Соколов Ю.А., Кулешов В.Г.

ПОСТРОЕНИЕ СИСТЕМ УПРАВЛЕНИЯ НА БАЗЕ КОНТРОЛЛЕРОВ Direct Logic.

Главы книги

1. О том, что есть в этой книге.

В настоящей книге отражён опыт проектирования систем управления технологического оборудования научно-технического центра ОАО «Электромеханика» (г. Ржев). Книга содержит не только общий обзор различных структур систем управления на базе программируемых логический контроллеров (ПЛК) семейства DirectLogic, но и подробное описание стиля

программирования и разработки алгоритмов управления технологическими процессами. Книга написана для пользователей, которые успели поработать с Direct Logic, но желают расширить свои знания.

Представленный материал иллюстрируется структурными схемами, примерами программ на языке RLL plus и содержит советы по программированию контроллеров для реальных технологических процессов. Такой подход поможет оперативно применить полученные знания в новых ситуациях при разработке СУ. Мы надеемся, что книга послужит не только руководством по конкретным технологическим моделям, но и источником для размышлений при решении различных задач управления.

Особенностью книги является её незавершённость. Конечно, мы старались придать структурность и последовательность при изложении, но в тоже время не претендовали на всестороннее и подробное рассмотрение архитектуры и возможностей всего семейства контроллеров Direct Logic, а также разнообразных модулей, которые могут входить в состав систем управления на базе этих контроллеров; кроме того, мы решили не повторять техническую документацию, в которой описана система команд, предпочтя обратить внимание на специфику команды, либо обстоятельства, при которых применение той или иной команды более рационально и целесообразно.

Повышение эффективности управления во многом зависит от улучшения организации структуры системы управления и информационного обеспечения. С позиций управления можно выделить следующие основные классы структур СУ: децентрализованную, централизованную, централизованную рассредоточенную и иерархическую.

Функции СУ подразделяют на управляющие, информационные и вспомогательные управляющие. Это функции, результатом которых является выработка и реализация управляющих воздействий на технологический объект. К управляющим функциям относят регулирование технологических переменных, логическое управление операциями, оптимальное управление режимами, адаптивное управление; информационные функции – сбор, обработка и представление информации для последующего анализа; вспомогательные функции - обеспечение контроля за состоянием технических и программных средств системы.

Определение архитектуры и базовых программно-аппаратных средств при проектировании СУ с повышенными требованиями к качеству, срокам разработки, стоимости и надёжности представляет на сегодня довольно непростую задачу. Рынок ПЛК иностранных и отечественных фирм достаточно большой. Поэтому проектирование СУ целесообразно разделить на несколько этапов:

- ✓ определение класса системы: PLC, DCS, PC Control – soft PLC, PC Control – Win PLC, PLC Network;
- ✓ выбор аналоговых и дискретных интерфейсных модулей;
- ✓ определение инструментальных программных средств;
- ✓ разработка прикладного программного обеспечения.

ПЛК давно уже стали неотъемлемой частью современной системы управления технологического оборудования. Поэтому разработке прикладного программного обеспечения уделяется достаточно серьёзное внимание. Мы убеждены в необходимости профессионального подхода к разработке программного обеспечения систем управления.

Эффективная работа больших технологических систем (Т-систем) требует построения многоуровневой СУ, связанных между собой информационной сетью. Сетевые средства промышленной коммуникации обеспечивают надёжное и гибкое управление. Технология электронно-лучевой сварки, вакуумного литья и отжига охватывает большой комплекс физических и химических процессов, характеризующихся стохастическим поведением, отсутствием математической модели. Поэтому при управлении важным становится не только степень достижения оптимума целевой функции, но и выполнение технологических ограничений, качество выпускаемой продукции, автоматическое обнаружение неисправностей, анализ аварийных и внештатных ситуаций.

Уровень оборудования во многом определяет система управления как наиболее наукоёмкая составляющая технологической системы. На предприятии разработан и освоен целый ряд технологических систем с различными структурами системы управления: CNC, PCNC, PC Control, PLC, PLC Network:

- малые Т-системы (сварочные автоматы типа "АДСВ" и "УСКС", вакуумная установка «УВС-4» автоклав типа "А9-КСС", установка для нанесения металлических и керамических порошковых материалов методом плазменного напыления "УПУ-8М");
- средние Т-системы (вакуумные плавильные установки типа "ВИП-5", установка для нанесения светоотражающих покрытий "1АП-976", установка для нанесения защитных покрытий «АПН-250»);
- большие территориально распределённые Т-системы (установки для отжига сварных конструкций типа «ПВ-900», установки для литья лопаток двигателей из жаропрочных сплавов методом высокоскоростной и направленной кристаллизации в вакууме типа «ВИП НК», установки для электронно-лучевой сварки в вакууме типа “ЭЛУ”).

Успешное выполнение программы автоматизации предъявляет новые требования к исследованию задач развития производственных систем: повышение уровня системного мышления и подготовки технических заданий, применение новых методов исследования. Определяющим фактором повышения эффективности производственной системы является наличие мобильной и оптимальной системы управления реального времени, адекватно отображающей протекающие в системе процессы.

Эффективной эксплуатации систем управления на базе контроллеров Direct Logic способствует удобный программный пакет DirectSOFT, возможность применения нескольких типов центральных процессоров различной производительности, наличие разнообразных модулей ввода-вывода дискретных и аналоговых сигналов, функциональных модулей и коммуникационных портов.

Кроме улучшенных потребительских свойств контроллеры Direct Logic обладают богатыми коммуникационными возможностями, позволяющими включать их в промышленную сеть. Надёжным и недорогим средством для коммуникации в условиях производственного участка является сеть на основе интерфейса RS-485 или RS-422.

Для программирования ПЛК и SoftPLC Международный электротехнический комитет (МЭК) принял стандарт IEC 1131-3, который описывает пять языков программирования – графических: релейных диаграмм (Ladder Diagrams – LD), функциональных боковых диаграмм (Function Block Diagramm – FBD), последовательных функциональных схем (Seguential

Function Chart - SPC); текстовых: список инструкций (Instruction List – IL), структурированный текст (Structured Text - ST). Связь между контроллерами и станциями управления верхнего уровня осуществляется по сети Industrial Ethernet.

Контроллеры DL-205 предоставляет четыре основных способа программирования технологических приложений: стандартный RLL, $RLL^{plus,}$ (с использованием стадий), временной/событийный командоаппарат и контуры ПИД-регулирования. Все эти способы достаточно просты и помогают упростить последующую модификацию программы. Очевидно, что такое разнообразие даёт возможность выбирать средства программирования адекватно приложениям и задачам. Каждый способ программирования имеет свои особенности, касающиеся как общих вопросов, связанных типами команд, так и более сложных команд вплоть до контуров ПИД-регулирования.

Программирование в виде диаграмм RLL – хороший инструмент для решения логических и обычной работы с регистром/аккумулятором процессора. На наш взгляд, область технологических приложений, решаемых средствами RLL невелика.

Семейство процессоров DirectLogic поддерживают встроенные команды RLL^{plus}. Эти команды существенно облегчают проектирование и создание конкретных технологических приложений. RLL^{plus} - вариант языка программирования, который, несмотря на свой достаточно низкий уровень, не требует глубокого знания особенностей аппаратной части контроллеров. Структура программы на RLL^{plus}, как правило, совокупность команд, вырабатывающих события в определённом порядке. Программистам удобно иметь совокупность команд, позволяющую управлять последовательными и параллельными операциями в задачах управления.

Разнообразие разработанных на предприятии средств автоматизации, программно-аппаратных средств и алгоритмов регулирования позволяют в короткие сроки при минимальных затратах разработать СУ для различных технологических приложений. Отличительными чертами такой СУ являются гибкость, многофункциональность, возможность программной модернизации алгоритма работы и технологии, удобный интерфейс оператора. Высокий уровень профессионализма специалистов позволяет создавать СУ высокой сложности и с повышенными требованиями к надёжности.

2. Структуры систем управления, построенных на базе ПЛК семейства Direct Logic.

На сегодняшний день научно-технический центр ОАО «Электромеханика» разработал более 15 проектов систем управления (СУ) с различными структурами на базе контроллеров Direct Logic.

Модульный принцип конструктивного исполнения программируемых контроллеров Direct Logic, применяемых в СУ, позволил скомпоновать устройства программно-логического управления в соответствии с особенностями решаемых заказчиком задач.

Разнообразие модулей контроллеров делает оптимальным выбор адекватного и экономичного изделия для любой конкретной технологической задачи – от реализации простейших централизованных систем управления с ограниченным количеством обрабатываемой информации до децентрализованных систем с практически неограниченными объёмами обрабатываемой информации.

2.1. Системы управления, построенные на базе процессора D2-250.

Печи типа «ПАП» предназначены для реализации технологии аэродинамичесого нагрева, суть которой состоит в преобразовании механической энергии центробежного вентилятора, создающего поток воздуха в закрытом объёме, в тепловую. Аэродинамический нагрев обеспечивает высокую равномерность температурного поля по объёму рабочей камеры. Конвективный теплообмен обеспечивает более равномерный прогрев изделий.

Агрегат «ПАП-8» предназначен для закалки и искусственного старения алюминиевых сплавов и отпуска титана, а также для других технологических процессов, требующих нагрева в интервале рабочих температур от 55 до 500^0C $\pm 2^0$C.

Получение теплового эффекта в агрегате производится за счёт молекулярного трения в потоке воздуха, двигающегося по замкнутому объёму, а также за счёт трения между потоком воздуха и стенками термической камеры.

Система управления установки «ПАП-8» реализована на базе микроконтроллера Direct Logic семейства DL-205 и персонального компьютера, который имеет выход в локальную сеть участка цеха.

ПЛК семейства DL-205 предназначен для автоматизации малых и средних технологических установок. Модульная конструкция контроллера позволила выбрать оптимальный состав средств дискретного и аналогового ввода-вывода. Программное обеспечение контроллера обеспечивает ПИД регулирование рабочей температуры в камере, архивирование параметров технологического процесса, ведение журнала событий и аварийных ситуаций, а также выполнение блокировок, переход установки в безопасное состояние при возникновении аварийных ситуаций.

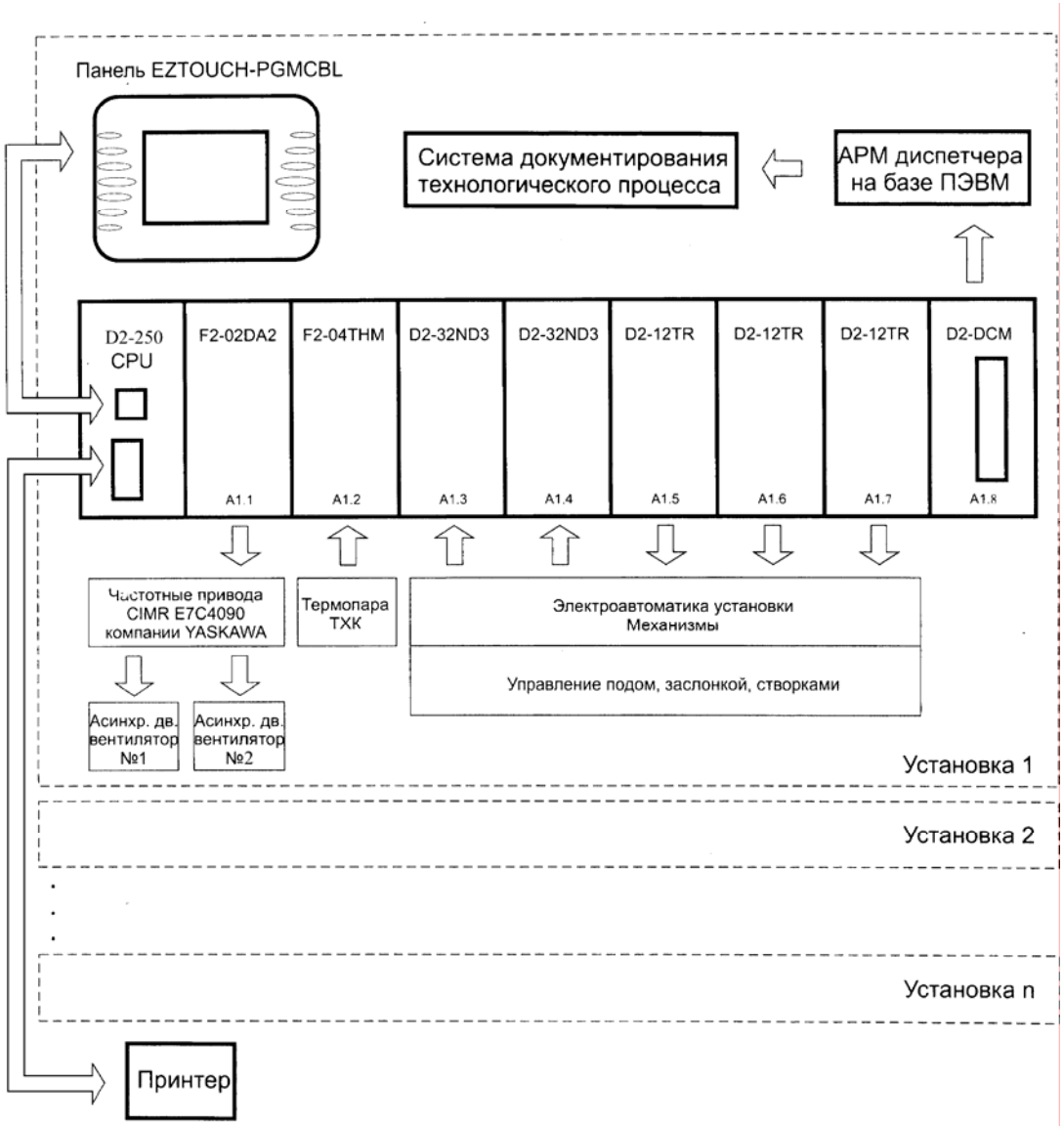
Контроллер реализует два режима регулирования температуры рабочей камеры: дроссельный (изменением площади поперечного сечения всасывающего отверстия) и качественный – изменением частоты вращения асинхронных двигателя мощностью 90 кВт каждый.

При дроссельном регулировании проходное отверстие жалюзийной решётки закрывается рядом приводных лопаток, которые при этом накладываются одна на другую. Лопатки получают движение от привода регулятора через приводной шток и переходные звенья. Для более «чувствительной» работы регулятора время полного движения лопаток составляет до 20 секунд.

Реализация алгоритма качественного регулирования температуры выполнена на базе следующих аппаратных средств контроллера:

- 4-канального аналогового модуля «F2-04THM»;

- 2-канального модуля «F2-02DA-2».



Структурная схема системы управления "ПАП-8"

Модуль «F2-04THM» конвертирует входной аналоговый сигнал от термопары типа ТХК в градусы Цельсия. Контроллер получает информацию о текущей температуре, обрабатывает с точностью 0,1 градуса Цельсия и обеспечивает первичную обработку (достоверность, выбраковка, усреднение) для формирования массива данных, включающего текущую и заданную температуры, скорость изменения температуры, производную скорости температуры. Данные массивов служат исходной информацией для алгоритма регулирования.

Управляющий сигнал с помощью цифроаналогового модуля «F2-02DA-2» преобразовывается в пропорциональный аналоговый сигнал 0...10V для управления частотно-регулируемыми приводами серии CIMR E7C4090 компании YASKAWA, обеспечивающие: мягкий пуск без электрических и механических перегрузок, регулируемое время ускорения и замедления, точное

поддержание требуемой скорости, возможность дистанционного управления, сопряжение с контроллером или компьютером.

Оптимизация работы привода с помощью регулирования скорости вращения ротора обеспечивает уменьшение потребляемой мощности и экономию энергии. При снижении скорости снижаются также момент, давление и механическая нагрузка на детали машины. Это увеличивает срок службы оборудования, сокращает потребность в ремонте и снижает расходы на техническое обслуживание.

Управление элементами вакуумной электрической автоматики реализовано на базе трёх модулей «D2-12TR» (36 каналов дискретных релейных выходов) и двух модулей «D2-32 ND3» (64 канала дискретного входов).

Для выполнения технологических задач системой управления используются три коммуникационных порта, два из которых содержит процессор D2-250, третий организован на базе модуля D2-DCM. Верхний порт процессорного модуля используется для соединения с сенсорной панелью EZTouch, обеспечивающей оперативное управление технологическим процессом термообработки: индикация параметров, номер ошибки при внештатной ситуации, рекомендаций оператору; ввод, просмотр и редактирование управляющих программ нагрева. К нижнему 15-контактному порту D2-250, поддерживающему интерфейсы RS-232, RS-485 и протоколы DirectNET ведущий/ведомый, MODBUS RTU ведущий/ведомый, подключён принтер, имеющий вход в формате ASCII.

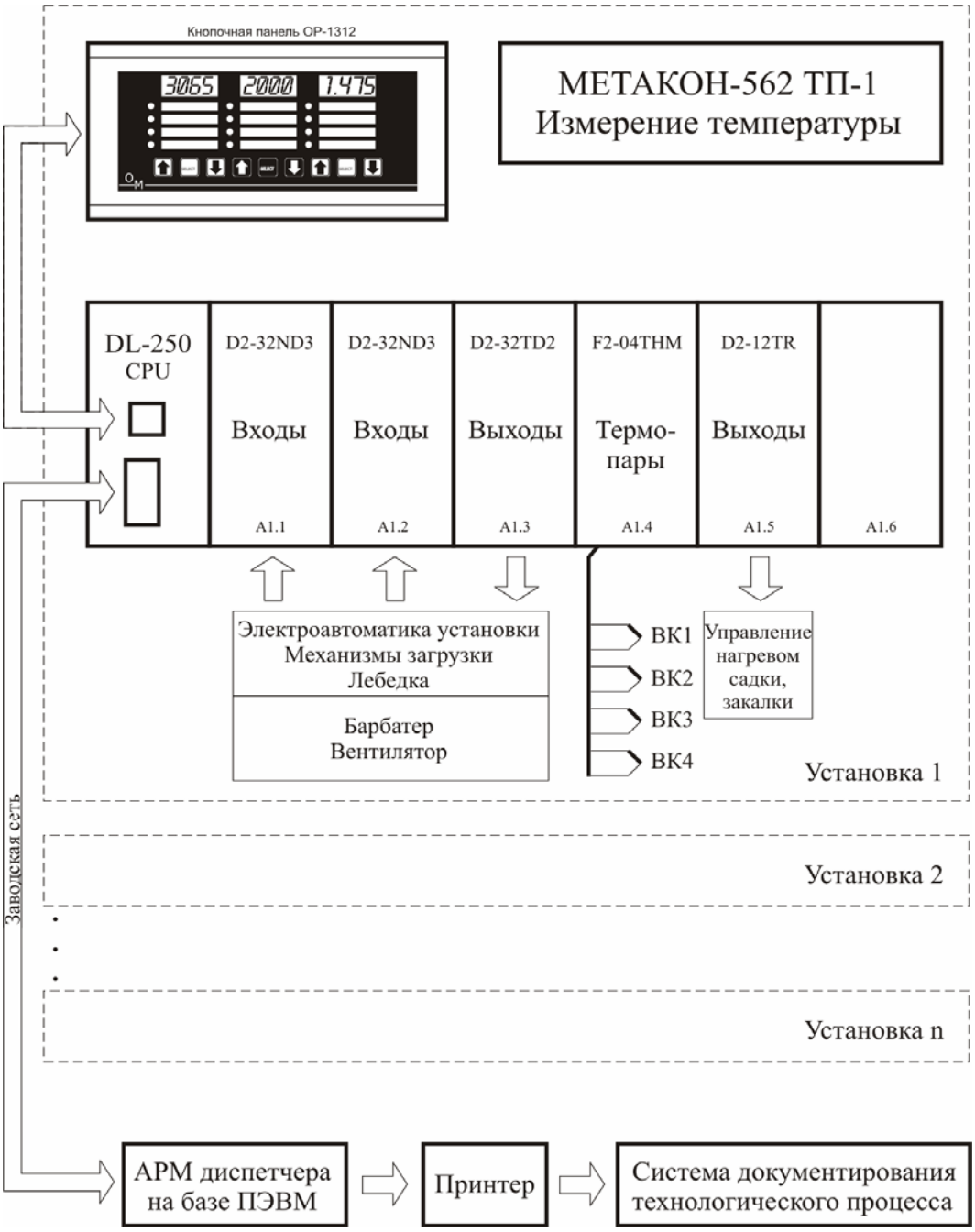
Модуль передачи данных D2-DCM организует универсальный коммуникационный порт для организации обмена данными между контроллером и персональным компьютером по протоколу DirectNET или MODBUS RTU. Модуль не нужно программировать, для его работы достаточно задать коммуникационные параметры и соединить устройства кабелями.

Верхний уровень СУ, построенный на базе персонального компьютера, разработан средствами языка Visual Basic 6 в среде операционной системы Windows 2000. Обмен информацией между ПЛК и ПЭВМ осуществляется с помощью DDE-сервера, который управляет обменом данных между программами верхнего и нижнего уровня. Система Windows 2000 показала при эксплуатации высокую надёжность и приемлемое быстродействие при обмене информацией через DDE-сервер.

После окончания каждого технологического процесса формируется отчёт, который записывается в отдельный файл и хранится на жёстком диске компьютера. Периодичность опроса и записи в память определяет оператор. Файл отчёта может быть вызван на монитор для визуального анализа или на печать. СУ обеспечивает также формирование файлов событий и внештатных ситуаций.

