.А. Нежметдинов, П.А. Никишечкин, Р.Л. Пушков // Промышленные АСУ и контроллеры. - №6. - 2016. - с. 19-24.

188. Martinova, L. I. The Russian multi-functional CNC system AxiOMA control: Practical aspects of application / L. I. Martinova, , N. V.Kozak, R. A. Nezhmetdinov // Automation and remote control. - Vol: 76. - No: 1. - p. 179-186. - JAN 2015.

189. Минаев, И.Г. Свободно программируемые устройства в автоматизированных системах управления / И.Г. Минаев, В.В. Самойленко, Д.Г. Ушкур, И.В. Федоренко // Ставрополь: АГРУС. 2016. - 168 с.

190. Официальный сайт компании ОВЕН [электронный ресурс]: Раздел - ТРМ210 ПИД-регулятор с универсальным входом и RS-485 - Режим доступа: https://owen.ru/product/trm210 (дата обращения 01.09.2019).

191. Юдицкий, С. А., Разработка принципов и методов построения устройств логического управления дискретными технологическими процессами на основе сетей Петри: диссертация ... доктора технических наук: 05.13.05. - Москва, 1985. - 354 с. : ил.

192. Пневматические системы управления приводом машин-автоматов: (Методы построения). - Москва: Энергия, 1968. - 88 с.: черт.

193. Шалыто, А.А. Объектно-ориентированный подход к автоматному программированию, А.А. Шалыто, Д. Г. Шопырин, Информационно- управляющие системы, №5, 2003, с. 29 – 39.

194. Рогов, С.Л. Подсистемы противоаварийной защиты опасных производственных объектов в составе информационно-измерительных и управляющих систем: диссертация ... кандидата технических наук: 05.11.16 / Рогов Сергей Львович; [Место защиты: Пенз. гос. ун-т]. - Пенза, 2012. - 173 с.

195. Александровская, Л. Н. Современные методы обеспечения безопасности сложных технических систем/ Л.Н. Н.Александровская, А. П. Афанасьев, А. А Лисов. М.: Логос 2001. - 206с.

196. Базовский, И. Надежность. Теория и практика / И. Базовский. М.: Мир. 1965. - 373с.Г

197. Байхельт, Ф. Надежность и техническое обслуживание. Математический подход / Ф. Байхельт, П. Франкен М.: Сов. радио. 1971. -272 с.7

198. Барзилович, Е. Ю. и др. Вопросы математической теории надежности / Е. Ю. Барзилович. М.: Радио и связь. 1983. - 376 с.о

199. Федосеев, А.М. Релейная защита электроэнергетических систем. Релейная защита сетей. –М.: Энергиздат, 1984.

200. Справочник по релейной защите / Под общей редакцией М.А. Берковича. М. –Л.: госэнергоиздат, 1963 г., 512 с.

201. Чернобровов, Н.В. Релейная защита, изд.3, перераб. и допол., М.-Л. «Энергия», 1966, 760 с.: ил.

202. Программное управление станками / под редакцией В.Л. Сосонкина – М.: Машиностроение, 1981. – 398 с.: ил.

203. Сосонкин, В.Л. Микропроцессорные системы числового программного управления станками / В. Л. Сосонкин. - М. : Машиностроение, 1985. - 288 с. : ил.

204. Соломенцев, Ю.М. Управление гибкими производственными системами / Ю. М. Соломенцев, В. Л. Сосонкин. - М. : Машиностроение, 1988. – 351 с. : ил.

205. Сосонкин, В.Л. Программное управление технологическим оборудованием / В.Л. Сосонкин, - М.: Машиностроение, 1991. – 509 с.: ил.

206. Буч Гради, Объектно-ориентированный анализ и проектирование с примерами приложений / Г. Буч, Р. Максимчук, М. Энгл и др., 3-е изд.: Пер. с англ. – М : ООО «И.Д. Вильямс», 2008. – 720 с. : ил.