Аналогично выполнены СУ электротермической установки «ЭТА-2», предназначенной для термической обработки деталей из алюминия, и установки «УВС-4» для вакуумной аммиачной сушки керамических форм.

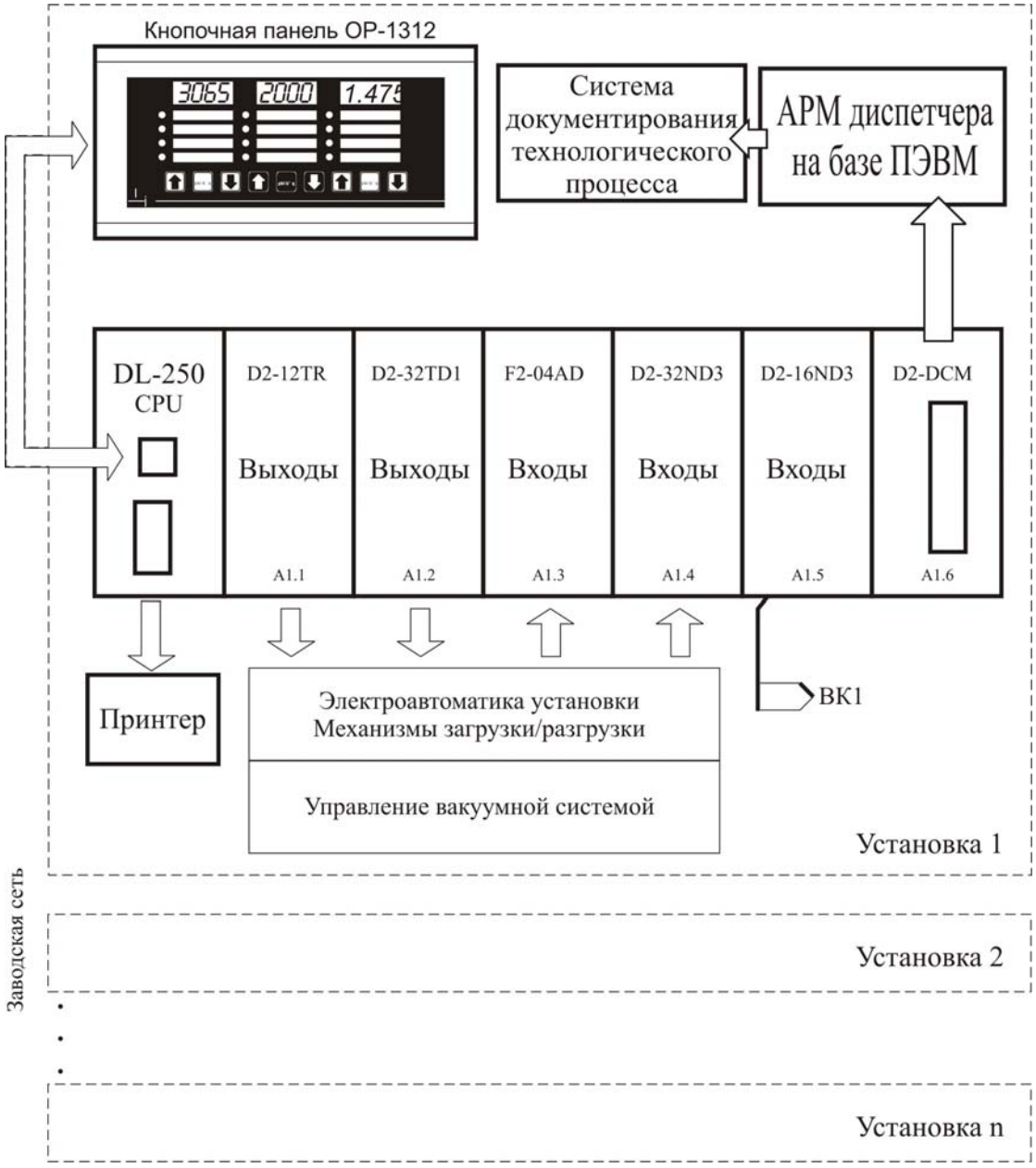
Отличительной особенностью СУ «ЭТА-2» от рассмотренной является использование для оперативного управления малогабаритного пульта управления DV-1000 с символьным дисплеем и набором клавиш. Пульт DV-1000 выполняет следующие функции: просмотр состояния памяти, просмотр состояния бита, изменение значений ячеек памяти, вывод определенных пользователем сообщений, управление битами, вывод системных и пользовательских сообщений об ошибках.



Структурная схема системы управления "ЭТА-2"

Программное обеспечение контроллера построено в соответствии с принципами многозадачности, открытости и гибкости.

Автоматизированная система управления установки «ЭТА-2» обеспечивает повышение эксплутационной надёжности за счёт строгого соблюдения технологии термообработки и предотвращения возникновения и развития неустойчивых и аварийных режимов.



Структурная схема системы управления "УВС-4"

Система управления «УВС-4» также реализована на базе контроллера Direct Logic семейства DL-205 с процессором D2-250.

Контроллер получает входные нормализованные сигналы (4 mA ... 20 mA) от датчиков температуры и давления, преобразует их значение

соответственно в градусы Цельсия и миллиметры ртутного столба, индицирует на малогабаритном пульте ОР-1312, выводит на печатающее устройство.

ПЛК обеспечивает работу установки в трёх режимах: «Ручное управление», «Программирование» и «Автоматический».

В режиме «Ручное управление» оператор имеет возможность управлять любым механизмом с соблюдением всех блокировок при некорректных действиях. Режим удобен для проведения пуско-наладочных, ремонтно-профилактических работ, завершения автоматического цикла в случае его сбоя.

Режим «Программирование» обеспечивает просмотр, ввод и редактирование технологических параметров. Для ввода технологических параметров и индикации текущих значений вакуума и температуры применяется панель ОР-1312. Окна дисплея содержат 7-сегментные жидкокристаллические индикаторы для отображения и редактирования данных.

В режиме «Автоматический» реализуется автоматический цикл вакуумной сушки.

Контроллер позволяет подключить принтер непосредственно к нижнему порту процессорного модуля D2-250, что позволяет в случае возникновения внештатной ситуации предоставить информацию о предыстории процесса.

2.2. Системы управления, построенные по структуре D2-250 (ведущий) – D2-250 (ведомый).

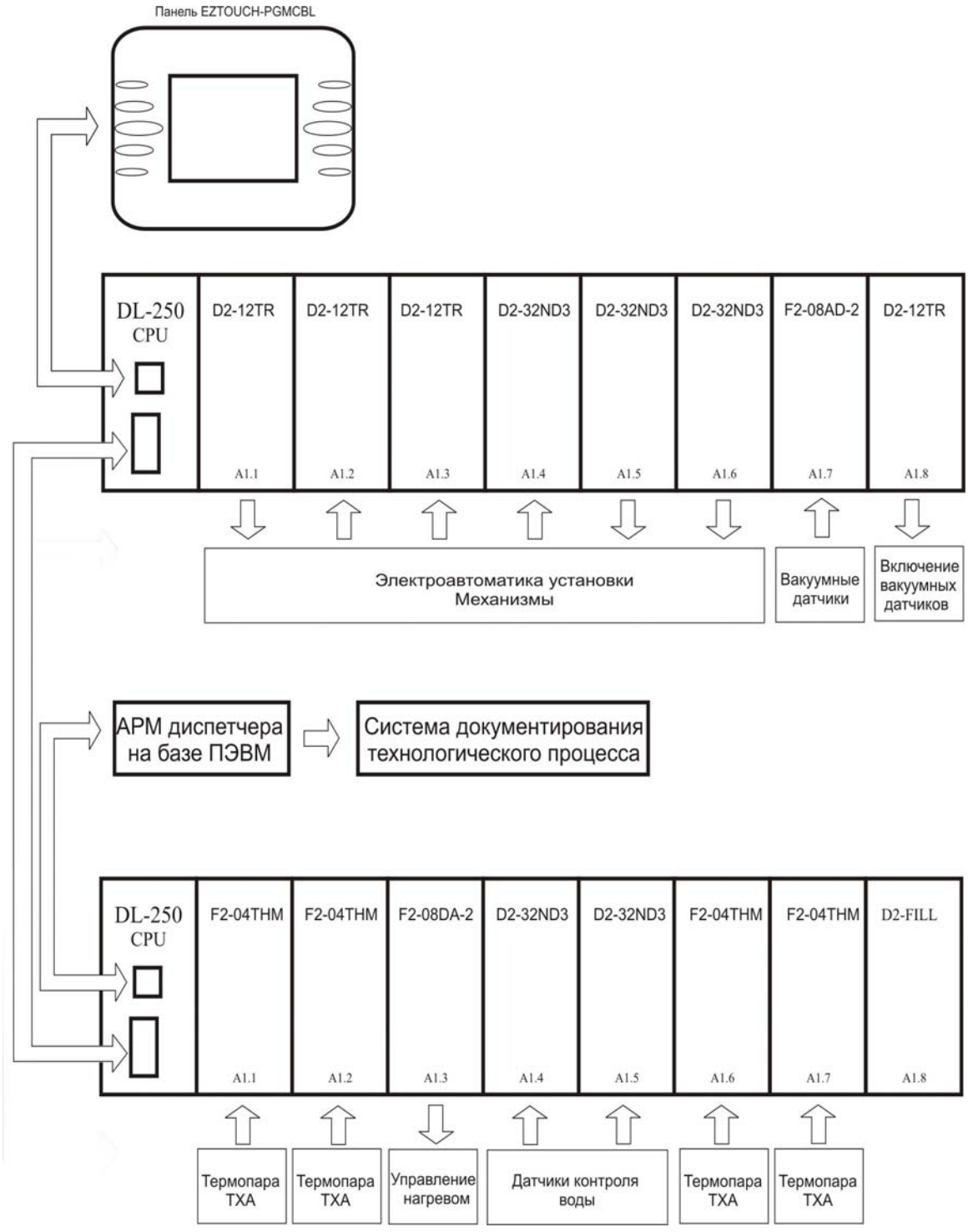
Печь вакуумная «ПВ-850» предназначена для реализации операции вакуумного отжига крупногабаритных сварных конструкций, что обеспечивает снижение содержания водорода до безопасного уровня и, как следствие, устранение склонности металла к водородной хрупкости; снятие остаточных напряжений; максимально возможное сохранение циклической прочности; нанесение защитных плёнок, предотвращение наводораживания в процессе эксплуатации.

Система управления «ПВ-850» является иерархической по информационной модели, функциям управления и архитектуре программного обеспечения. Организация связей и распределение задач между сетевыми ПЛК выполнены по территориальному и функциональному признакам. Такая архитектура позволяет легко адаптировать СУ к технологическому оборудованию любой сложности.

СУ установки «ПВ-850» построена на базе промышленного компьютера с сенсорным экраном AMB-655T компании ASTECH (верхний уровень) и сети промышленных контроллеров класса ПЛК Direct Logic DL-205 компании Koyo (нижний уровень). Обмен между контроллерами Direct logic семейства DL-205 осуществляется по протоколу DirectNET через нижний порт процессорного модуля D2-250.

СУ интегрирует весь поток информации: организацию интерфейса с оператором и технологом (терминальная задача); последовательно-параллельное управление механизмами вакуумной системы (логическая задача); программное управление процессом нагрева (технологическая задача), идентификацию состояния технологической системы (диагностическая задача), документирование технологического процесса (архивная задача), математическое моделирование технологического процесса (задача оптимизации), диспетчеризацию приведённых выше задач (системная задача).

Система управления обеспечивает работу установки в нескольких режимах: «Наладка», «Ручное управление», «Программирование», «Автоматический».



Структурная схема системы управления установки "ПВ-850»

В режиме «Наладка» управление механизмами установки осуществляется от мнемопанели пульта оператора. Режим реализован аппаратно, без контроллера, используется только при пуско-наладочных, ремонтно-профилактических работах и внештатных ситуациях.

Режим «Ручное управление» реализован на базе контроллера и обеспечивает безопасную эксплуатацию установки с соблюдением всех блокировок. Режим обеспечивает завершение автоматизированного цикла в случае его сбоя.

Режим «Программирование» обеспечивает просмотр, ввод и редактирование программ нагрева с любым количеством участков в графическом и цифровом виде. Оператор имеет возможность визуального контроля управляющей программы. Хранение программ организовано на жёстком диске промышленного компьютера.

Автоматический режим реализует рабочий цикл термической обработки при условии ввода оператором паспортных данных на изделия.

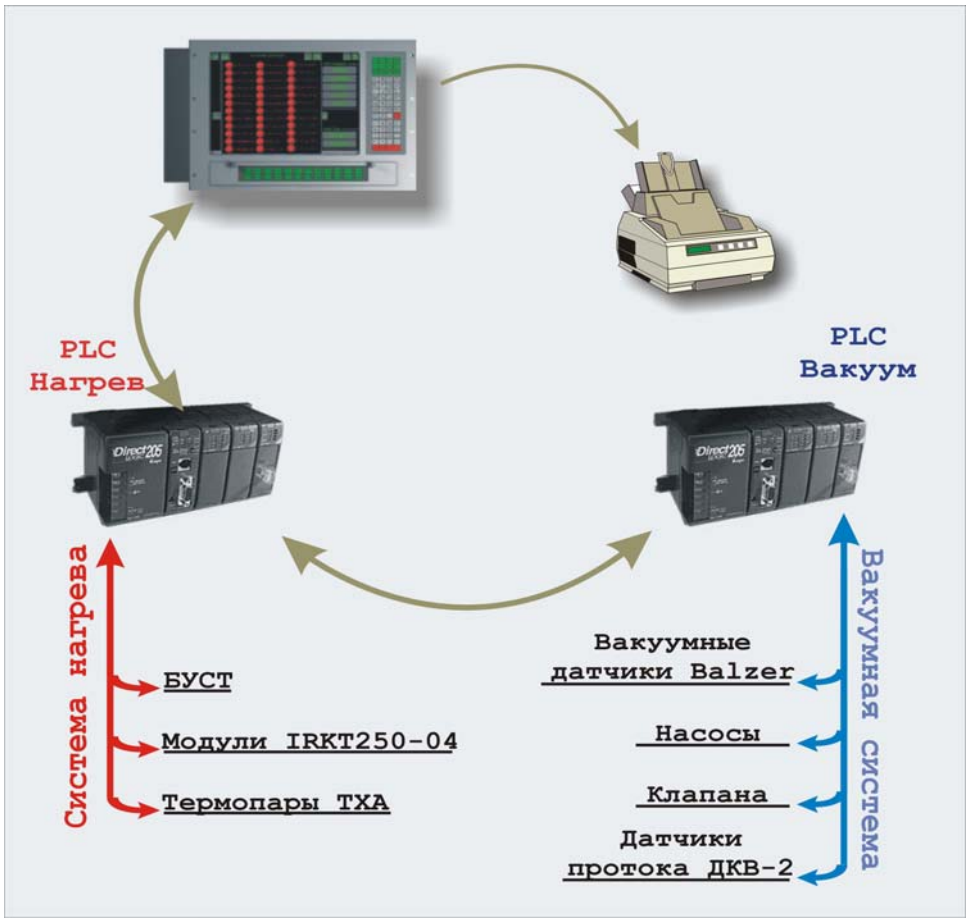
Программное обеспечение ПЛК обеспечивает многоконтурное регулирование температуры с любым количеством участков управляющей программы по ПИД-закону, расширенный диапазон изменения скорости нагрева и охлаждения от 25 до 200 градусов в час, адаптивное управление нарастанием температуры с организацией обратной связи по давлению – остановка нагрева до набора рабочего вакуума. Алгоритм автоматического поддержания вакуума сокращает время технологического процесса и оптимизирует его.

Алгоритм управления процессом вакуумного отжига, учитывающий обратные связи по давлению и температуре в рабочей камере, минимизирует нестабильность процесса, сокращает энергопотребление, увеличивает производительность.

Следует отметить преимущества СУ: самодостаточность и автономность вакуумной и нагревательной подсистем благодаря сетевой организации; фазовое управлением нагревом; открытость системы благодаря поддержке международного стандарта IEC 1131-3; резервирование функций управления, что позволяет довести технологический процесс в режимах "Ручное управление" (через контроллер), "Наладка" (аппаратная реализация) и значительно повышает безаварийность работы.

Системы управления вакуумных термических установок «УВН-1500М» и «УВН-1545М» структурно также построены по схеме D2-250 (ведущий) – D2-250 (ведомый). Верхний уровень СУ реализован на базе промышленного компьютера WS-855. При этом контроллеры вакуумной и нагревательной подсистем, как наиболее надёжный элемент, выполняют функции регулирования, выравнивания температур по шести зонам установки, блокировок и аварийной защиты. Вероятность отказа контроллера DL-205 намного меньше вероятности отказа компьютера. Компьютер выполняет терминальную задачу управления – визуализация технологического процесса отжига; ввод, редактирование и запись программ нагрева; документирование и хранение файлов истории процесса, диагностики и ошибок. Отказ компьютера не приводит к потере управления процессом.

Регулятор для управления нагревом реализован на базе тиристорных модулей IRKT250-04 (IR) и блока управления тиристорами «БУСТ», обеспечивающего два режима управления: по числу полупериодов и фазовый, устройство контроля перехода через ноль формирует импульс в начале каждого полупериода соответствующей фазы, устройство контроля тока обеспечивает защитное отключение нагрузки при превышении установленной величины.



Структурная схема системы управления установки "УВН-1500М»

Аналогично построена система управления вакуумной литейной установки «УВНК-9», в состав которой входят три ПЛК семейства DL-205 с процессором D2-250.

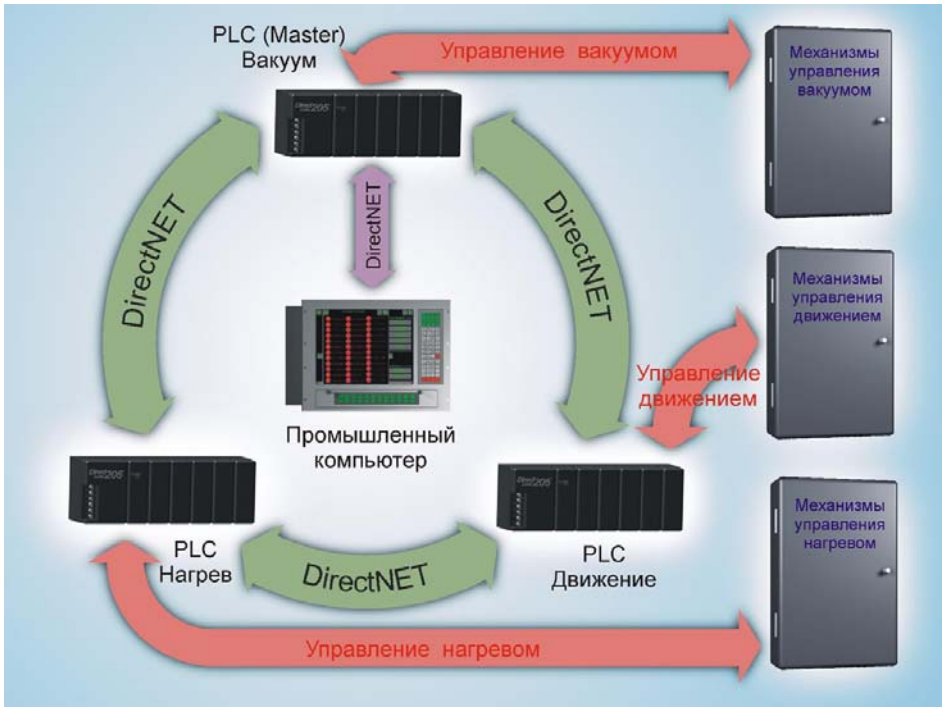
Обмен между контроллерами Direct logic осуществляется по протоколу DirectNET через сетевой порт 2 процессорных модулей.

Организация связей обеспечивает взаимодействие между следующими элементами СУ:

- промышленным компьютером и сетью ПЛК;
- ПЛК №1, ПЛК №2 и ПЛК №3;
- ПЛК №1 и элементами вакуумной подсистемы;
- ПЛК №2 и приводами перемещения форм;
- ПЛК №3 и подсистемой нагрева.

Основными технологическими параметрами управления процесса вакуумного литья являются:

- температура верхней и нижней зон печи подогрева форм;
- температура заливаемого в форму расплава;
- скорость заполнения формы изделия;
- скорость перемещения формы из зоны нагрева в зону охлаждения;
- градиент температур на фронте роста;
- высота твёрдожидкой зоны;
- степень вакуума рабочей, шлюзовой и загрузочной камер.



Структурная схема системы управления »УВНК-9»

Автоматизированная система управления «УВНК-9» позволила реализовать адекватные процессу направленного затвердевания математические модели, собирать и обрабатывать статистическую информацию на базе архивных файлов о ходе технологического процесса.

2.3. Система управления, построенная по структуре D2-250 – RSSS.

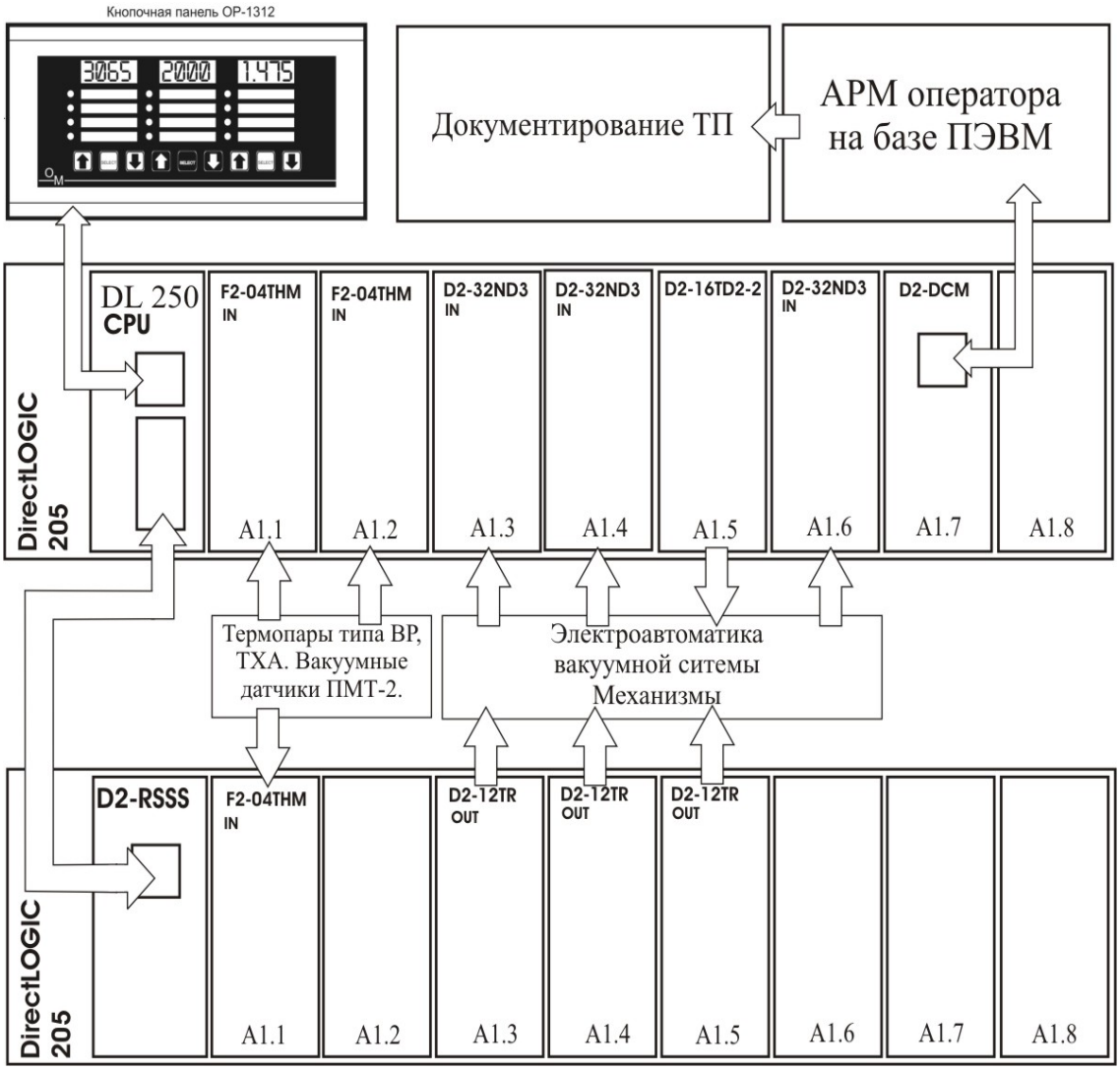
Система управления вакуумной установки «УППФ-3МКН», предназначенной для равноосного литья лопаток газотурбинных двигателей, обеспечивает надёжную непрерывную работу в условиях литейного производства. Основой СУ является микроконтроллер Direct Logic серии DL-205 с процессором D2-250, операционная система которого имеет встроенный журнал диагностики, поддерживает сети с протоколами K-sequence, DirectNET^{TM} master/slave, MODBUS RTU master/slave.

Сеть с протоколами DirectNET и MODBUS представляет эффективное и надёжное решение для задачи взаимодействия на цеховом уровне. При организации автоматизированного рабочего места диспетчера участка вакуумного литья существенно не время реакции, а функциональные возможности.

Характеристики сети с протоколами DirectNET и MODBUS:

- средства передачи – экранированная витая пара;

- физический интерфейс - RS-422, RS-485;
- длина линии до 1200 метров без повторителей;
- скорость передачи до 38400 бод;
- метод доступа смешанный: master/slave («ведущий-ведомый») между активными и пассивными узлами.



Структурная схема системы управления установки "УППФ-3МКН»

Сигналы ручного управления реализованы через контроллер DL-205. Надёжность канала не уступает применяемой ранее релейной схеме управления, при этом контроллер отслеживает ручные воздействия, учитывая их при автоматизированном управлении и диагностике, исправляя ошибки оператора.

Для оперативной индикации основных параметров технологического процесса литья применяется панель «OP-1312», на которой в цифровом виде индицируются основные параметры процесса плавки:

- температура заливаемого металла;
- давление в камере при заливке;
- текущее время;
- время выдержки залитой формы в вакууме;
- скорость заливки металла;

- натекание в объём печи;
- время от установки формы в форвакуумную печь до заливки.

Прикладная программа контроллера DL-205 организует работу установки в нескольких режимах:

✓ наладочном, обеспечивающим управление механизмами установки от переключателей пульта оператора;

✓ ручном, дополняющим наладочный режим советчиком оператора-технолога, блокирующим некорректные действия оператора;

✓ автоматизированном, осуществляющим рабочий цикл вакуумного литья при условии ввода оператором паспортных данных на выплавляемые лопатки: марки материала, номера плавки, индивидуального номера лопатки.

Прикладные программы контроллера разработаны средствами языка RLL^{PLUS} на базе стандарта Международной Электрической Комиссии (МЭК) IEC 1131-4 с широким использованием графических средств программного пакета DirectSOFT под Windows 95/98/NT/2000.

Автоматизированное рабочее место диспетчера, реализованное на базе персонального компьютера, позволяет собирать и обрабатывать информацию о процессе вакуумного литья с нескольких установок типа «УППФ-3МКН».

2.4. Система управления, построенная на базе процессора D2-260.

Установка «ВИП-НК» предназначена для получения изделий из жаропрочных сплавов с равноосной, направленной и монокристаллической структурой, с различной кристаллографической ориентацией.

Система управления установки, построенная по структуре ПЛК - ПЭВМ, включает логический программируемый контроллер семейства DL-205 с процессором D2-260 и промышленный компьютер с сенсорным монитором.

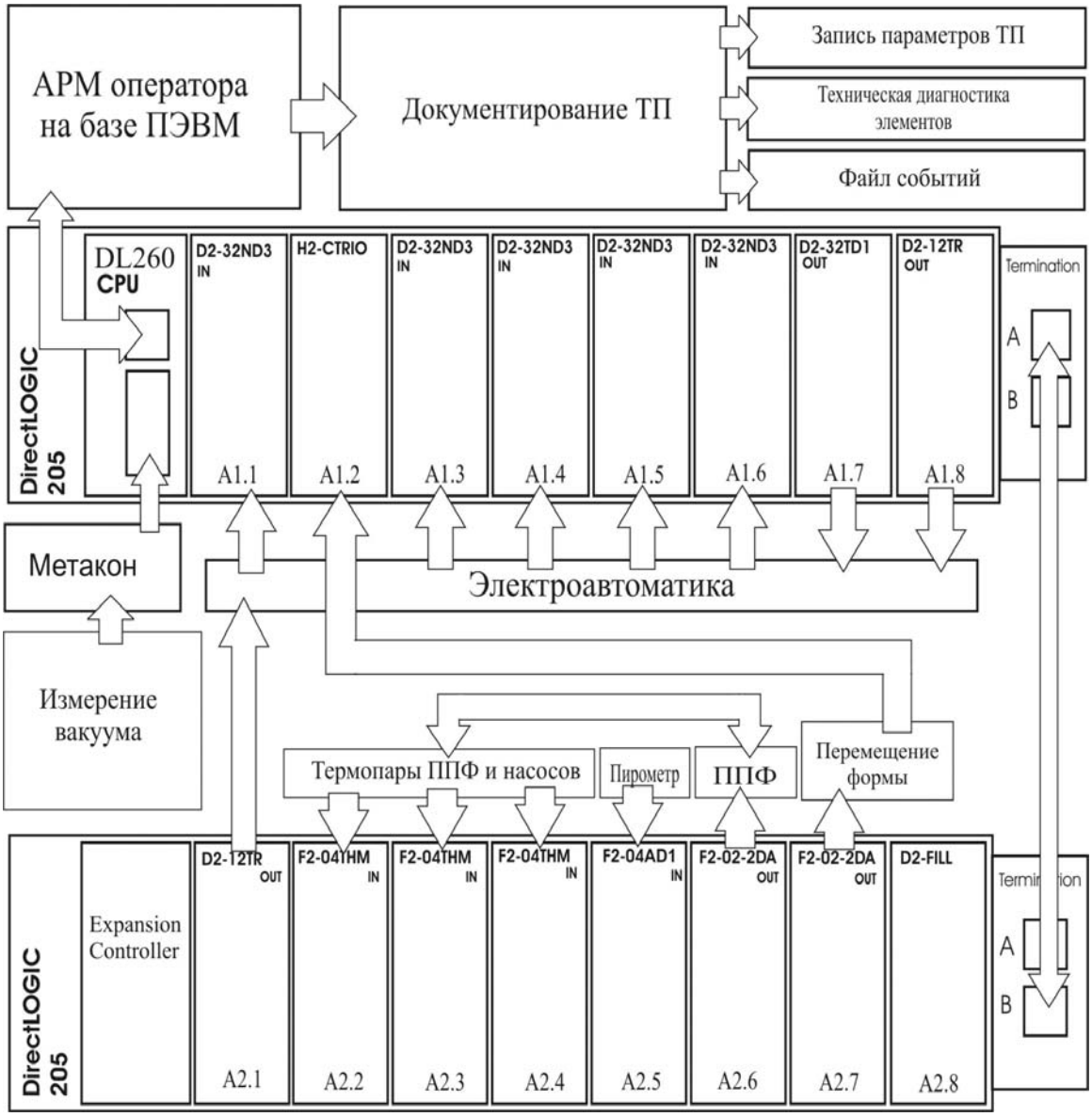
Контроллер выполняет все функции управления технологическим процессом, компьютер - математическое моделирование процесса направленной кристаллизации и терминальную задачу управления – ввод, редактирование, запись программ нагрева и движения, визуализация состояния элементов технологического оборудования, хранение файлов истории технологического процесса, диагностики и ошибок.

Контроллер состоит из двух каркасов, связанных между собой с помощью модулей связи D2-EM Termination.

Процессор D2-260 – самый мощный в серии DL-205, отличающийся возможностью работы с четырьмя удалёнными каркасами расширения, расстояние между которыми может составлять до 32 метров. Встроенные порты обеспечивают поддержку следующих протоколов: K-sequence, DirectNET master/slave, MODBUS RTU master/slave, ASCII In/Out, Remote I/O (удалённый ввод-вывод). Полный набор из 225 команд включает работу с числами в формате с плавающей запятой, таблицами, вычисление тригонометрических функций; чтение, запись, поиск, сравнение, замену и выделение данных в ASCII кодах. Процессор позволяет реализовать закон автоматического управления практически любой сложности: ПИД с самонастройкой, ПИД с элементами нечёткого регулирования, адаптивное управление.

Для реализации геометрической задачи управления используются модуль «H2-CTRIO», позволяющий обрабатывать сигналы частотой до 100 кГц от двух фотодатчиков типа ЛИР; модуль цифроаналогового преобразователя «F2-02DA-2», обеспечивающий выдачу управляющих аналоговых сигналов

±10V на электрические приводы наклона тигля и вертикального перемещения форм.



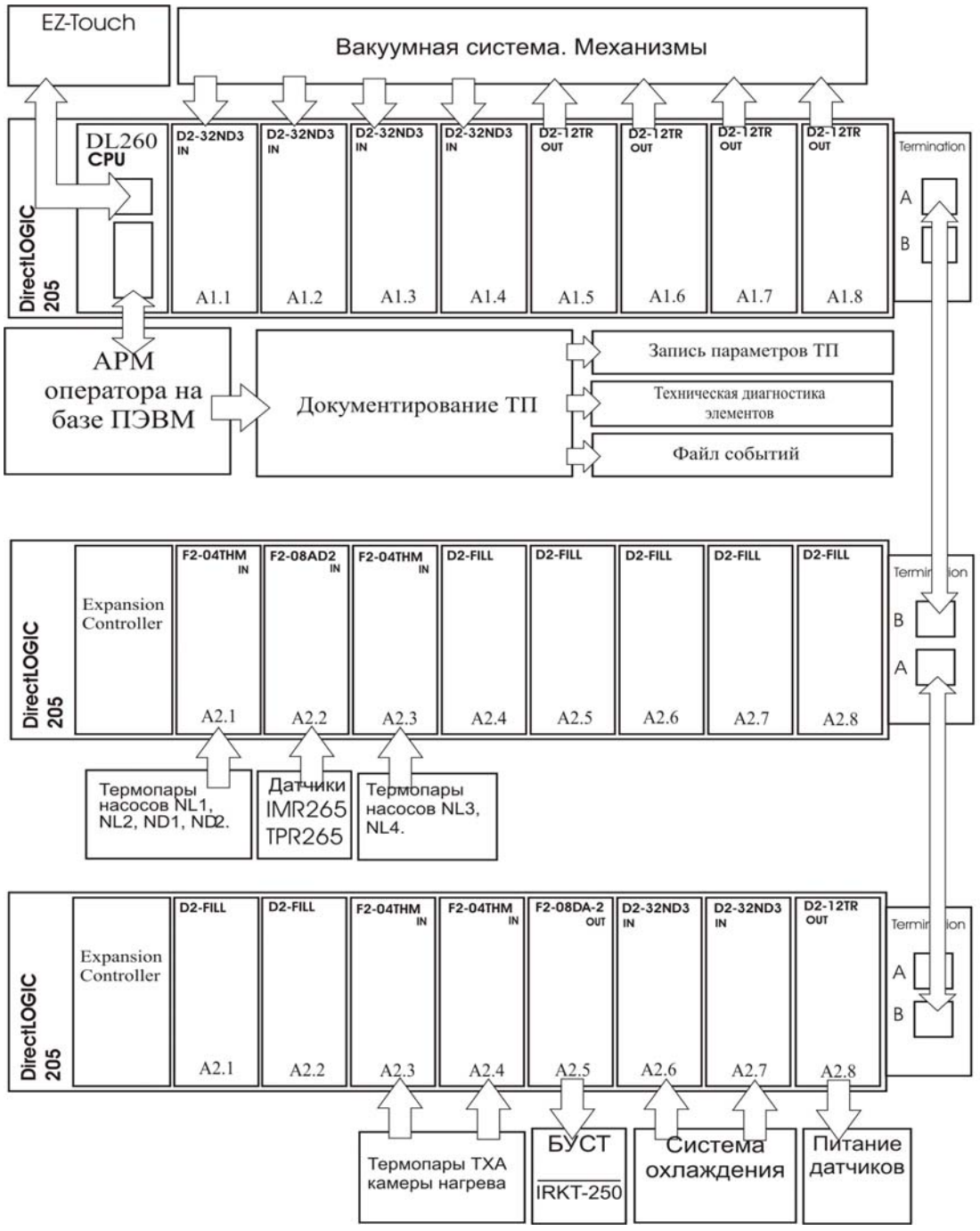
Структурная схема системы управления «ВИП НК»

Для технологической задачи (нагрев печи форм) применяются модуль «F2-04THM», обрабатывающий информацию от четырёх термопар типа ВР; «F2-02DA-2», осуществляющий аналоговое управление двумя блоками «БУСТ; для логической задачи – модули «D2-32ND3» (ввод дискретных сигналов) и «D2-12TR» (вывод релейных дискретных сигналов). Модуль «F2-04AD-1» принимает сигнал от пирометра «Луч» 4-20mA.

Контроллер принимает сигналы от пяти термопар, измеряющих температуру верхней и нижней зон печи форм и кристаллизатора, расплавленного металла. Каждый сигнал подвергается стандартной математической обработке: контроль на достоверность, масштабирование, выбраковка ложных измерений. Кроме этого, между контроллером и прибором для измерения вакуума «Метакон-562» осуществляется обмен информацией

через нижний сетевой порт процессорного модуля D2-260. Прибор обеспечивает измерение давления в шести точках вакуумной системы установки. Обмен информацией между прибором «Метакон-562» и ПЛК осуществляется по протоколу ASCII In/Out.

АРМ оператора реализовано на базе промышленного компьютера. Задание управляющих воздействий и ввод программируемых параметров технологического процесса литья осуществляется с помощью клавиш, отображаемых на мониторе.



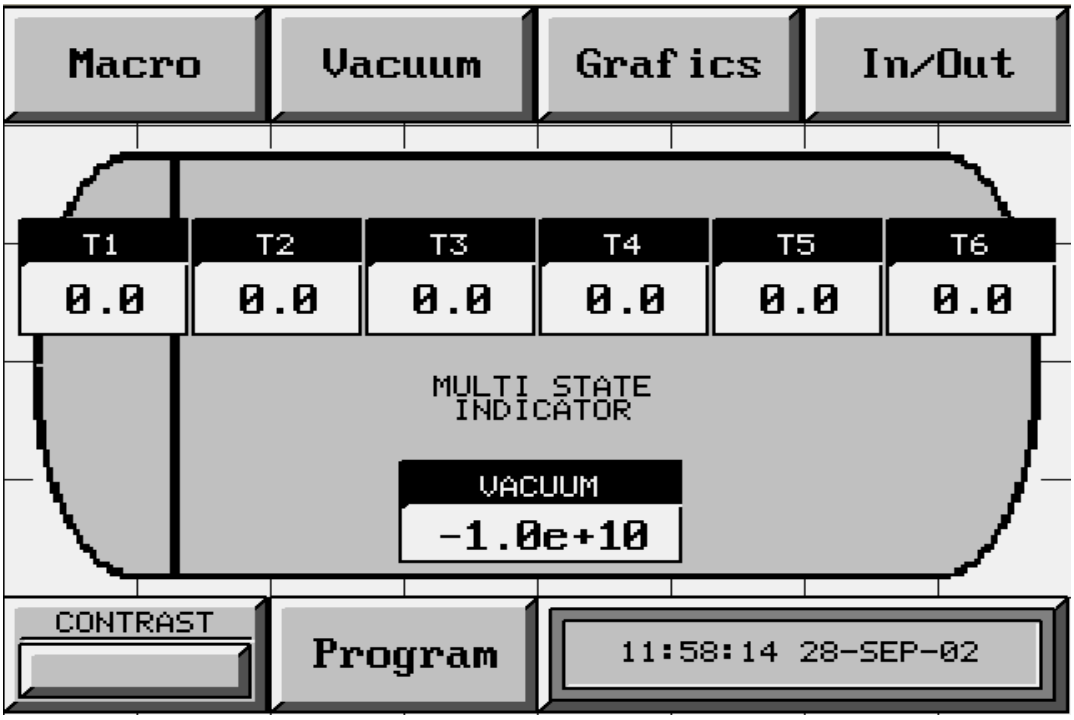
Структурная схема системы управления «ПВ-900»

Система управления вакуумной установки «ПВ-900», предназначенной для термической обработки крупногабаритных титановых конструкций весом до 3000 килограмм, построена на базе трёх каркасов D2-09B. Процессорный модуль D2-260 обеспечивает логические и технологические функции управления. Каркасы №1 и №2 расположены в электрическом шкафу автоматики и обеспечивают управление элементами вакуумной системы. В первом каркасе расположены модули для реализации логических функций управления, на втором – модули для обработки информации от шести датчиков температуры (термопары типа ТХА) и вакуумных датчиков IMR-265 и TPR-265 компании Pfeiffer.

Терминальная задача в составе системы управления термической установки во многом определяет функциональные возможности и удобство управления. К верхнему порту процессорного модуля подключена сенсорная панель EZTouch, программное обеспечение которой обеспечивает многооконный интерфейс оператора.

Разработка интерфейса предполагает: создание базового решения, реализацию экранов, разработку интерпретатора диалога, организацию информационных связей.

После подачи питания на EZTouch Panel появляется основное окно «Параметры технологическое процесса», которое отображает значения давления температуры в шести зонах рабочей камеры установки, клавиши «Макрооперации», «Вакуум», «График», «Вход/Выход», «Управляющая программа», «Контрастность», текущее время и дату.