Приложение А Документы об использовании результатов диссертационного исследования

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО «МГТУ «СТАНКИН»)

Вадковский пер., д. 1, Москва, ГСП-4, 127994. Тел.: (499) 973-30-76. Факс: (499) 973-38-85
E-mail: rector@stankin.ru

АКТ ОБ ИСПОЛЬЗОВАНИИ
результатов диссертационной работы
Нежметдинова Рамиля Амировича
на тему «ПРИНЦИПЫ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ»

Настоящим актом подтверждаем, что результаты диссертационной работы доцента кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» Нежметдинова Р.А. использованы при реализации следующих работ:

- государственного задания № 2.1237.2017/4 при поддержке Минобрнауки России;
- ФЦП «Научные и педагогические кадры инновационной России» (Минобрнауки РФ) по государственным контрактам: П-1313 от 09.06.2010 «Создание распределенных вычислительных систем управления производственно-технологическим оборудованием на базе логических контроллеров и контроллеров автоматизации» (руководитель проекта), П-1368 от 11.06.2010 «Обработка, хранение, передача и защита технологической информации в распределенных системах управления» (руководитель проекта), 16.740.11.0228 от 22.09.2010 «Распределенная компьютерная система управления механолазерным комплексом прецизионной обработки», 02.740.11.0488 от 18.11.2009 «Создание гетерогенной распределенной компьютерной системы для управления в реальном времени децентрализованными высокотехнологичными производствами, объединенными в виртуальные корпорации» (исполнитель в проекте);
- государственным контрактам по заданию Минобрнауки РФ: № 1357 «Методы, модели и алгоритмы построения компьютерных систем управления для цифрового производства» (исполнитель в проекте), 2.1237.2017/ПЧ «Исследование и разработка высокопроизводительных распределенных систем управления на базе микропроцессоров «Эльбрус» для цифровых производств» (исполнитель в проекте);

- грантам Президента РФ в поддержку молодых ученных кандидатов наук по государственным контрактам: 16.120.11.323-МК от 18.02.2011 «Создание кроссплатформенного программно реализованного логического контроллера для управления технологическим оборудованием» (руководитель проекта), 14.124.13.4353-МК от 04.02.2013 «Организация управления децентрализованными производствами с применением программно реализованных средств электроавтоматики» (руководитель проекта);
- ФЦП "Национальная технологическая база" (Минпромторг РФ) в рамках государственных контрактов: 9411.1003702.05.010 от 23.09.09 при создании установки гидроабразивной резки (совместно с ОАО НИАТ и ООО «Савеловский машиностроительный завод», исполнитель в проекте), 252и/60М от 29.09.2011 при создании гаммы токарно-фрезерных высокоточных обрабатывающих центров с числовым программным управлением наклонной компоновки (совместно с ОАО «САСТА», исполнитель в проекте);
- договору 14-45/х от 10.07.2014 с ОАО «Ковровский электромеханический завод» по разработке электрооборудования и изготовлении комплектной системы ЧПУ для оснащения станка MV184P/15C (исполнитель в проекте).

Результаты работы в виде программ и методических материалов внедрены в учебный процесс кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» в рамках освоения дисциплин:

- «Автоматика и управление движением», «Автоматные модели в системах управления» подготовки бакалавров по направлению 15.03.04 «Автоматизация технологических процессов и производств»;
- «Программируемые логические контроллеры в системах управления» подготовки магистров по направлению 15.04.04 «Автоматизация технологических процессов и производств»;
- «Информационные системы в автоматизированном производстве» подготовки аспирантов по научной специальности 05.13.06 «Автоматизация и управление технологическими процессами и производствами».