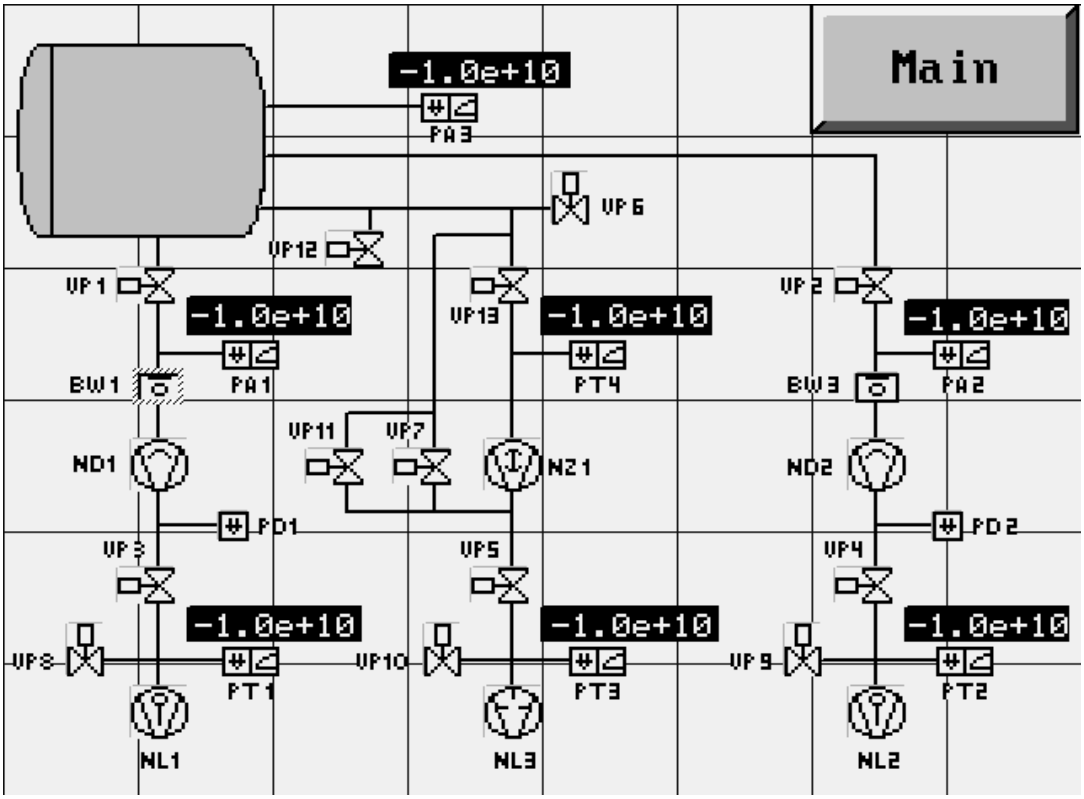


Окно «Параметры технологического процесса»l

Из основного окна можно вызвать 5 вспомогательных экранов, с помощью которых оператором даются команды включения и выключения макроопераций, индикации состояния дискретных входов и выходов и т.п.

Окно «Макрооперации» обеспечивает возможность выбора макрооперации технологического процесса: откачка форвакуума, выключение форвакуумной линии, подготовка диффузионных насосов ND, выключение диффузионных насосов ND, включение цикла вакуумной откачки, включение нагрева, включение автоматического цикла.

Окно «Вакуум» содержит мнемосхему вакуумной системы «ПВ-900», на которой отображается состояние основных элементов (насосы и клапаны) и текущие значения давления в семи точках вакуумной схемы.



Окно «Вакуум»

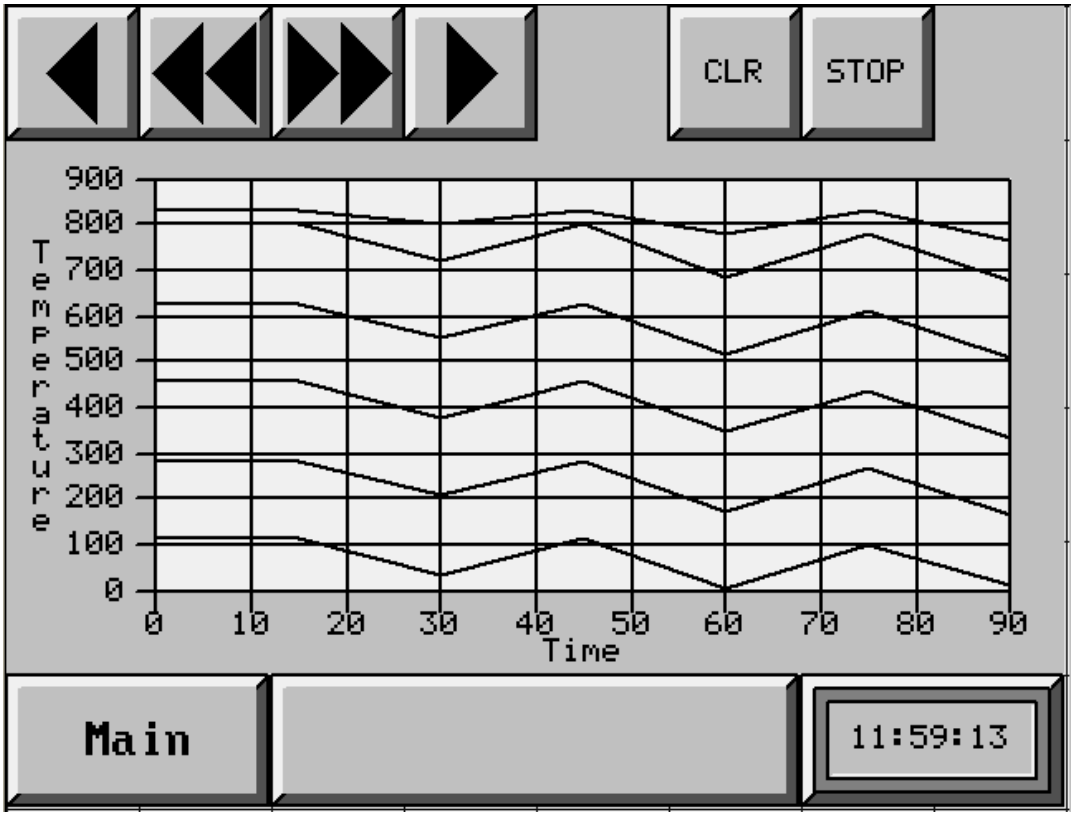
Окно «График» отображает графическую зависимость температуры и давления от времени.

Окна «Входы» и «Выходы» предназначены для диагностики состояния всех дискретных входов и выходов контроллера.

Окно «Управляющая программа» предназначено для ввода программы нагрева: количество участков, скорость нагрева или охлаждения, температура и время выдержки на каждом участке.

Таким образом, сенсорная панель может заменить персональный компьютер промышленного исполнения для решения терминальной задачи управления. Единственной нерешённой задачей в этом случае остаётся запись и хранение больших объёмов значений параметров технологического процесса в памяти и их отображение в графическом виде.

Третий каркас D2-09B, расположенный в электрическом шкафу нагрева, содержит модули F2-04THM, F2-08DA-2, D2-32 ND3, D2-12TR для реализации алгоритма регулирования температуры. Алгоритм обеспечивает выравнивание температуры по шести зонам с точностью 1ºС на участках «Нарастание», «Выдержка», «Охлаждение».



Окно «График»

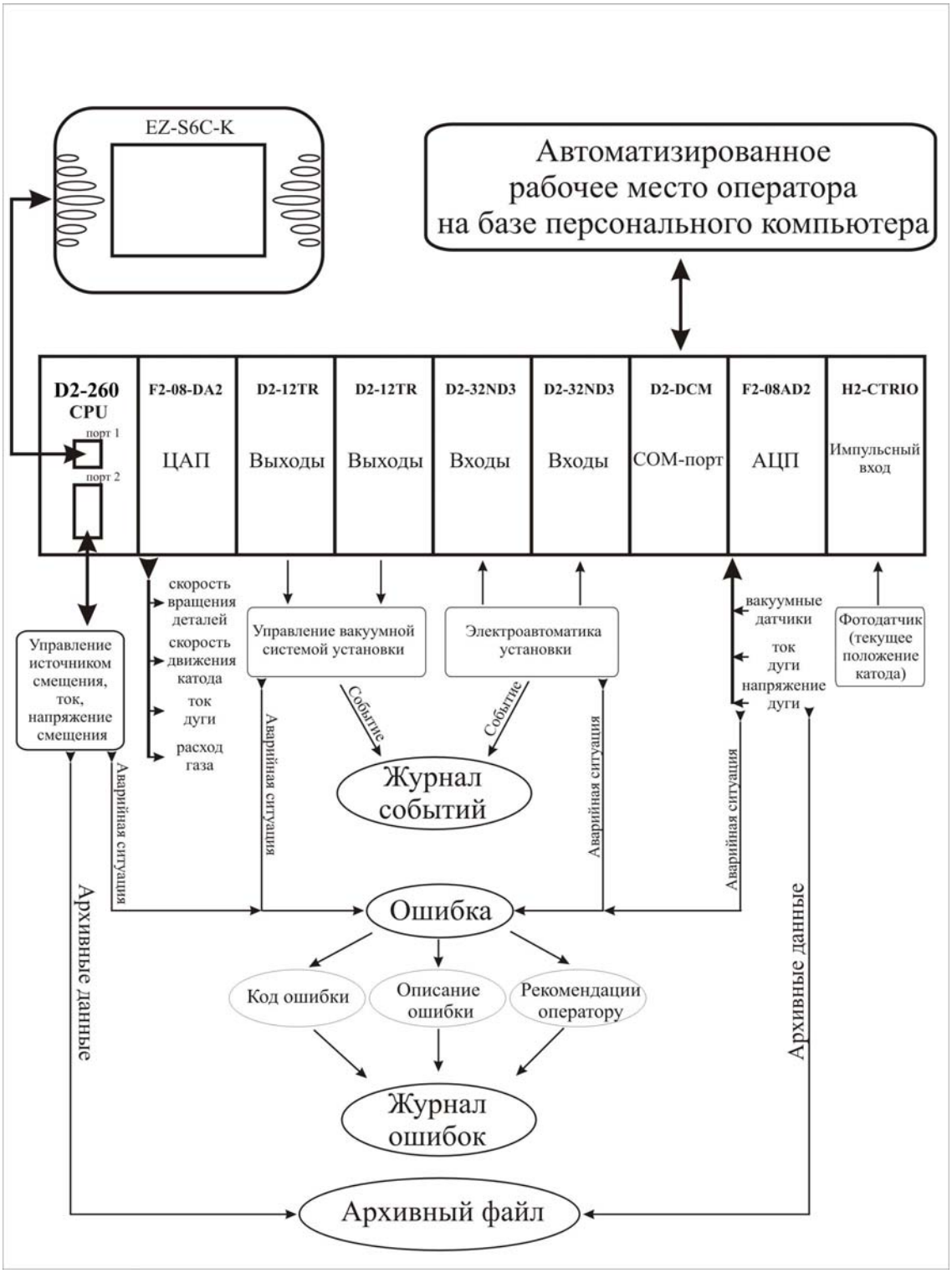
Система управления агрегата «АПН-250», предназначенного для нанесения защитных, жаростойких, износостойких покрытий ионно-плазменным методом, включает логический программируемый контроллер семейства DL-205 компании PCL Direct by Koyo Inc и персональный компьютер. Контроллер реализует последовательно-параллельное управление механизмами вакуумной системы (логическая задача), управление процессом напыления (технологическая задача), идентификацию состояния технологической системы (диагностическая задача) и диспетчеризацию приведённых выше задач (системная задача).

Кроме этого, СУ агрегата «АПН-250» выполняет геометрическую задачу: программное управление скоростью вращения изделий и перемещением катода в вертикальном направлении. Регулируемые электромеханические приводы реализованы на базе асинхронного двигателя, частотного регулируемого привода серии VS-606V7 компании YASKAWA, фотодатчика обратной связи по положению ЛИР.

Для реализации алгоритма перемещения применяется двухканальный модуль обработки сигналов фотодатчика «H2-CTRIO» и модуль ЦАП «F2-08DA-2». Управляющий сигнал с помощью модуля «F2-08DA-2» преобразовывается в пропорциональный аналоговый сигнал 0...10V для управления частотно-регулируемыми приводами.

Для работы в автоматическом цикле оператору необходимо задать координаты диапазона перемещения катода, скорость движения катода,

скорость вращения изделий, и осуществить операцию «Нулирование» по вертикальной оси.



Структурная схема системы управления «АПН-250»

Многооконный интерфейс оператора цветной сенсорной панели EZ-S6C-K, разработанный на базе пакета EZTouch ™ Programming Software (Version 2.2), обеспечивает реализацию следующих функций:
- визуализацию значений технологических параметров и состояния механизмов;
- ввод, просмотр и редактирование программы напыления;
- автоматическое формирование оперативных сообщений на основе анализа аварийных и внештатных ситуаций;
- диагностика ПЛК.

Интерфейс включает следующие окна: «Главное меню», «Откачка вакуума», «Макрооперации», «Рабочий экран», «Вакуумные датчики», «Графики», «Отладочный экран».

Окно «Макрооперации»

Окно «Рабочий экран»

Окно «Отладочный экран» позволяет перейти к выбору отладочных операций:
- «Движение»;
- «Проверка дуги»;
- «Натекание»;
- «Входы /Выходы».

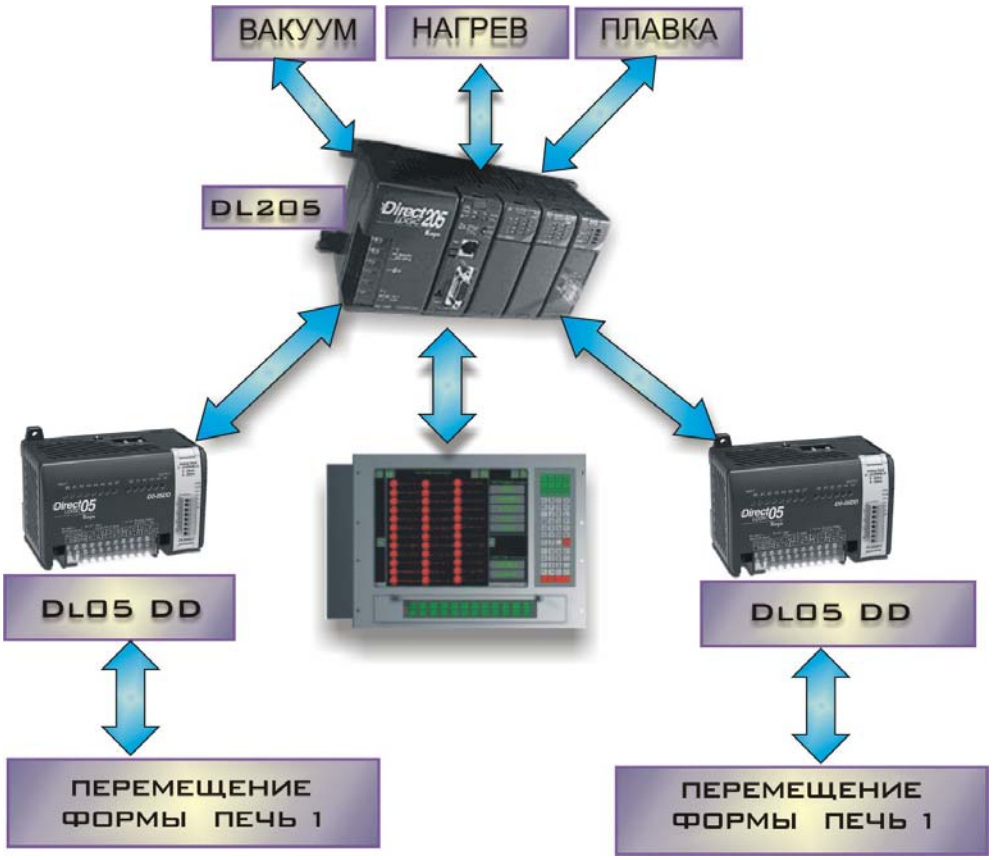
Связь контроллера с источником смещения БСМ (регулируемое напряжение 20 – 600 В, мощность 25 кВт), подающим отрицательное напряжение на изделия, осуществляется по протоколу ASCII.

Система управления «АПН-250» во время работы производит регистрацию и хранение параметров технологического процесса с привязкой к реальному времени, контроль событий, ошибок и аварийных ситуаций с записью на электронный носитель с функциями «чёрного» ящика» посредством создания архивного файла, файла параметров технологического процесса напыления, файла событий, файла ошибок.

2.5. Система управления, построенная на базе сети контроллеров семейств DL-205 и DL-05/DL-06.

Установка «ВПДС-1» относится к новому поколению вакуумных плавильных установок с донным сливом, оснащённых системой управления на базе компьютерной технологии. Система управления установки «ВПДС-1» построена на базе промышленного компьютера (верхний уровень) и сети промышленных программируемых логических контроллеров (нижний уровень).

Управление работой на нижнем уровне осуществляется от сертифицированных сетевых контроллеров DL-205 (ПЛК №1) и DL-05DD (ПЛК №2 и ПЛК №3) компании PLC Direct by Koyo Inc. Обмен между контроллерами Direct logic осуществляется по протоколу MODBUS.



Структурная схема системы управления установкой "ВПДС-1"

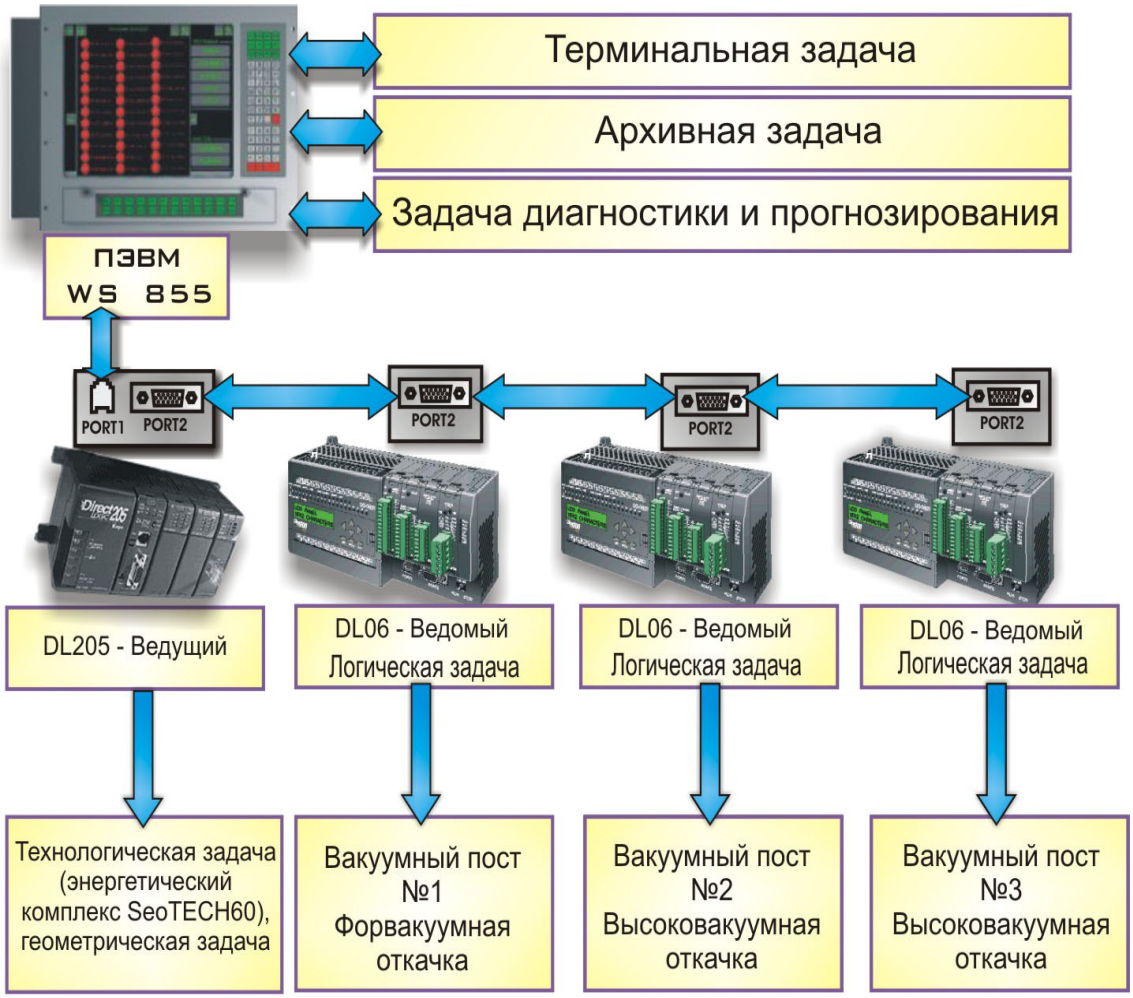
ПЛК №1 управляет процессами нагрева форм, плавки металла в тигле и вакуумной откачки. Непрерывный контроль температуры расплава осуществляется термопарой, инертной к воздействию магнитного поля индуктора. Это позволяет обеспечить автоматизацию технологического процесса: получение вакуума, нагрев формы, плавка и заливка металла, кристаллизация отливки.

ПЛК №2 и ПЛК №3 обеспечивают управление перемещением форм в обеих печах установки для формирования направленной структуры отливок. В качестве датчиков обратной связи по перемещению применяются фотодатчики типа ЛИР-158В.

Таким образом, контроллеры выполняют функции управления технологическим процессом вакуумного литья, компьютер - математическое моделирование процесса и терминальную задачу управления (ввод, редактирование, запись программ нагрева и движения, визуализация состояния элементов технологического оборудования, хранение файлов истории технологического процесса).

Верхний уровень СУ, построенный на базе промышленного компьютера AMB-655T компании ASTECH, работает под управлением операционной системы Windows 2000. С помощью компьютера оператор не только контролирует последовательность процесса, но и имеет возможность оперативно изменять параметры управления, просматривать архивные данные.

Традиционным направлением работы ОАО «Электромеханика» является разработка и изготовление технологического оборудования для реализации электронно-лучевых технологий.



Структурная схема системы управления «ЭЛУ ПМ»

Последней разработкой предприятия в области оборудования для реализации электронно-лучевых технологий является установка «ЭЛУ ПМ», предназначенная для сварки и термической обработки цилиндрических и конических заготовок из нержавеющей стали и жаропрочных сплавов двумя электронно-лучевыми пушками с вертикальным направлением луча сверху вниз для сварки круговых, кольцевых, продольных швов и с горизонтальным направлением луча поперек камеры для сварки кольцевых швов.

В состав сварочной установки «ЭЛУ ПМ» входят следующие узлы и механизмы:

- рабочая камера;
- манипулятор;

- механизм перемещения поперек камеры по оси Y;
- механизм вертикального перемещения по оси Z;
- тележка для перемещения изделия вдоль камеры по оси X;
- внешняя станина для загрузки и выгрузки изделий;
- вакуумная система;
- система выхлопа газов;
- система подачи воздуха;
- система водоснабжения;
- системы подвода энергопитания;
- шкаф электрической автоматики;
- автоматизированное рабочее место оператора;
- энергетический комплекс SeoTECH-60.

Нижний уровень СУ построен по структуре DL-205 (ведущий)-DL-06 (ведомые).

ПЛК DL-205 с процессором D2-260 обеспечивает управление станочным комплексом (геометрическая задача) и задание режима работы энергетического комплекса «SeoTECH-60» (технологическая задача).

Устройством рабочего перемещения вдоль камеры по оси X является стол с установленным на нем манипулятором с осью вращения шпинделя. Манипулятор обеспечивает наклон оси шпинделя в плоскости X-Z.

Тележка с манипулятором перемещается по направляющим вдоль оси X с рабочей скоростью 10…40 м/час при помощи реечной передачи, а также перемещается из камеры на внешнюю станину. Это необходимо для проведения процессов выгрузки готовых изделий и загрузки новых деталей.

Кроме этого, станочный комплекс содержит устройство настроечных перемещений манипулятора по осям Y и Z. Эти перемещения обеспечивают настройку на рабочее расстояние от торца пушки до поверхности стыка. В случае размещения пушек в карманах вакуумной камеры предусмотрен их установочный ход на расстояние 400 мм соответственно по оси Z (вертикальная пушка) и оси Y (горизонтальная пушка).

DL-205 обеспечивает управление перемещением изделия и пушки. В качестве датчиков обратной связи по перемещению применяются фотодатчики типа ЛИР-158В.

Энергетический комплекс «SeoTECH-60» включает:

- сканер, работающий в режиме отраженных электронов и позволяющий осуществлять точное наведение луча на шов свариваемых деталей и получать качественное видеоизображение зоны сварного шва;
- устройство развертки луча по окружности, полуокружности, эллипсу, линии и прямоугольному растру;
- две электронно-лучевые пушки с турбомолекулярными насосами.

Управление работой вакуумной системы, состоящей из трёх автономных откачных постов, осуществляется от контроллеров DL-06.

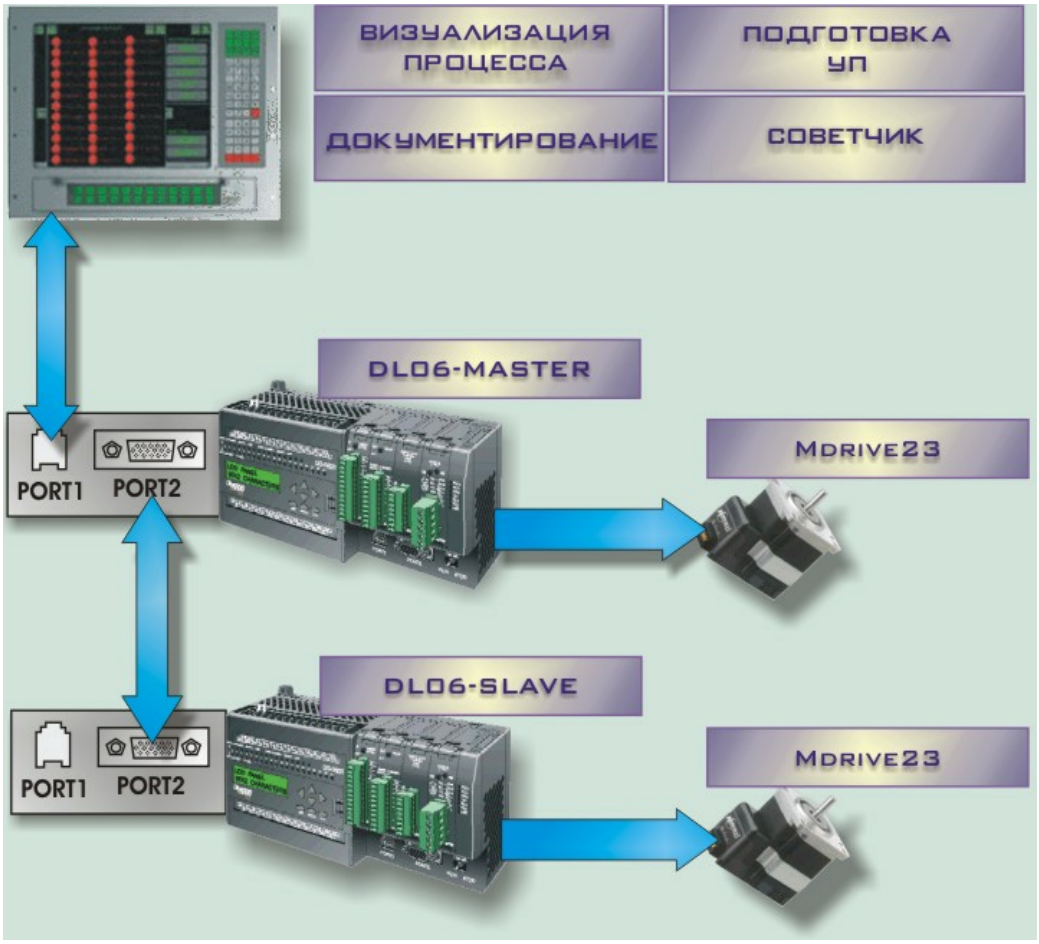
В состав поста №1, предназначенного для получения низкого вакуума, входят двухроторный насос «2ДВН-1500» (насос Рутса), механический насос «НВЗ-300», пневматические клапаны, датчики для измерения вакуума, система охлаждения, система управления на базе контроллера DL-06.

В состав постов №2 и №3, предназначенных для получения высокого вакуума, входят механические насосы «АВЗ-180» и диффузионные насосы «НД-500», хладоновые ловушки, пневматические клапаны, датчики для измерения вакуума, система охлаждения, системы управления на базе контроллеров DL-06.

АРМ оператора реализовано на базе промышленного компьютера с сенсорным экраном. Задание управляющих воздействий и ввод программируемых параметров технологического процесса осуществляются с помощью клавиш, отображаемых на мониторе.

2.6. Система управления, построенная на базе сетевых контроллеров DL-06.

Система управления «КОНТУР-2» обеспечивает высокоточное перемещение исполнительного органа установки в заданную позицию. По каждой оси программируется перемещение и скорость, а траектория может быть произвольной.



Структурная схема системы управления «Контур-2»

Архитектура СУ «КОНТУР-2» имеет многоуровневую структуру: персональный компьютер, ведущий контроллер Direct Logic DL-06 и набор ведомых контроллеров Direct Logic DL-06 по количеству управляемых осей (A, B, C, D и т.д.). В пользу такого решения говорят встроенные программные и аппаратные возможности контроллеров по управлению шаговым приводом, относительная простота реализации транслятора управляющей программы на персональном компьютере средствами языка Visual Basic 6.

Важнейшей задачей при построении СУ является организация связей и распределение задач между сетевыми ПЛК по территориальному и функциональному признакам. Обмен между контроллерами Direct logic осуществляется через нижний сетевой порт. Контроллеры DL-06 с развитыми коммуникационными возможностями гарантирует надежную работу систем управления с распределенной структурой. Оба порта контроллера DL-06 поддерживают связь по протоколам: K-Sequence, DirectNet, ModBus.

СУ обеспечивает взаимодействие между:

- промышленным компьютером и ведущим ПЛК**;**
- ПЛК и элементами станочной электрической автоматики;
- ПЛК и шаговыми приводами перемещения.

Функции управления распределены по подсистемам, программным модулям. Архитектура СУ позволяет относительно легко адаптировать её к станочному оборудованию с любым числом степеней свободы. Сетевые средства промышленной коммуникации обеспечивают надёжное и гибкое управление.

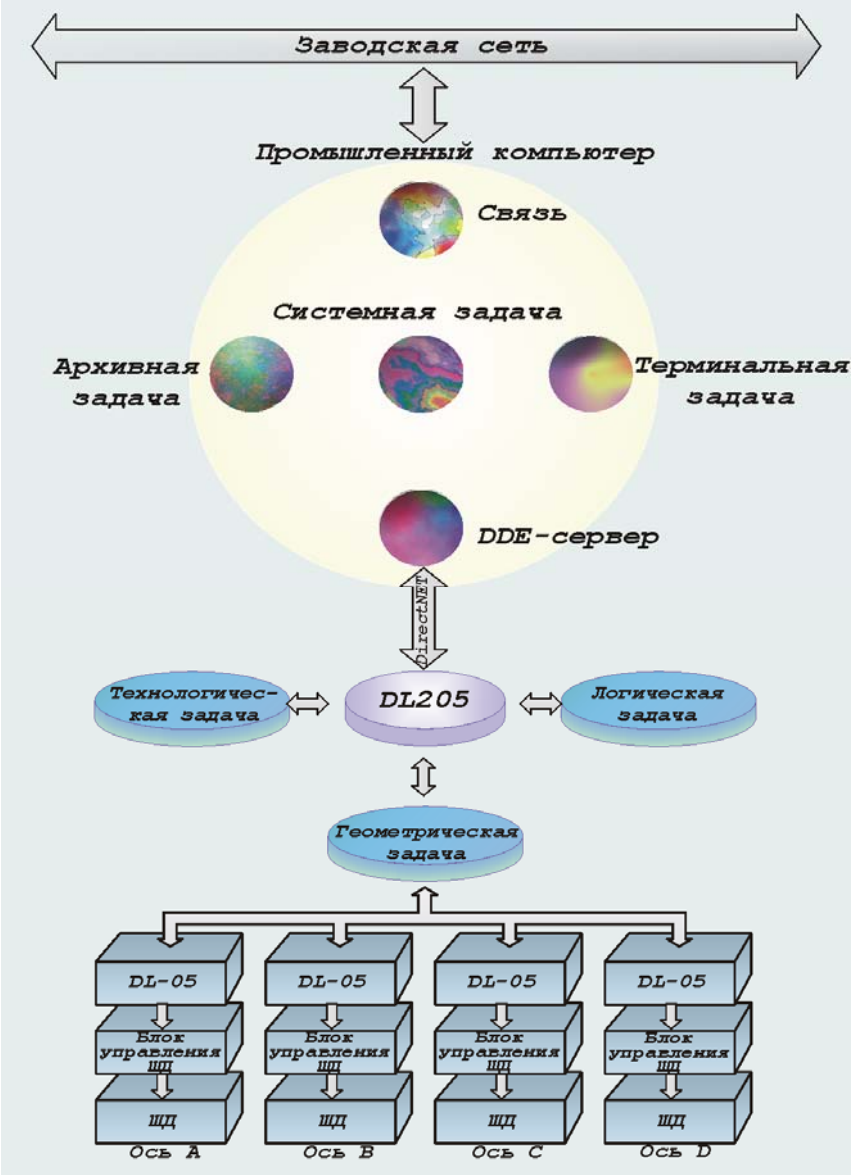
Система позиционирования выполнена на базе шагового модуля Mdrive, объединяющего в одном корпусе шаговый двигатель и силовой привод. Это позволило сократить количество кабелей, минимизировать помехи. Серия модулей MDrive обеспечивает диапазон моментов от 0,1 Нм до 10,34 Нм, работает в двух режимах: микроделения шага и скоростном (скорость вращения шагового двигателя пропорциональна напряжению на входе). Программно задаются настройки движения (ток удержания и движения, ускорение/торможение, начальная и максимальная скорость, режим микроделения шага до 256).

Оператор должен запрограммировать параметры трапециевидного профиля (профиль автоматического ускорения/замедления): конечное значение перемещения, начальная скорость, постоянная скорость, период ускорения и период замедления. После задания перечисленных параметров контроллер автоматически управляет скоростью ускорения/замедления и импульсным выходом.

Аналогичную архитектуру имеет СУ «Контур-1», построенная на базе контроллеров DL-205 и DL-05.

К преимуществам системы "Контур-1" относятся:

- простота наращивания дополнительных осей;
- модульный принцип построения СУ;
- развитые программные средства, позволяющие минимизировать усилия оператора-технолога при подготовке управляющей программы;
- автономность и высокая надёжность элемента управления по каждой координате;
- индикация расчётных и текущих координат исполнительного механизма установки, режимов работы, аварийно-предупредительных сообщений;
- удобство эксплуатации;
- низкая стоимость.



Структурная схема системы управления «Контур-1»

Верхний уровень СУ реализован на базе персонального компьютера. Программное обеспечение верхнего уровня, разработанное средствами языка Visual Basic 5 в среде операционной системы Windows NT**,** обеспечивает реализацию следующих функций:

- цветной графический интерфейс с оператором, визуализацию значений технологических параметров и состояния механизмов;
- интеллектуальную поддержку функций управления;
- ввод, редактирование и просмотр управляющей программы, настроек регуляторов, архива;
- автоматическое формирование оперативных сообщений на основе анализа аварийных и внештатных ситуаций.

2.7. Система управления, построенная на базе процессора D2-250 и модуля Ethernet H2-ECOM.

Печь вакуумная «СНВ-80» предназначена для вакуумного отжига малогабаритных изделий из титановых сплавов и дисперсионного твердения деталей из медных сплавов.

Структура СУ «СНВ-80» – двухуровневая. Нижний уровень составляет ПЛК DL-205, осуществляющий первичную обработку данных и управление.

ПЛК по интерфейсу Ethernet связан с верхним уровнем (персональный компьютер), обеспечивающим визуализацию элементов технологической системы, индикацию численных значений параметров технологического процесса, просмотр, ввод и редактирования технологических управляющих программ, архивирование данных, ведение журнала событий, ошибок и аварийных ситуаций.

Модуль Ethernet H2-ECOM обеспечивает высокоскоростную передачу данных по протоколу Ethernet между контроллером DL-205 и верхним уровнем (персональный компьютер). Соединение контроллеров с компьютерами осуществляется посредством стандартных кабелей, концентраторов HUB, повторителей.

К преимуществам построения систем управления с коммуникационным модулем H2-ECOM можно отнести:

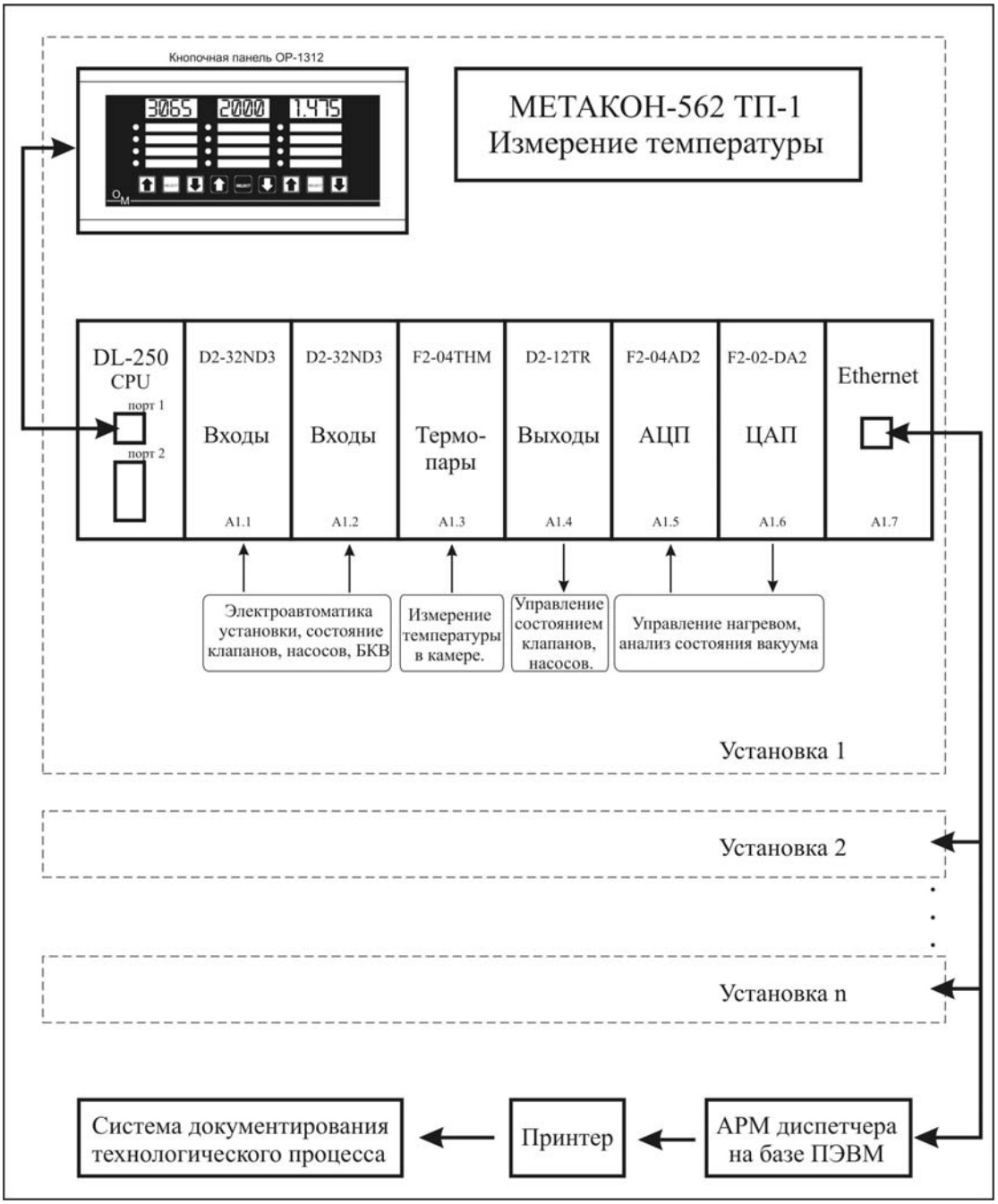
- соединение контроллеров в высокоскоростную сеть с равноправными узлами;
- использование DDE/OPC сервера обеспечивает эффективный обмен с персональным компьютером, программными продуктами для Windows;
- практически неограниченное количество узлов в сети;
- несложная установка и настройка.

Для оперативной индикации основных параметров технологического процесса термической обработки применяется малогабаритный пульт управления ОР-1312.

На пульте управления в цифровом виде индицируются основные параметры процесса (температура и давление в камере, текущее время и время выдержки изделия в вакууме), световая и звуковая сигнализации об отклонениях технологического процесса и неисправности установки (нарушения в системе охлаждения, обрыв термопары, нарушение допустимого диапазона температур и временных параметров).

Для реализации технологии вакуумного отжига разработаны следующие алгоритмы: регулирование температуры по ПИ-закону; адаптивное управление нарастанием температуры с организацией обратной связи по давлению, обеспечивающей остановку нагрева до набора рабочего вакуума. Алгоритмы минимизируют нестабильность процесса, сокращают энергопотребление, увеличивают производительность.

Исходную информацию для алгоритмов обеспечивает модуль «F2-04THM», который конвертирует входной аналоговый сигнал от термопары типа ТХА в градусы Цельсия. Контроллер DL-205 получает информацию о текущей температуре с точностью 0,1 градуса Цельсия и обеспечивает её первичную обработку для формирования массива данных.



Структурная схема системы управления «СНВ-80»

Управляющий сигнал с помощью цифроаналогового модуля «F2-02DA-2» преобразовывается в пропорциональный аналоговый сигнал 0...10V для блока управления тиристорами «БУСТ», который обеспечивает фазовое регулирование модуля IRKT500-08 (IR). Напряжение на нагревателях снижено до 30V при максимальной скорости подъёма температуры.

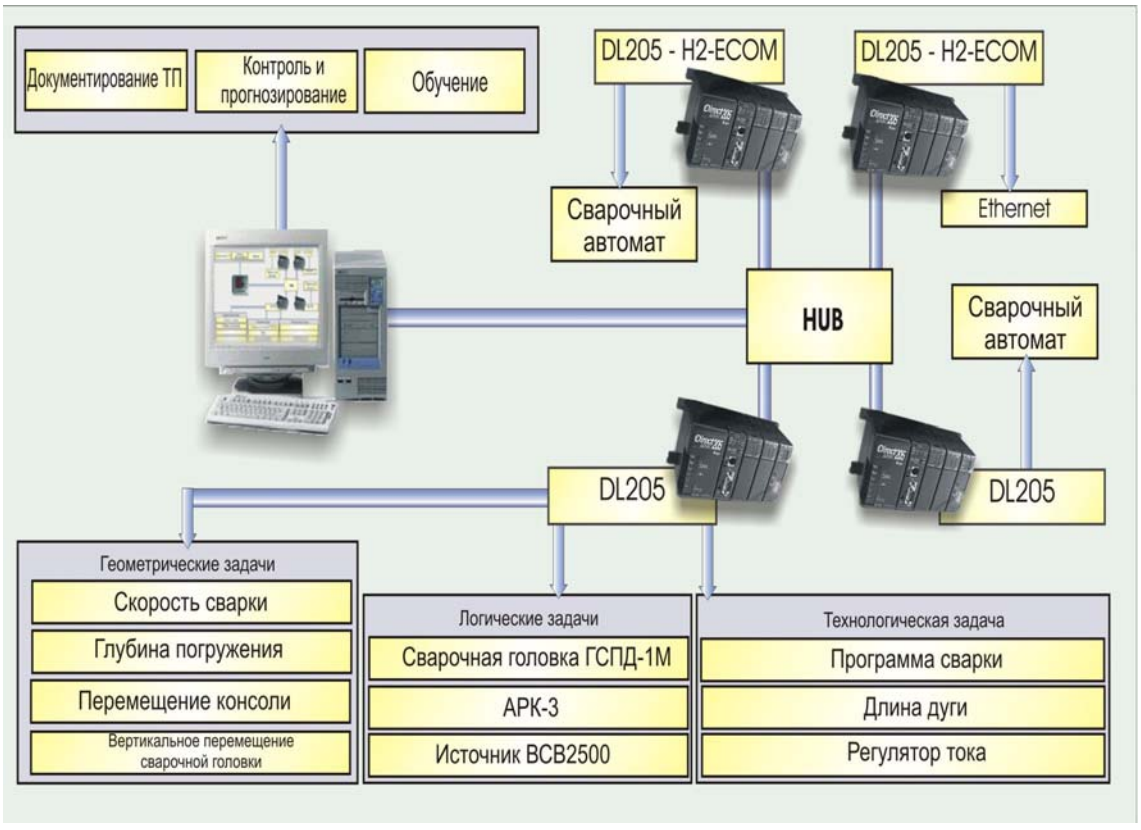
Управление элементами электрической автоматики реализовано на базе модуля D2-12TR – (12 каналов дискретных релейных выходов) и двух модулей D2-32 ND3 (64 канала дискретных входов).