Проректор по
образовательной деятельности д.э.н., проф. Ю.Я. Еленева

Проректор по научной работе и научно-
технической политике к.т.н. А.А. Зеленский

ОТКРЫТОЕ АКЦИОНЕРНОЕ ОБЩЕСТВО
«НАЦИОНАЛЬНЫЙ ИНСТИТУТ АВИАЦИОННЫХ ТЕХНОЛОГИЙ»
ОАО НИАТ

АКТ О ВНЕДРЕНИИ

результатов диссертационной работы

Нежметдинова Рамиля Амировича

на тему «ПРИНЦИПЫ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ»

Настоящим актом подтверждаем, что результаты диссертационной работы доцента кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» Нежметдинова Р.А. использованы при реализации опытно-конструкторской работы в рамках ФЦП «Национальная технологическая база» по государственному контракту № 9411.1003702.05.010 от 23 сентября 2009 г. при создании пятикоординатной установки гидроабразивной резки (совместный проект ОАО НИАТ, ООО «Савеловский машиностроительный завод» и ФГБОУ ВО «МГТУ «СТАНКИН»).

Генеральный директор А.В. Попов

Заместитель генерального директора
по научно-исследовательской
деятельности А.В. Коваленко

АКЦИОНЕРНОЕ ОБЩЕСТВО
«КОВРОВСКИЙ ЭЛЕКТРОМЕХАНИЧЕСКИЙ ЗАВОД»
завод основан в 1898 г.

601919,Россия, г.Ковров, Владимирской обл., ул. Крупской, 55
Тел.: (49232) 93546, 93457.
Факс: (49232) 93544, 30077, 30846.
Телетайп 218724 **ПРИБОР.**
E-mail: kancelar@kemz.org
Internet: www.kemz.kc.ru.

АКТ

об использовании результатов диссертационной работы

Нежметдинова Рамиля Амировича на тему:

«ПРИНЦИПЫ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ»

Настоящим актом подтверждаю, что результаты диссертационной работы доцента кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» Нежметдинова Рамиля Амировича использованы при реализации научно-исследовательской и опытно-конструкторской работы по теме «Разработка электрооборудования и изготовление комплектной системы ЧПУ «АксиОМАКонтрол» для оснащения и испытания станка QuaserMV184P/15C».

Директор по станкостроению
АО «КЭМЗ» И.А. Врублевский

ПУБЛИЧНОЕ АКЦИОНЕРНОЕ ОБЩЕСТВО
ТУЛЬСКИЙ ОРУЖЕЙНЫЙ ЗАВОД
Советская улица, дом 1-а, город Тула, 300002, Российская Федерация
телефон (4872) 32-17-01 факс (4872) 32-17-60 mail@tulatoz.ru www.tulatoz.ru
ОГРН 1027100507147 ИНН 7107003303 КПП 710701001 ОКПО 08627649

08.11.2019 № МОБ-564

На № _____ от _____

МГТУ «СТАНКИН»

Вадковский пер., 1, Москва

АКТ ОБ ИСПОЛЬЗОВАНИИ

результатов диссертационной работы

Нежметдинова Рамиля Амировича

на тему «ПРИНЦИПЫ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ»

Настоящим актом подтверждаем, что результаты диссертационной работы доцента кафедры компьютерных систем управления ФГБОУ ВО «МГТУ «СТАНКИН» Нежметдинова Р.А. использованы ПАО «ТУЛЬСКИЙ ОРУЖЕЙНЫЙ ЗАВОД» при реализации опытно-конструкторских работ и в учебных программах при организации повышения квалификации сотрудников предприятия.

Заместитель генерального директора
по безопасности и управлению персоналом М.А. Барышев

Приложение Б Объекты интеллектуальной собственности

РОССИЙСКАЯ ФЕДЕРАЦИЯ

ПАТЕНТ

НА ПОЛЕЗНУЮ МОДЕЛЬ

№ 126855

ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС ДЛЯ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ С ЧПУ

Патентообладатель(ли): ***Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования "Московский государственный технологический университет "СТАНКИН" (ФГБОУ ВПО МГТУ "СТАНКИН") (RU)***

Автор(ы): ***см. на обороте***

Заявка № **2012146709**

Приоритет полезной модели **02 ноября 2012 г.**

Зарегистрировано в Государственном реестре полезных моделей Российской Федерации ***10 апреля 2013 г.***