Модуль диагностики и прогнозирования процесса отжига обеспечивает выполнение блокировок, переход установки в безопасное состояние при возникновении аварийных ситуаций: отсутствие охлаждения, отклонение температуры от допустимых ограничений, неправильные действия оператора.

После окончания каждого технологического процесса формируется отчёт, который записывается в отдельный файл и хранится на жёстком диске компьютера. Файл отчёта может быть вызван на монитор для визуального анализа или на печать.

АРМ диспетчера, реализованное на базе персонального компьютера, позволяет собирать и обрабатывать информацию о процессе вакуумной термической обработки с нескольких установок аналогичного типа.

Комплекс «СКПД-2500», предназначенный для сварки продольных и кольцевых швов погружённой дугой вольфрамовым электродом в среде инертных газов изделий из конструкционных, жаропрочных, нержавеющих сталей и титановых сплавов, включает радиально-консольный автомат «АРК - 3АВ», сварочную головку «ГСПД-1М», манипулятор, источник тока «ВСВ-2500».



Структурная схема системы управления «СКПД-2500»

Управление работой комплекса на нижнем уровне осуществляется от контроллера DL-205.

СУ выполняет следующие функции:

- оперативное отображение на пульте основных параметров процесса сварки (скорость сварки, ток сварки и скорость и перемещение электрода); просмотр, редактирование управляющих программ сварки – терминальная задача.

- управление скоростью перемещения электрода в вертикальном направлении (глубина позиционирования электрода), скоростью сварки, скоростью перемещения консоли, скоростью вертикального перемещения сварочной головки (геометрическая задача);
- управление электрической автоматикой автомата «АРК - 3АВ», сварочной головки «ГСПД-1М», манипулятора и источника тока «ВСВ-2500» (логическая задача;
- слежения за длиной дуги в режиме регулятор тока (технологическая задача);
- передача информации о параметрах процесса сварки на ПЭВМ (задача связи).

АРМ диспетчера участка сварочных автоматов реализован на базе персонального компьютера. Соединение контроллеров DL-205 с компьютером осуществляется с помощью модуля Ethernet H2-ECOM, концентратора HUB и стандартных кабелей. Компьютер обеспечивает документирование процесса сварки, техническую диагностику.

2.8. Система управления, построенная на базе процессора D4-450.

Установка «УВНК-8ПМ» предназначена для получения изделий из жаропрочных сплавов с направленной и монокристаллической структурой, с заданной кристаллографической ориентацией.

Архитектура системы управления установки «УВНК-8ПМ» имеет структуру ПЛК - ПК. Выбор для нижнего уровня системы управления ПЛК Direct Logic семейства DL-405 осуществлён по следующим соображениям:

1. Мощные вычислительные возможности процессора D4-450: быстродействие 0,96 мкс на одну операцию, 30.8 Кслов общая флеш-память, 15 Кслов память переменных, 256 счётчиков, 2048 таймеров, 256 прерываний, часы-календарь, 16 встроенных ПИД-регуляторов с автонастройкой, команды для работы с числами в формате с плавающей запятой, развитые сетевые средства для работы с протоколами K-sequence, DirectNET^{TM}, MODBUS RTU, три порта RS232/RS422/RS485, общее количество точек ввода-вывода – 3584.
2. Широкий выбор модулей для ввода-вывода аналоговой и дискретной информации.
3. Высокая надёжность при работе в промышленных условиях: повышенная вибрация, перепады температуры, броски питания, электромагнитные помехи.

Помимо центрального процессора D4-450, контроллер нижнего уровня, реализованный на базе аппаратно-программных средств семейства DL-405, включает следующие модули:
- сопроцессор D4-INT, обеспечивающий обработку прерываний на 8 входах в диапазоне 0.08 …0.59 мс;
- дискретный ввод постоянного тока D4-32 ND3 - 1 (32 канала);
- дискретный ввод постоянного тока D4-64 ND2 (64 канала);
- дискретный вывод постоянного тока D4-64 TD1 (64 канала);
- ввод аналоговой информации F4-08AD (8 каналов, 12 бит разрешение, диапазоны: 4-20 mA, 1-5 V, 0-20 mA, 0-5 V, 0-10 V, ± 5 V, ±10 V);
- вывод аналоговой информации F4-04DA-2 (4 канала, 12 бит разрешение, диапазоны: 0-5 V, 0-10 V, ±5 V, ±10 V);
- вывод аналоговой информации D4-02AD (2 канала, 12 бит разрешение, диапазоны: 4-20 mA, 1-5 V, 0-10 V);

ПЛК обеспечивает сбор и обработку информации от датчиков температуры и положения. Каждый контролируемый параметр на стадии его определения подвергается стандартной математической обработке, которая

включает: масштабирование измеренных сигналов, контроль достоверности путём логического анализа значений взаимосвязанных параметров, выбраковку ложных измерений, контроль нарушения заданного диапазона изменения.

Контроллер обеспечивает регулирование по ПИД-закону контуров нагрева (верхняя и нижняя зоны печи подогрева форм) и контуров перемещения (наклон тигля для слива металла в формы, вертикальное перемещение блока форм в жидкометаллическом кристаллизаторе) по программной траектории. Технические средства нижнего уровня обеспечивают автономную работу СУ без верхнего уровня.

Для оперативного управления технологическим процессом применяется малогабаритный пульт управления DV-1000 с символьным дисплеем и набором клавиш.

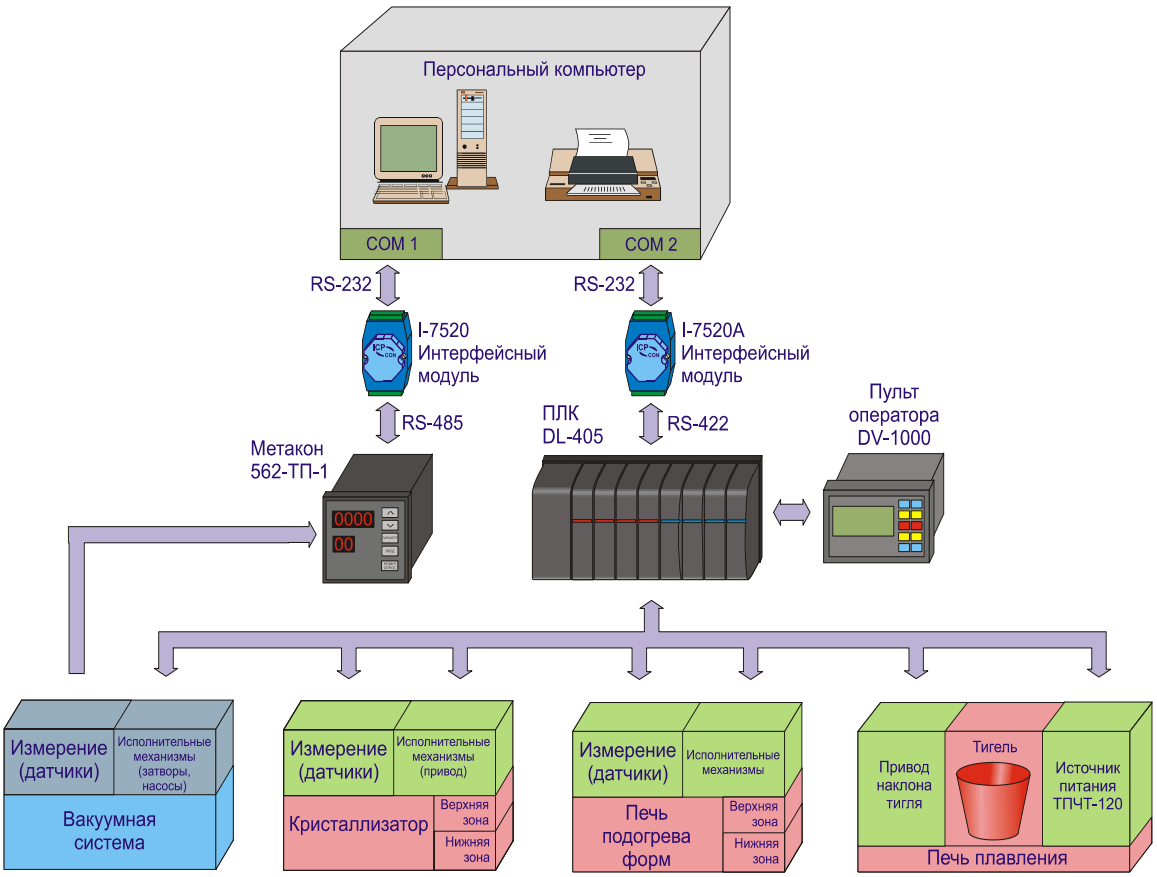
Верхний уровень СУ, реализованный на базе персонального компьютера, собирает и обрабатывает информацию о технологическом процессе. При аварийной ситуации или отсутствии связи между компьютером и ПЛК можно перейти на ручное управление.

Программное обеспечение верхнего уровня, разработанное средствами языка Visual Basic в среде операционной системы Windows (95, 98, NT**),** реализует цветной графический интерфейс с оператором и интеллектуальную поддержку функций управления. DDE-сервер осуществляет связь с Windows-приложениями: Excel, Visual Basic, In Touch.

Программное обеспечение верхнего уровня СУ обеспечивает:

- отображение состояния механизмов и датчиков до четырёх установок на мониторе компьютера;
- индикацию текущих значений параметров установки в реальном масштабе времени;
- регистрацию и хранение параметров технологического процесса с привязкой к конкретной лопатке и реальному времени.
- конвертирование сформированной базы данных в формат программы Excel;
- вывод на принтер паспорта на проведенную плавку;
- светозвуковую сигнализацию при отклонениях технологического процесса: нарушение водоохлаждения; обрыв термопар; отклонение параметров вакуумной системы; отклонение скорости перемещения формы в жидкометаллическую ванну.

Отличительными чертами СУ являются гибкость, многофункциональность, программная модернизация алгоритма работы и технологии. Её преимущество проявляется в более высоких точностных и динамических показателях управления, удобном интерфейсе оператора.



Структура системы управления "УВНК-8ПМ"

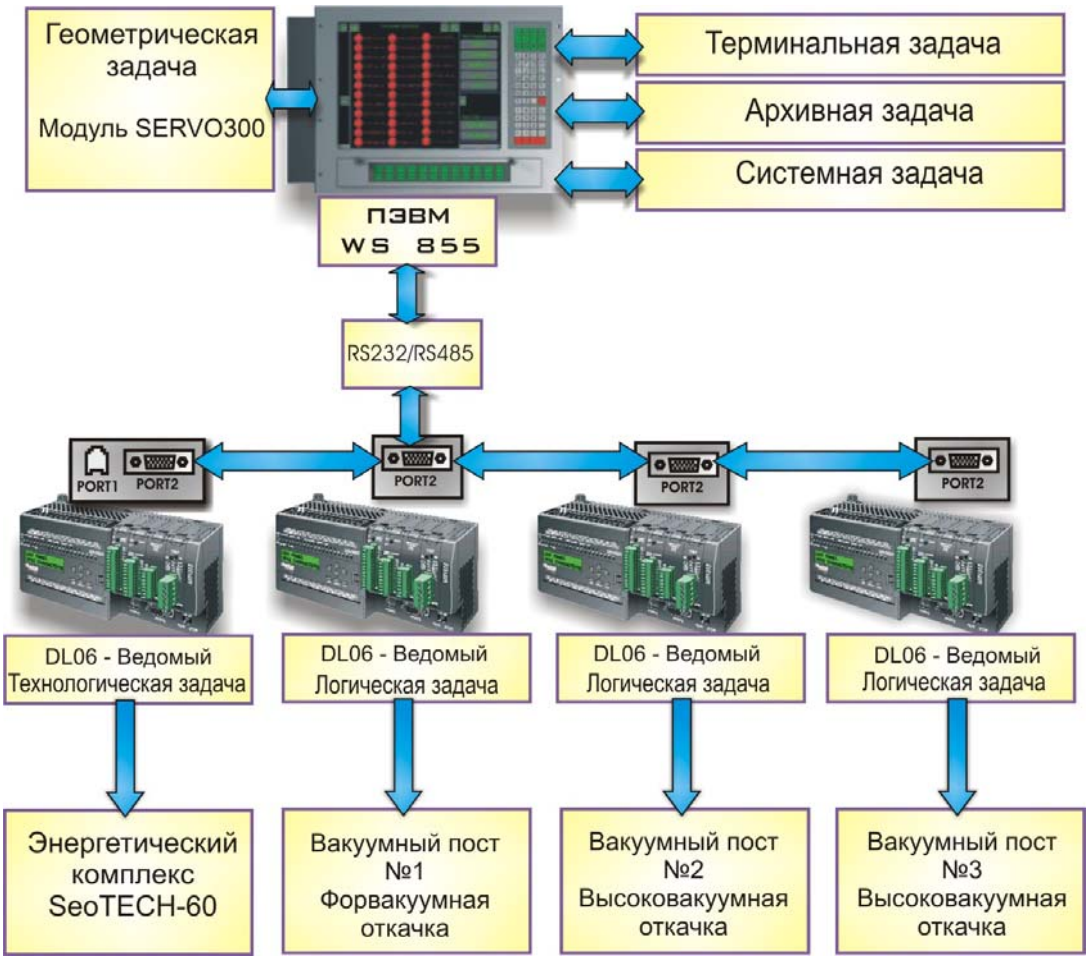
Система управления в процессе работы создает архивный файл, файл событий и файл ошибок. Данные хранятся на жестком диске в формате Excel.

Важным элементом компенсации нестабильности параметров технологического процесса, действий при внештатной ситуации является использование алгоритмов нечёткой логики. Введение в управление нечёткого регулятора позволяет учитывать информацию качественного характера, имитировать нестандартные действия опытного оператора.

2.9. Система управления, построенная на базе промышленного компьютера и сети контроллеров DL-06.

СУ установки «ЭЛУ-9М» построена на базе промышленного компьютера WS-855, работающего под управлением операционной системы Windows 2000, модуля управления движением SERVO 300 фирмы ICP DAS и сети ПЛК, включающей четыре контроллера DL-06.

СУ позволяет реализовать, кроме электронно-лучевой сварки, такие технологические процессы, как пайка, термическое упрочнение, локальный отжиг, наплавка, резка, плавка, напыление, модифицирование поверхностей.



Структурная схема системы управления ЭЛУ-9М»

Промышленный компьютер, оснащённый модулем «SERVO 300» для управление тремя раздельными приводами подач, выполняет не только терминальную, но и геометрическую задачу управления.

Установка «ЭЛУ 9М» имеет три степени свободы (рабочее перемещение изделий вдоль по координате X, рабочее вращение планшайбы, вертикальное перемещение накамерной пушки по координате Z).

Возможности СУ во многом определяются программным обеспечением**,** которое разрабатывается с учетом конкретных требований заказчика по технологическому процессу.

Для уменьшения количества соединений между исполнительными элементами и устройством управления, а также для расширения возможностей системы по обслуживанию далеко стоящих от центрального устройства исполнительных органов установки применяются автономные вакуумные станции, управление которыми построено на базе контроллеров DL-06. При этом отдельные части системы обмениваются между собой данными по последовательному каналу, подчиняясь командам компьютера.

Таким образом, СУ обеспечивает выполнение следующих функций:

- автоматическое управление работой вакуумной системы с возможностью работы в ручном и наладочном режимах;

- слежение и управление величинами и скоростями рабочих перемещений изделий, вращения планшайбы, перемещения электронно-лучевых пушек;
- контроль наличия подачи воды на агрегаты установки;
- автоматический выход на режим сварки и автоматическое окончание сварочного процесса по заранее выбранному закону;
- автоматическое и полуавтоматическое наведение луча на стык свариваемых изделий;
- программное управление энергетическими характеристиками (током луча, током фокусировки, развертками луча);
- автоматический контроль основных параметров процесса сварки (ускоряющее напряжение, ток луча, ток фокусировки, скорость сварки);
- документирование основных параметров сварки (распечатка паспорта сварки детали с указанием даты, номера детали, режима сварки, отклонений от нормативного режима сварки с привязкой к траектории стыка).

3. Анализ систем управления.

Каждый подход к организации систем управления имеет как свои достоинства, так и недостатки. Определение оптимального состава программно-аппратных средств СУ для решения технологических приложений представляет достаточно сложную многокритериальная задачу. Это касается не только выбора модулей ПЛК, но и средств ввода-вывода информации. Альтернатива здесь такова: панель контроллера или промышленный компьютер, так как применение офисного компьютера в производственных условиях не всегда оправдано. В этом случае, принцип достаточности оказывается приоритетным: для малых и средних Т-систем – это цифровые и сенсорные панели ПЛК, для больших систем – промышленный компьютер.

Поэтому цифровая панель OP-1312 для решения простых интерфейсных задач оказалась вполне достаточной (глава 2.1), а цветная сенсорная панель «EZ-S6C-K» с развитым многооконным интерфейсом (глава 2.4) практически не уступает возможностям персонального компьютера. В этом случае, функции математического моделирования технологического процесса, документирования параметров технологического процесса, архивирования, формирования файлов событий организованы на базе АРМ диспетчера участка или цеха. Оптимальным решением для связи между ПЛК и цеховым компьютером является сетевая технология Ethernet (глава 2.7).

Наиболее сложной в плане программирования оказались системы управления установок «УВНК-9», «ПВ-850», «УВН-1500М» (см. главу 2.2). Обмен информацией между контроллерами ведется с помощью сетевых команд по протоколу DirectNet. Несмотря на относительную автономность каждого контроллера по решаемым задачам управления, при программировании не удалось минимизировать объём информации для обмена между контроллерами. Необходимость выделения большого количества памяти для буферных массивов данных, их постоянный анализ, работа независимо от циклов сканирования сетевых команд потребовали значительных усилий в плане программирования и отладки. Но, с другой стороны, достаточно сложная система оказывается вполне жизнеспособной в случае нарушения связей или при отказе одного из контроллеров - производственный процесс поддается доведению до конца в ручном режиме с полной диагностикой неполадок, что

является неоспоримым достоинством системы управления, состоящей из нескольких контроллеров, которые решают независимые задачи. Программное обеспечение каждого из контроллеров при этом оказывается относительно несложным, за исключением задачи обмена.

Структура с использованием модуля удаленного ввода-вывода D2-RSSS также доказала свою жизнеспособность (см. Главу 2.3). Удобство ее состоит в том, что контроллер имеет большое количество дискретных входов и выходов, при этом дополнительные модули располагаются на другом каркасе, а адресное пространство входов и выходов является продолжением того, которое сформировалось на основном каркасе. Скорость доступа к дискретным точкам удаленного ввода-вывода практически не изменяется. При программировании такого рода системы управления требуется несколько несложных команд для настройки модуля удаленного ввода-вывода, хорошие примеры имеются в описании. Недостаток данной системы - возможное нарушение связи с удаленным вводом-выводом, что приведет к неадекватному отображению входов и невозможности управления выходами. Кроме того, если каркасы удаленного ввода-вывода включают в себя модули ЦАП/АЦП, то применение чрезвычайно удобного метода указателя при записи/чтении данных в этом случае становится невозможным. И, наконец, в случае сложной многофункциональной установки программа может оказаться достаточно громоздкой.

Система на базе процессора D2-260 (Глава 2.4) отличается лишь наличием модуля D2-EM Termination, благодаря которому отпадает необходимость включать в программу специальные команды для настройки удаленного ввода-вывода: это уже реализовано аппаратно, что, несомненно, является шагом вперед по сравнению с использованием модуля D2-RSSS. К достоинствам этой структуры можно отнести большое расстояние между каркасами (до 32 метров), простоту программирования.

Что касается связи с компьютером верхнего уровня, то здесь вариант пока только один - использование программы-посредника под названием DDE Server, которая считывает с контроллера значения ячеек памяти и отдельных битов и делает их доступными для пользовательской программы. Для корректной работы с контроллером DDE-Server должен быть загружен перед запуском программы верхнего уровня. Даже на достаточно мощных компьютерах эта программа работает относительно медленно, особенно если требуется отобразить содержимое большого количества ячеек контроллера. И если для управления или визуального наблюдения за состоянием установки такая производительность DDE-сервера является удовлетворительной, то она может оказаться неприемлемой, если предъявляются особые требования к скорости документирования: производить запись данных процесса с частотой более 1 раза в секунду не имеет смысла, потому что DDE-сервер просто не будет успевать их обновить. Практика показала, что при подключении к верхнему уровню системы управления нескольких ПЛК возникают большие временные задержки с обменом информацией при возникновении внештатной ситуации на одном из контроллеров.

В случае, если программное обеспечение верхнего уровня должно отображать состояние нескольких установок, целесообразно применение коммуникационного модуля H2-ECOM (см. Главу 2.7), благодаря которому контроллеры соединяются в высокоскоростную сеть, и обмен данными ведется

по протоколу Ethernet. Количество узлов в такой сети практически неограничено.

Промышленные контроллеры класса ПЛК, на протяжении десятков лет являясь непременным элементом технологических систем, оптимальны по надёжности. Однако для быстродействующих систем, когда время цикла вычислений не должно превышать несколько десятков микросекунд, применение контроллеров Direct Logic невозможно, поскольку даже при отсутствии пользовательских команд время цикла сканирования составляет не менее 1 миллисекунды. Для такого рода систем управления, критичных к быстродействию, существует отличное решение - недорогие и производительные, до 8 миллион операций в секунду, микроконтроллеры семейства AVR фирмы Atmel. Критичные по времени задачи управления целесообразно перенести на однокристальные контроллеры. Обмен информацией между микроЭВМ и ПЛК, в этом случае, достаточно просто можно организовать по протоколу ASCII.

СУ на базе контроллера DL-205 позволяет эффективно решить задачу управления следящим приводом на базе асинхронного двигателя или двигателя постоянного тока. Для аппаратной реализации алгоритмов: «Выход в точку», «Стабилизация скорости», «Нулирование» выбраны модули «H2-CTRIO» и «F2-08DA-2, что обеспечило надёжную работу станочных узлов вакуумных установок (глава 2.4). Программно-аппаратные средства контроллера DL-06 достаточно просто решают задачу организации многокоординатной системы позиционирования на базе шагового привода (глава 2.6).

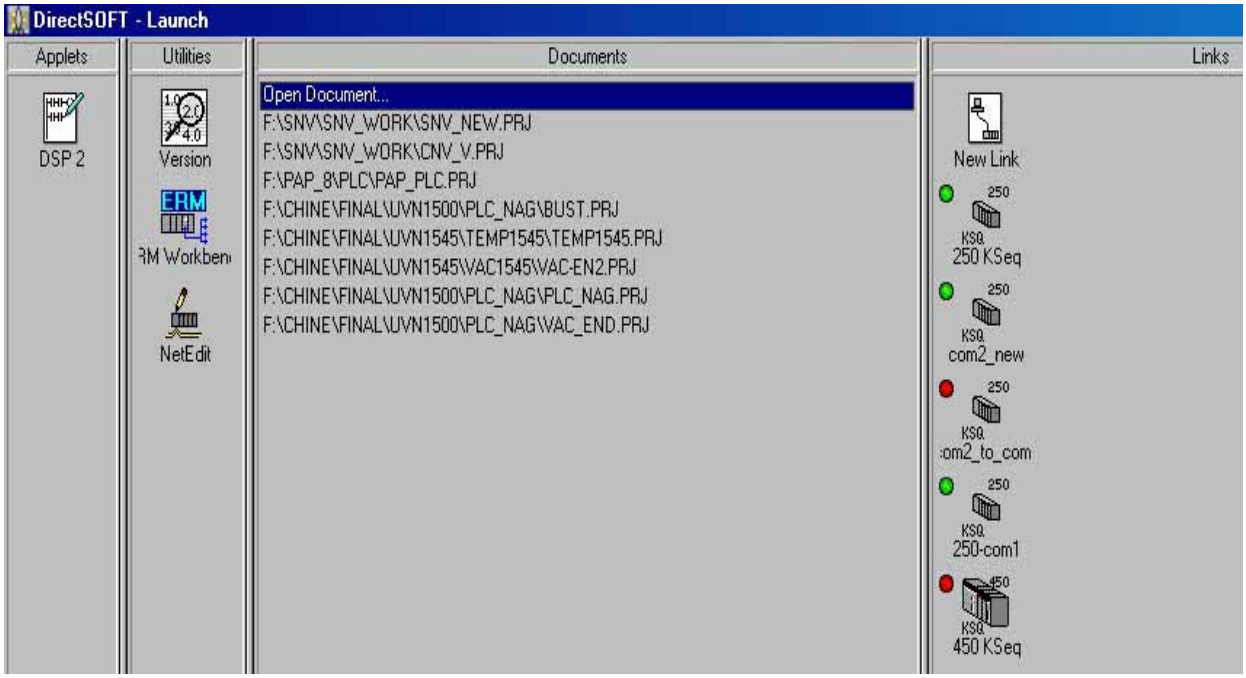
Выполнение интерполяционных алгоритмов на ПЛК, имеющих значительное время вычислительного цикла, нерационально. Приемлемым решением на сегодняшний день является решение траекторной задачи на базе модуля «SERVO 300» в составе промышленного компьютера (глава 2.9).

Дальнейшее развитие СУ во многом определяется, с одной стороны, наличием математической модели технологического процесса, алгоритмов проектирования технологии, адаптивного управления, с другой стороны, аппаратными и программными возможностями самой СУ. Практическая реализация наукоемких математических моделей представляется возможной только на СУ, содержащих в своей архитектуре компьютер. Вычислительный потенциал таких СУ обеспечивает возможность интеграции функций проектирования технологии в машинном масштабе времени и управление процессом обработки детали в реальном масштабе времени. Объединение функций проектирования технологии и управления осуществляется на основе принципа структурно-функциональной интеграции, который заключается в образовании единой иерархической структуры управления технологическим оборудованием и позволяет на научно-методической основе обеспечить эффективную структуру производственного цикла за счет сокращения затрат на передачу информации, единой координации целей. Принцип интеграции согласуется с другими основными принципами эффективной организации сложных целенаправленных систем: обратной связи, иерархичности, распараллеливания, координации.

4. Обзор версий Direct Soft.

Программирование на RLL Plus осуществляется в среде DirectSoft. Эта оболочка проста в использовании, позволяет просматривать ячейки и биты процессора во время работы программы, оперативно вносить изменения, настраивать процессор (области сохранения, коммуникационный порт, ПИД-регулятор, сторожевой таймер, дату и время и т.д.), управлять его работой и получать диагностику его состояния. DirectSOFT работает под управлением операционной системы Windows 95 и выше.

Первоначально для программирования использовалась 16-битная версия Direct Soft. В принципе, этот пакет и сейчас будет вполне работоспособен. Ограничения его в том, что он не поддерживает новые модели процессоров – DL-06 и D2-260 и, соответственно, их систему команд, а также не может "распознать" новые модули, которые постоянно разрабатываются и появляются на рынке. Одним из существенных недостатков среды этой версии Direct Soft являлось отсутствие подсказок при подведении указателя мыши к какой-нибудь кнопке на панели инструментов - обычно солидное программное обеспечение под Windows такие мелочи обязательно учитывает.



Оболочка 16-битной версии Direct Soft

Оболочка 32-битной версии Direct Soft

Следующее поколение пакета Direct Soft уже было 32-битным. Отличие от 16-битной версии было в интерфейсе оболочки, и до появления последнего на сегодняшний день пакета Direct Soft 4 было несколько промежуточных версий, которые включали в себя опознавание очередного нового модуля от фирмы-разработчика и ставились поверх своей предыдущей версии. В этих версиях по части удобства работы со средой программирования сделан шаг вперед: появились подписи как на самих кнопках, так и всплывающие подсказки. Широкое распространение "мышек" с функцией Net Scroll, к сожалению, не нашло понимания у разработчиков Direct Soft, и эта функция не поддерживается, хотя о том, насколько она удобна, даже не стоит говорить. Пакет Direct Soft 4 поддерживает все семейства контроллеров DirectLogic и все существующие на данный момент модули. Можно предположить, что с появлением новых модулей, появятся свои программные инструменты, которые легко встраиваются в оболочку подобно, например, приложению ERMWorkbench.

5. Состав программного обеспечения технологической установки.

Перед тем как приступить к разработке программного обеспечения системы управления технологической установки, необходимо определить круг его задач. Разумеется, для каждой установки существует своя специфика в соответствии с ее назначением, однако практически у каждой системы управления, разработанной на базе контроллера DirectLogic средствами языка RLL Plus, имеются общие элементы, которые присутствуют в том или ином виде в каждой программе. О них и поговорим в этой главе.

Прежде всего – качественный и надежный проект практически невозможно реализовать без использования стадий, этого удобного изобретения фирмы-разработчика контроллеров DirectLogic. Благодаря стадиям в большинстве случаев удается избавиться от решения задач, связанных с устранением дребезга, с параллельными процессами, с дополнительными блокировками и прочими условиями, которые приводят к нагромождению контактов. Стадия – совершенно уникальный подход, требующий особого понимания, зато предоставляющий программисту, помимо других средств языка

RLL Plus, исключительную гибкость, а также возможность создавать прекрасно структурированные программы. Аналога стадий в языках высокого уровня и ассемблерах нет. Несмотря на то, что язык RLL Plus является языком низкого уровня, опыт программирования на нем позволяет говорить о выработанном стиле, который мы также обсудим в этой главе. Но сначала – коротко и простыми словами о том, что такое стадия.

5.1. Использование стадий.

Стадия – это фрагмент программы, который начинается с заголовка стадии и продолжается до заголовка следующей стадии. Этому фрагменту ставится в соответствие бит стадии, и в зависимости от состояния этого бита данный фрагмент сканируется и выполняется либо игнорируется. Манипулируя этими битами, можно переключаться из одной стадии в другую, а также запускать несколько стадий, что позволяет говорить о параллельных процессах. На самом деле, конечно же, процессор выполняет только одну стадию, но за один цикл сканирования, достаточно короткий по времени (обычно несколько десятков миллисекунд) он выполняет последовательно несколько стадий, биты которых установлены, что и дает эффект одновременности. Переход от одной стадии к другой обычно производится командой JMP S<номер стадии>. Начинающим программистам обычно невдомек, что эта команда не вызывает немедленного перехода. На самом деле эта команда объединяет в себе две команды: RST S<номер стадии A> и SET S<номер стадии N>. Работает это так: пусть в стадии S1 есть команда JMP S10. Даже если она стоит сразу за заголовком стадии (команда SG S1), все ветви стадии S1 будут сканированы и выполнены, в том числе и команда JMP. А сделает она следующее: установит бит S10 и подготовит бит S1 к сбросу в начале следующего цикла сканирования. Но после выполнения стадии S1 процессор проанализирует биты S2, S3,…S7 и выполнит логику этих стадий, если эти биты были установлены, и только потом выполнит логику стадии S10. Если в стадии S10 есть команда JMP S2, т.е. переход на стадию, которая сканируется раньше, она будет выполнена только в следующем цикле сканирования. Более подробно о стадиях и их применении (вместе с примерами) можно прочитать в Документации по контроллеру, глава «Стадийное программирование RLL Plus».

Таким образом, при программировании системы управления технологической установки мы можем создавать замкнутые независимые циклы, состоящие из одной или нескольких стадий, которые будут решать свои локальные задачи в составе системного программного обеспечения, т.е. в полной мере реализовать один из основных принципов рационального программирования – «разделяй и властвуй». К таким задачам, которые присутствуют в большинстве систем управления, относятся:

- измерения (получение исходных данных, пересчет);
- ручное управление;
- автоматический режим;
- диагностика состояния установки;
- обработка ошибок.

В некоторых системах также могут решаться отдельно задачи индикации, сетевого обмена и т.п. Каждая задача представляет собой стадию или группу стадий, которые активируются одновременно. Обычно это делается в начальной стадии (команда ISG). Также в начальной стадии целесообразно произвести настройки всех модулей, входящих в состав системы управления, и проинициализировать переменные.

Ячейки V-памяти, которые при начале работы должны равны нулю, инициализировать не требуется – если они не входят в область сохранения, память обнуляется процессором при запуске. Инициализировать мы будем массивы; вернее, указатели на них. В языке RLL Plus к V-памяти можно обращаться косвенно, посредством

указателей, и это дает еще большую гибкость при составлении программ для контроллеров DirectLogic. Использование массивов в программировании на RLL Plus практически ничем не отличается от их использования в других языках, поэтому в таком контексте их рассматривать не будем. Предположим, что один из локальных циклов, состоящий из нескольких стадий, готовит данные для обработки, а другой цикл их обрабатывает. Каким образом передать данные в другой цикл? – Организуем для этого очередь, в которую один цикл записывает данные по мере их готовности, а другой цикл просматривает эту очередь и, если находит на текущем шаге элемент, готовый для обработки, производит ее. Один из примеров такого рода очереди – очередь ошибок. При возникновении ошибки она записывается в очередь. Стадия обработки ошибок, обнаружив в очереди код ошибки, каким-то образом сигнализирует оператору (высвечивание кода на панели индикации, подача звукового сигнала) о сбое, но, поскольку оператору требуется некоторое время, чтобы отреагировать на сигнал, в работе системы могут произойти другие ошибки, и благодаря очереди оператор будет информирован обо всех произошедших ошибках. Еще один пример использования очереди – создание файла событий – будет подробно рассмотрен в одной из последующих глав.

5.2. Рекомендации по стилю.

Помимо деления программы на функциональные группы стадий, что делает ее более структурированной, дадим еще несколько рекомендаций по стилю, хотя, разумеется, следование им – дело вкуса программиста и не предполагает какой-либо обязательности. Итак, представляется нерациональным использование «самоубийства» стадии, т.е. когда она сама себя деактивирует: в этом случае не видно, как дальше развивается логика работы программы. Хотя, надо признаться, иногда без этого не обойтись.

Бывает, что не обойтись и без пресловутого оператора GOTO, который присутствует в RLL Plus, как и во всяком солидном языке программирования. Этот оператор прерывает сканирование ветвей программы до указанной метки. Правда, в отличие от других языков, использовать его в программе можно лишь 128 раз. Но при рациональном делении программы на стадии необходимость в его использовании возникает крайне редко.

То же самое можно сказать в отношении процедур (команда GTS): хотя вызывать их можно также 128 раз, благодаря использованию стадий редко когда приходится прибегать к процедурам. Процедура окажется полезной, если, например, нужно организовать вложенный цикл. Язык RLL Plus не допускает вложенных циклов FOR, однако обойти это ограничение можно, используя в цикле обращение к процедуре, в которой тоже есть цикл FOR. Использование этой команды приводит к увеличению времени сканирования, что, в свою очередь, может вызвать срабатывание сторожевого таймера, если количество циклов велико. Но если производить все вычисления в одном цикле не обязательно, лучше делать один проход за один цикл сканирования, модифицируя при этом счетчик, и отказаться от использования команды FOR/NEXT.

При программировании стадии важно следить за логикой работы программы, т.е. за переходами по команде JMP. Лучше, если по выполнении условия или их совокупности будет использована одна JMP. Если при этом нужно активировать еще какие-то стадии, сделаем это командой SET и постараемся не забыть, что мы включили параллельные процессы. В принципе, разницы никакой нет, однако программа будет более удобочитаемой, если переход к следующему этапу вычислений будет обозначаться командой JMP, а разветвление процесса – командой SET. Нежелателен также переход в одном цикле сканирования на одну стадию с нескольких разных, если, конечно, мы не

используем сливающиеся стадии – это тоже нарушает структурированность и удобочитаемость. И, наконец, хотя деактивировать стадию командой RST никто не запрещает, в большой программе, где велик риск запутаться, такой прием может привести к ее непредсказуемому поведению, поэтому деактивацию лучше производить командой JMP, а для удобочитаемости размещать ее в конце стадии.

Блоки стадий рационально использовать, если нужно организовать слежение за каким-либо процессом. Следящая стадия работает постоянно, проверяя условия, при которых процесс можно вести, и если условия выполнены, устанавливает бит, соответствующий блоку стадий процесса. При нарушении условий бит сбрасывается, а управление передается завершающей стадии, которая деактивирует следящую.

И заключительная рекомендация – не создавать сложные цепочки условий, в которых также легко запутаться; лучше при выполнении более-менее определенного условия перейти на новую стадию и продолжить дальнейшую проверку в ней. Дефицит стадий вряд ли грозит программистам: всего в программе может быть 1024 стадии (0-1777_8).

6. О программировании на RLL Plus.

Даже у бывалого программиста, писавшего на своем веку программы на разных языках высокого уровня и ассемблере, впервые уведенная программа на RLL Plus вызывает удивление, а знакомство с работой процессора DirectLogic и вовсе заставляет пересмотреть многие подходы к составлению программ, которым он пользовался на протяжении многих лет. Программы на RLL Plus не столько пишутся, сколько рисуются в виде схем Релейно-Лестничной Логики (Relay Ladder Logic -RLL), которые ближе конструктору-электрику, нежели программисту. Программирование можно отождествить с проектированием релейно-контактной схемы, да и терминология в описании команд языка RLL Plus используется соответствующая – реле, контакт, шина питания. Все это способно вызвать у программиста «до мозга костей» чувство неприятия и дискомфорта, поскольку столь конкретные предметы, какими являются реле и провода, не вписываются в его мышление, привыкшее оперировать достаточно абстрактными понятиями и категориями. Вообще разработчики RLL Plus и его графического представления предполагали, что этим языком легко смогут пользоваться конструкторы и наладчики, чтобы легко и быстро создавать системы управления, пользуясь привычными понятиями и на основе собственных схем.

Но мы считаем, что программирование, в каком бы виде оно ни существовало, своего рода искусство, и чтобы создавать качественные и красивые программы, нужно умение логически мыслить и интерес к разного рода логическим задачкам типа головоломок и кроссвордов. Составлять программы должен человек, обладающий такими качествами, а не обязанный это делать по долгу службы в качестве дополнения к основной работе, поэтому мы убеждены, что программирование на RLL Plus – это работа для профессионального программиста, который имеет представление о стиле, ценит экономичность, надежность кода и способен поломать голову в поисках красивого и мощного алгоритма. Наш опыт программирования на RLL Plus позволяет утверждать, что язык этот при умелом пользовании и знании специфики работы процессора достаточно гибок и позволяет реализовывать красивые решения достаточно сложных логических задач. И для успешного освоения этого языка совсем не обязательно проникаться «аппаратной» терминологией – любая программа на RLL Plus при внимательном рассмотрении удивительно напоминает хорошо знакомые всем профессионалам Паскаль и Си! Рассмотрим фрагмент программы на PLL Plus:

Этот фрагмент называется ветвью программы или ступенью. Каждая ступень состоит из условной (проверка состояния бита C20) и исполняемой (установка бита Y2 и записи числа 7 в ячейку V1500) частей. На Паскале это выглядело бы так:

```
IF  C20 = True  THEN
   BEGIN
     Y2 := 1   (* или Y2=True *);
     V1500 := 7;
   END;
```

Более сложный пример:

аналогичен коду на Паскале:

```
IF  C20  OR  C30  THEN
   BEGIN
     Y2 := 1;
     V1500 := 7;
   END;
ELSE  Y2=0;
```

Таким образом, вся программа на RLL Plus представляет собой набор операторов IF…THEN с условиями разной сложности. И никаких там контактов с обмотками и цепей питания! Правда, если в Паскале условия могут быть сколь угодно сложные (лишь бы программист в них не запутался!), то в RLL Plus существует ограничение: процессор DL имеет 8-уровневый логический стек, в который записывается результат проверки условия (0 или 1), а при выполнении логических операций OR, AND, XOR он продвигается на один уровень выше.

На практике такого набора условий, который вызвал бы переполнение логического стека, еще встречать не приходилось. Довольно часто встречаются ситуации, когда нужна только исполнительная часть ветви, которая должна выполняться всегда. Условие в этом случае тоже задавать необходимо, но это должно быть условие, которое выполняется всегда. В RLL Plus для этого предназначено реле SP1, которое всегда установлено.

Язык RLL Plus позволяет оперировать как отдельными битами, так и словами, а также двойными словами, причем все операции со словом (двойным словом) можно

выполнять только в аккумуляторе. Ячейку памяти можно лишь инкрементировать/декрементировать, но зато она может участвовать в условной части ступени:

Сравнить содержимое аккумулятора можно только командами CMP/CMPD/CMPF/CMPR и CMPS (доступна для DL260 и DL440), а результат определять по состоянию Специальных реле SP60, SP61, SP62. Правда, процессор DL440/450 в системной памяти содержит копию аккумулятора, и если к нему обращаться как к ячейке V-памяти, он тоже может использоваться в условной части. Это иногда бывает очень удобно, но процессоры семейства DL2**, к сожалению, копии аккумулятора не имеют. Необходимо помнить, что в начале каждого цикла сканирования содержимое аккумулятора очищается.

Оперативная память процессора (V-память, 7.5K для D2-250 и 15.5K для D2-260) удобна тем, что можно обращаться к ней как к отдельным битам, так и полным ячейкам. К ячейкам можно обращаться как напрямую, так и посредством указателей, т.е. когда одна ячейка памяти содержит адрес другой. Использование указателей очень удобно при обработке массивов, а также при работе с «объектом» в виде группы ячеек. Многие команды разрешают в качестве аргумента использовать не ячейку памяти, а указатель на нее, что делает программирование более гибким.

Некоторые адреса V-памяти имеют специальное назначение: так, например, адреса V0-V377 используются в качестве аккумуляторов таймеров T0-T377, к ним можно также обращаться, используя мнемонику TA. Биты входов (X-биты) начинаются с адреса V40400, для этого адреса правильным также будет обращение VX0 – если нужно обработать группу битов (диапазон битов также можно задавать в командах SET и RST), но чаще используется обращение к отдельным битам: X10, Y33, C100, S50. Для обращения к биту «обычной» ячейки используется следующая мнемоника: B1500.5 – это пятый бит ячейки V1500. Полные карты памяти процессоров DL имеются в главе «Спецификации и работа процессора» Руководства пользователя.