Срок действия патента истекает **02 ноября 2022 г.**

Руководитель Федеральной службы по интеллектуальной собственности

Б.П. Симонов

Автор(ы): ***Мартинов Георги Мартинов (RU), Нежметдинов Рамиль Амирович (RU), Любимов Александр Борисович (RU)***

РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **126 855** (13) **U1**

(51) МПК
G05B 19/00 (2006.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(12) ТИТУЛЬНЫЙ ЛИСТ ОПИСАНИЯ ПОЛЕЗНОЙ МОДЕЛИ К ПАТЕНТУ

(21)(22) Заявка: 2012146709/08, 02.11.2012

(24) Дата начала отсчета срока действия патента:
02.11.2012

Приоритет(ы):
(22) Дата подачи заявки: 02.11.2012

(45) Опубликовано: 10.04.2013 Бюл. № 10

Адрес для переписки:
127994, Москва, ГСП-4, Вадковский пер., 1,
ФГБОУ ВПО МГТУ "СТАНКИН",
помощнику ректора по интеллектуальной
собственности А.Л. Храмцову

(72) Автор(ы):
Мартинов Георги Мартинов (RU),
Нежметдинов Рамиль Амирович (RU),
Любимов Александр Борисович (RU)

(73) Патентообладатель(и):
Федеральное государственное бюджетное
образовательное учреждение высшего
профессионального образования
"Московский государственный
технологический университет "СТАНКИН"
(ФГБОУ ВПО МГТУ "СТАНКИН") (RU)

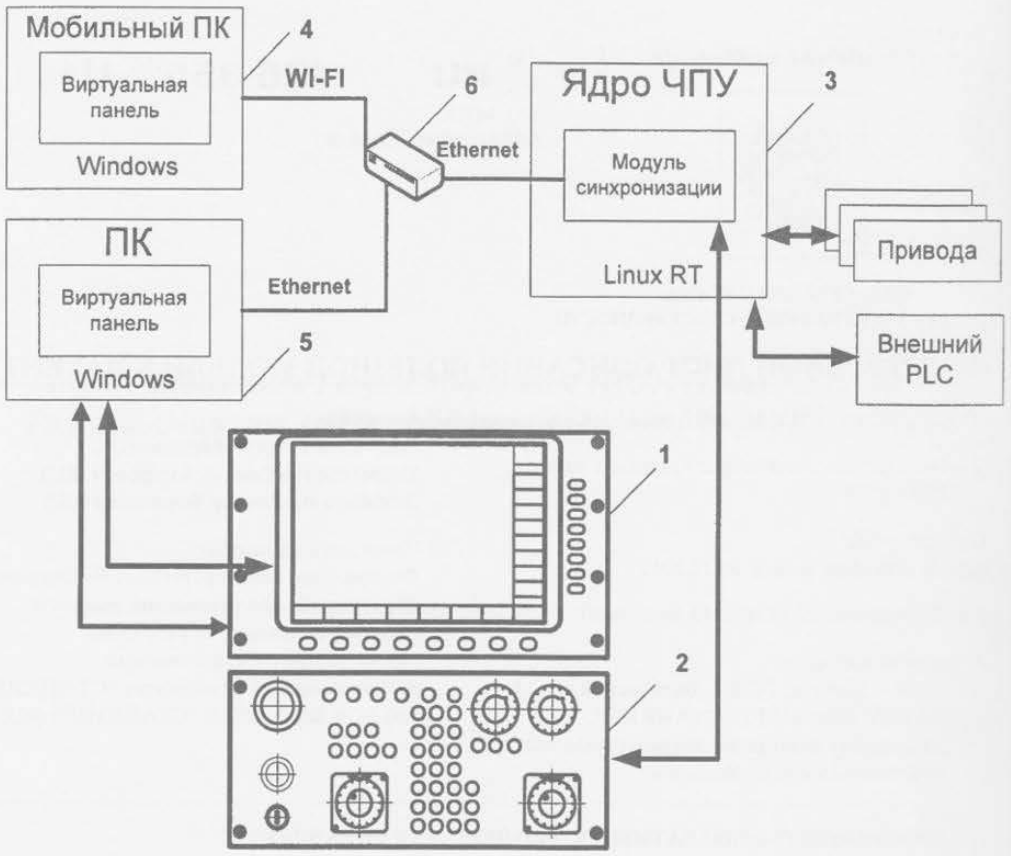
(54) ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС ДЛЯ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ОБОРУДОВАНИЕМ С ЧПУ

(57) Формула полезной модели

Программно-аппаратный комплекс для управления технологическим оборудованием с ЧПУ, содержащий коммутационно-соединенные панель оператора системы управления, станочную панель системы управления, промышленный компьютер с установленным ядром системы управления, отличающийся тем, что он содержит, по меньшей мере, один мобильный компьютер с установленной виртуальной панелью, персональный компьютер с установленной виртуальной панелью, сетевой маршрутизатор, при этом ядро системы управления выполнено с модулем синхронизации, а станочная панель подключена напрямую к промышленному компьютеру с возможностью обработки и передачи данных на модуль синхронизации, кроме того, мобильный и персональный компьютеры подключены к промышленному компьютеру системы управления посредством сети Ethernet.

RU 126855 U1

RU 126855 U1



RU 126855 U1

RU 126855 U1

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2010614074

Универсальный редактор управляющих программ для систем числового программного управления

Правообладатель(ли): ***Государственное образовательное учреждение высшего профессионального образования Московский государственный технологический университет «Станкин» (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Мартинова Лилия Ивановна, Нежметдинов Рамиль Амирович, Григорьев Антон Сергеевич, Пушков Роман Львович (RU)***

Заявка № **2010612277**

Дата поступления **29 апреля 2010 г.**

Зарегистрировано в Реестре программ для ЭВМ

23 июня 2010 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2010617381

Универсальная виртуальная панель управления технологическим оборудованием

Правообладатель(ли): ***Государственное образовательное учреждение Высшего профессионального образования Московский государственный технологический университет «Станкин» (RU)***

Автор(ы): ***Григорьев Антон Сергеевич, Нежметдинов Рамиль Амирович, Кулиев Абай Уангалиевич (RU)***

Заявка № **2010615580**

Дата поступления **14 сентября 2010 г.**

Зарегистрировано в Реестре программ для ЭВМ

10 ноября 2010 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011610551

Компонент графической визуализации параметров технологической системы

Правообладатель(ли): ***Государственное образовательное учреждение высшего профессионального образования Московский государственный технологический университет «Станкин» (RU)***

Автор(ы): ***Нежметдинов Рамиль Амирович, Соколов Сергей Владимирович, Саламатин Евгений Валериевич (RU)***

Заявка № **2010616728**

Дата поступления **2 ноября 2010 г.**

Зарегистрировано в Реестре программ для ЭВМ

11 января 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011610554

Универсальный интерпретатор языка управляющих программ для механо-лазерных станков с ЧПУ

Правообладатель(ли): ***Государственное образовательное учреждение высшего профессионального образования Московский государственный технологический университет «Станкин» (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Нежметдинов Рамиль Амирович, Обухов Александр Игоревич (RU)***

Заявка № **2010616732**

Дата поступления **2 ноября 2010 г.**

Зарегистрировано в Реестре программ для ЭВМ

11 января 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011611201

Программный комплекс интерпретации языка управляющих программ для механо-лазерного технологического оборудования с ЧПУ

Правообладатель(ли): ***Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Нежметдинов Рамиль Амирович, Обухов Александр Игоревич (RU)***

Заявка № **2010617700**

Дата поступления **7 декабря 2010 г.**

Зарегистрировано в Реестре программ для ЭВМ

4 февраля 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011612404

Программный эмулятор задач контроллера электроавтоматики систем ЧПУ

Правообладатель(ли): ***Государственное образовательное учреждение высшего профессионального образования Московский государственный технологический университет «Станкин» (RU)***

Автор(ы): ***Козак Николай Владимирович, Нежметдинов Рамиль Амирович, Абдуллаев Роман Ахматалиевич (RU)***

Заявка № **2011610524**

Дата поступления **1 февраля 2011 г.**

Зарегистрировано в Реестре программ для ЭВМ

23 марта 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011615522

Программный компонент системы управления для организации работы с контроллерами управления приводами по протоколу Memobus

Правообладатель(ли): ***Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Пушков Роман Львович, Нежметдинов Рамиль Амирович, Сорокоумов Артём Евгеньевич (RU)***

Заявка № **2011613812**

Дата поступления **24 мая 2011 г.**

Зарегистрировано в Реестре программ для ЭВМ

14 июля 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011615523

Компонент графической визуализации параметров скорости системы Числового Программного Управления

Правообладатель(ли): ***Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Нежметдинов Рамиль Амирович, Дубровин Иван Андреевич (RU)***

Заявка № **2011613813**

Дата поступления **24 мая 2011 г.**

Зарегистрировано в Реестре программ для ЭВМ

14 июля 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2011615524

Программный инструментарий для разработки и тестирования шаблонов регулярных выражений

Правообладатель(ли): ***Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Нежметдинов Рамиль Амирович, Григорьева Валерия Сергеевна (RU)***

Заявка № **2011613814**

Дата поступления **24 мая 2011 г.**

Зарегистрировано в Реестре программ для ЭВМ

14 июля 2011 г.

Руководитель Федеральной службы по интеллектуальной собственности, патентам и товарным знакам

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2012660015

Программный компонент проектирования и редактирования управляющих программ для ПЛК на языке Functional Blocks

Правообладатель(ли): ***Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)***

Автор(ы): ***Мартинов Георги Мартинов, Нежметдинов Рамиль Амирович, Кулиев Абай Уангалиевич, Никишечкин Петр Анатольевич, Ковалев Илья Александрович (RU)***

Заявка № **2012617646**

Дата поступления **13 сентября 2012 г.**

Зарегистрировано в Реестре программ для ЭВМ

8 ноября 2012 г.

Руководитель Федеральной службы по интеллектуальной собственности

Б.П. Симонов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

RU 2013615768

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

ГОСУДАРСТВЕННАЯ РЕГИСТРАЦИЯ ПРОГРАММЫ ДЛЯ ЭВМ

Номер регистрации (свидетельства):
2013615768

Дата регистрации: **20.06.2013**

Номер и дата поступления заявки:
2013613431 25.04.2013

Дата публикации: **20.09.2013**

Авторы:
Мартинов Георги Мартинов (RU),
Нежметдинов Рамиль Амирович (RU),
Григорьев Антон Сергеевич (RU),
Ковалев Илья Александрович (RU)

Правообладатель:
Российская Федерация, от имени которой выступает Министерство промышленности и торговли Российской Федерации (RU)

Программа для ЭВМ создана по государственному контракту

Открытое акционерное общество "Саста" (RU) является исполнителем работ по государственному контракту

Название программы для ЭВМ:
Программный компонент реализации машины состояний ядра системы управления электроавтоматикой

Реферат:
Программа реализует функционал ядра SoftPLC (программно-реализованного контроллера электроавтоматики), который создан на базе автоматной парадигмы программирования, при которой система имеет конечное число состояний, переход между которыми осуществляется при выполнении ряда условий. Машина состояний ядра SoftPLC реализована по типу конечного автомата с пятью состояниями и переходами между ними.

Тип реализующей ЭВМ:	IBM PC – совмест. ПК на базе процессора Intel Pentium III
Язык программирования:	C++/C#
Вид и версия операционной системы:	Windows NT/2000/XP/Vista/7, Linux
Объем программы для ЭВМ:	8,86 Кб