Процессор после подачи питания очищает всю V-память, за исключением области сохранения: для битов C,T,CT,S а также для V-памяти можно задать диапазон, который при подаче питания будет сохранен. Иногда это бывает удобно, хотя какую-то важную для работы программного обеспечения информацию сохранять таким образом мы не рекомендуем. Ее лучше размещать в блоках данных, расположенных в программе за пределами области сканирования (т.е. после команды END) и в процессе работы программы считывать в память. Содержимое этих блоков также допускается изменять и программными средствами, что немаловажно.

Процессор DirectLogic работает по замкнутой циклограмме, в которой прикладная программа выполняется на определенном этапе, которому предшествует считывание физического состояния входов в X-биты, а после выполнения программы состояние Y-битов переписывается в выходные модули, определяя физическое состояние выходов.

Таким образом, программист не должен заботиться о передаче управления на начало программы или какое-либо другое место: в новом цикле сканирования лестничная логика программы будет просмотрена и выполнена заново в зависимости от условий, предваряющих исполняемую часть ветви. Кроме того, при анализе X-битов нужно отдавать себе отчет в том, что анализируется их состояние, которое было в момент считывания, а не в момент выполнения программы. Время между этими событиями может

составлять от нескольких микросекунд до десятков миллисекунд, и реальное состояние входа может измениться. Если нужно знать состояние входа именно в момент выполнения программы, в языке RLL Plus существует команда, которая позволяет прочитать состояние входа именно в момент выполнения команды:

Аналогично также существуют команды, которые позволяют изменять состояние выходных битов в момент выполнения программы:

В программировании на RLL Plus обычно используются 8-ричная (для задания адресов ячеек памяти и битов) и 16-ричная (для обработки данных) системы счисления. Для удобства 16-ричные числа можно представлять в виде двоично-десятичного числа, и некоторые команды требуют в качестве аргумента именно двоично-десятичное число, что облегчает программирование расчетов. Наличие нескольких систем счисления может привести к путанице и является богатой почвой для ошибок. Можно еще пользоваться действительными числами, но здесь возможность ошибиться несколько меньше.

Единственное, что можно здесь посоветовать – сразу определиться, в какой системе будут представлены числа, и пользоваться соответствующими командами. Начинающие программисты на RLL Plus обычно задают вопрос: а как узнать, какое у нас число – 16-ричное или двоично-десятичное? С «точки зрения» процессора все числа 16-ричные, а вот как их интерпретировать – решает программист, т.е. он выбирает, какими командами он будет изменять содержимое ячейки V-памяти. Двоично-десятичное сложение, вычитание, умножение и деление приведут к двоично-десятичному результату.

Все команды языка RLL Plus делятся на несколько групп, большинство из них работает с аккумулятором, некоторые используют и его стек. Поддерживаются все математические операции как с двоично-десятичными, так и 16-ричными, а также с действительными числами, и все логические операции. Имеется очень мощный набор команд для работы с битами и для преобразования чисел. В систему команд процессора DL260 добавлены команды для работы с таблицами и строками, ряд команд для выполнения тригонометрических функций. Полное описание системы команд приведено в Руководстве пользователя, глава «Стандартные команды RLL».

7. Программирование ручного режима.

Система управления большинства технологических установок часто включает режим ручного управления исполнительными устройствами с операторской панели. При программировании ручного режима управления необходимо решить следующие задачи:

- защита от дребезга контактов кнопок, переключателей;
- независимая отработка действия каждым исполнительным устройством;
- блокировка некорректных действий;
- выявление и диагностирование ошибок.

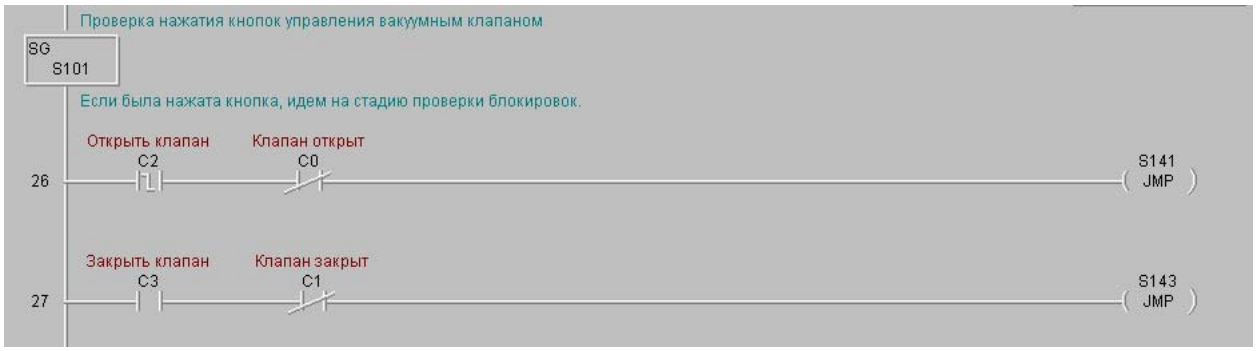
Попробуем применить объектно-ориентированный подход. Пусть у нас имеется некое исполнительное устройство; например, вакуумный клапан, который имеет 2 состояния: «Открыт» и «Закрыт» (входы X0 и X1), эти состояния изменяются с помощью

выходов «Открыть» и «Закрыть» (выходы Y0 и Y1) по нажатию кнопок, соединенных с входами X2 – «Открыть клапан» и X3 – «Закрыть клапан». Для того, чтобы избежать дребезга и обеспечить независимую работу клапана, воспользуемся преимуществами стадийного программирования, которые предоставляет язык RLL plus, и разобьем процессы открытия и закрытия клапана на отдельные стадии:

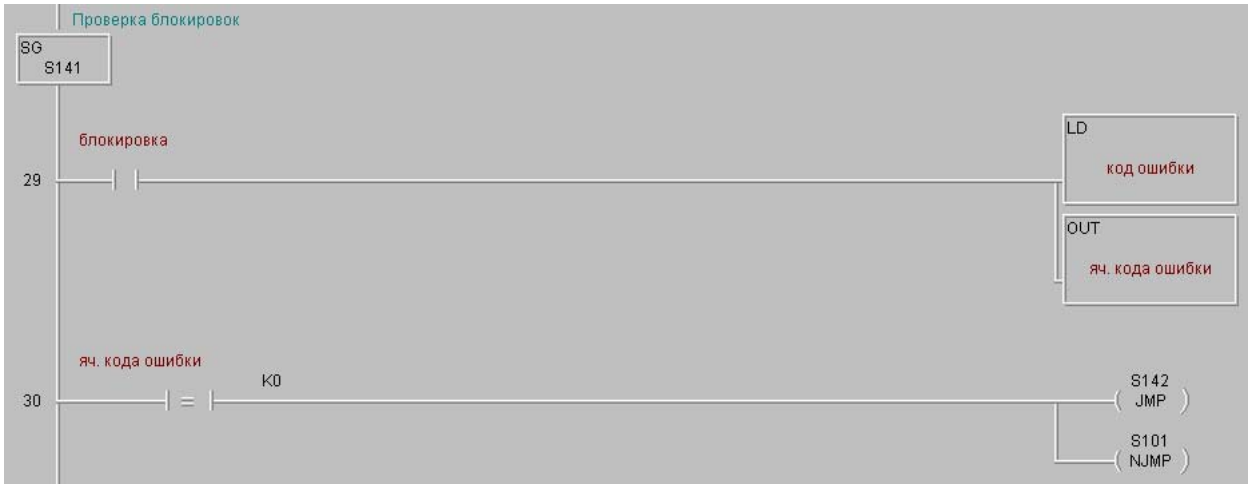
1) опрос кнопки управления;
2) проверка допустимости операции;
3) выполнение операции;
4) проверка результата выполнения и возврат к опросу кнопок.

Таким образом, мы создали объект «Вакуумный клапан», независимый от остальных органов управления системы, «объекты» которых можно создать аналогичным образом.

Ветви программы, реализующей данный подход, в упрощенном виде будут выглядеть так (вместо X-входов используем их образы в виде C-битов):



Стадии опроса кнопок управления другими механизмами запускаются параллельно с этой и активны, пока не будет нажата соответствующая кнопка. После этого управление передается другой стадии, на которой проверяется допустимость этой операции, а дребезг на входе, даже если он будет иметь место, программа просто «не заметит», так как до конца выполнения операции вход опрашиваться уже не будет. Если условие корректности операции выполняется (никаких блокировок нет), переходим на стадию выполнения операции. Ее необходимо разобрать более тщательно. Дело в том, что мы не можем просто установить соответствующий выход и со спокойной совестью идти на возобновление опроса кнопок. Мы ведь не знаем, чем закончилась операция! А если мы подали сигнал на открытие клапана, но он не открылся, т.е. мы не получили сигнала на входе «Клапан открыт»? Может случиться и такое – включили выход, но все равно получаем сигнал, что клапан закрыт. При отработке операции, даже самой простейшей на первый взгляд, нужно проверять все. А на выполнение операции задавать некоторое время, которое подбирается опытным путем. Внеся в нашу программу ветви,

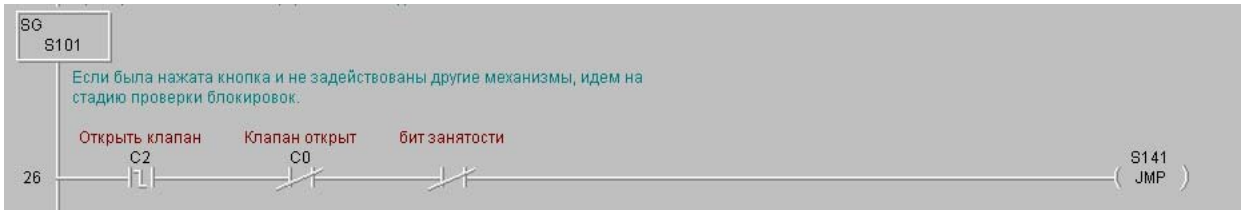


выполняющие эти проверки, мы всегда сможем адекватно диагностировать неполадку, если она вдруг произойдет.

В заключительной стадии нашего цикла обслуживания механизма проверяем наличие ошибки, и, если она произошла, производим ее обработку (например, индикацию или запись в очередь), после чего возобновляем опрос кнопок.

Конечно, если механизмов в системе много, программирование их функций по такой схеме представляется достаточно рутинной работой, но для нас главное – надежность.

Если же одновременная работа нескольких механизмов не допускается, можно ввести специальный «бит занятости» и в стадии проверки нажатия кнопки проверять, сброшен ли этот бит:



8. Программирование автоматического цикла.

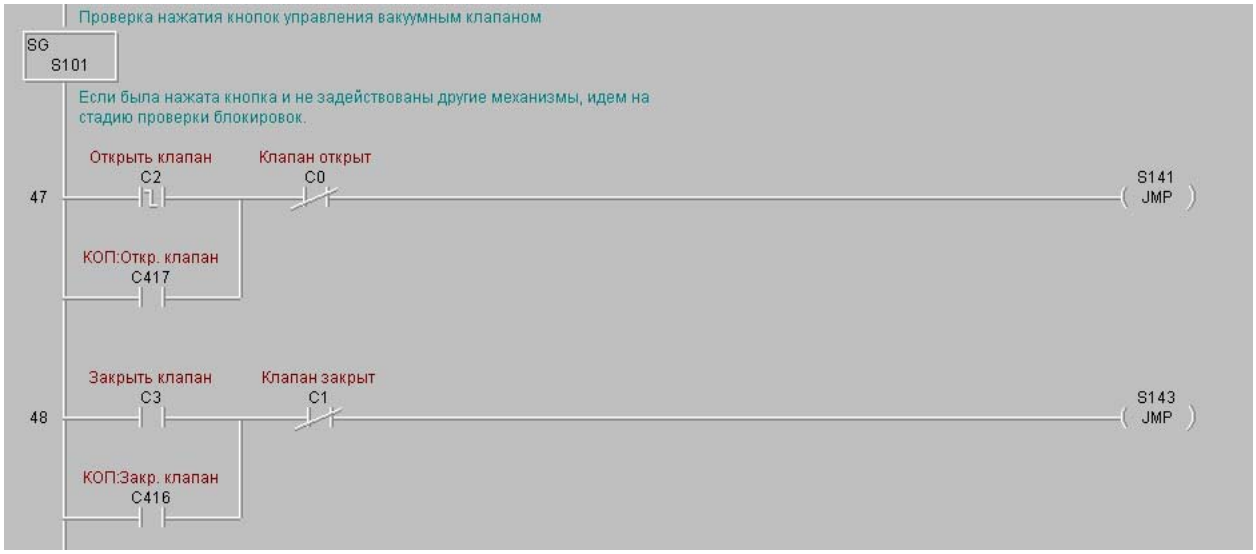
Для того, чтобы программа управления технологической установкой обеспечивала успешное и гибкое функционирование в автоматическом режиме, при ее создании необходимо решить следующие задачи:

- возможность запуска одновременно нескольких независимых технологических процессов;
- легкость изменения последовательности технологических операций;
- использование в полной мере наработок по ручному режиму управления.

Последнее предполагает, что мы будем использовать весь цикл стадий, относящихся к ручному режиму. Это, во-первых, даст возможность заблокировать при отработке цикла кнопки ручного режима, а во-вторых, избавит от необходимости отдельно программировать каждую последовательность производственных операций – в этом случае изменение последовательность операций в цикле будет представлять собой достаточно трудоемкую и рутинную задачу. Тогда перед нами встает другая задача – как объединить независимые стадии, обслуживающие ручное управление исполнительными устройствами, в некую последовательность, предусмотренную конкретной технологией? Другими словами – как «заставить» стадии, в которых опрашиваются кнопки ручного управления, включать в определенной последовательности имеющиеся в системе управления механизмы?

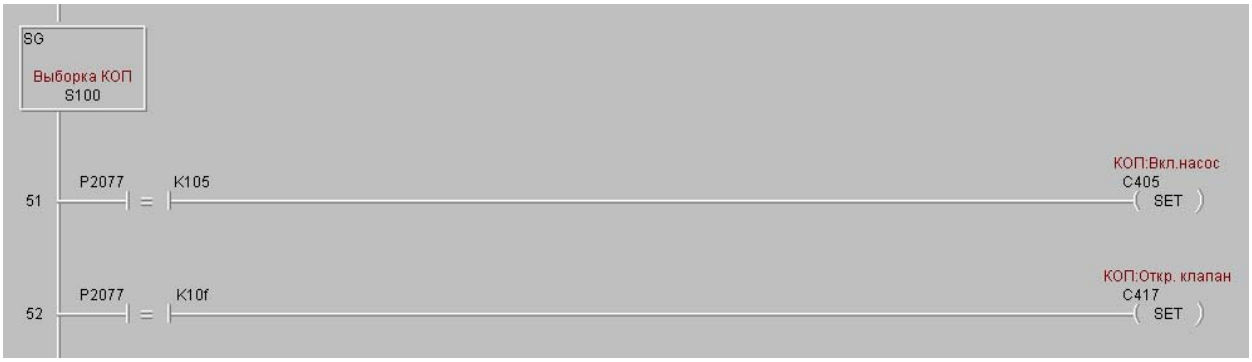
Ответ находится в самом вопросе: если в режиме ручного управления механизм включается по нажатию кнопки, т.е. установленному биту, то почему бы в автоматическом режиме не включать его также по наличию факта установленного бита, но – другого? Замечательно, но кто тогда будет устанавливать этот бит? Ответ – бит будет устанавливаться в специальной стадии, которая запускается по нажатию кнопки пуска автоматического цикла. Откуда эта стадия знает, какой бит нужно установить? Ответ – эта стадия занимается тем, что просматривает ячейки памяти, в которых и записаны номера нужных битов, которые требуется последовательно установить, чтобы обеспечить выполнение технологических операций в определенном порядке. Следовательно, создание производственного цикла сводится, таким образом, к созданию области числовых данных, в которой в 16-ричном коде будем хранить номера битов, а также к программированию стадии, которая и будет устанавливать эти биты.

В приведенном примере 16-ричная константа 105 – это 8-ричное число 405, 10F – 417, а 107 – это 407 в 8-ричной системе исчисления. Назовем эти константы Кодом операции (КОП). Таким образом, чтобы выполнить, допустим, операцию открытия клапана, нужно установить бит С417, а в стадию опроса кнопки управления этим механизмом внести проверку этого бита. Тогда эта стадия будет выглядеть так:

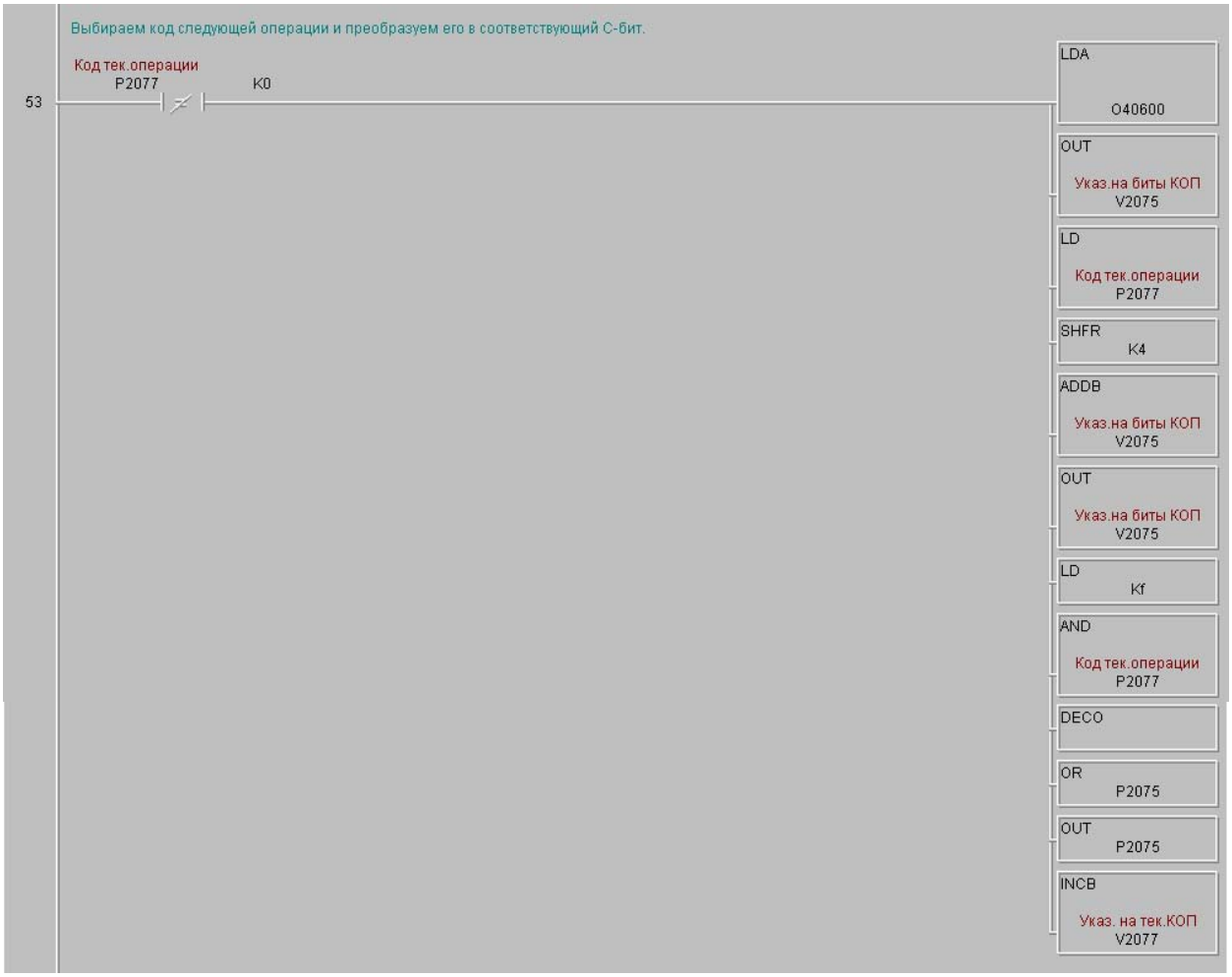


По нажатию кнопки «Пуск» считываем коды операций в V-память и создаем указатель на этот массив. Пусть в нашем примере цикл состоит из 20 операций (количество слов задаем в 16-ричном коде). Последовательность кодов завершаем нулем – это будет означать, что выборку нужно прекратить. Это придется делать не только после выполнения всех операций цикла, но и в случае возникновения на каком-нибудь этапе ошибки.

Теперь напишем стадию выборки КОП из V-памяти. Само по себе это не представляет затруднений: в процессе выборки проверять содержимое ячейки, на которую ссылается указатель, на 0, и модифицировать его. Главное в этой стадии – преобразовать 16-ричный код операции в соответствующий С-бит. (Заметим, она должна отработать раньше стадий проверки кнопок ручного управления и битов КОП) Разумеется, задача элементарно решается:



Однако можно все сделать гораздо короче и изящнее, поскольку в языке RLL Plus есть парочка команд, которых во всех известных языках программирования нет – это команды DECO (устанавливает бит по коду в аккумуляторе) и ENCO (выполняет обратную операцию – установленный в аккумуляторе бит превращает в числовое представление). По коду операции вычислим адрес ячейки V-памяти, в которой находится искомый C-бит (пространство C-битов начинается с адреса V40600), и установим соответствующий бит в аккумуляторе, после чего скопируем его в нужный C-бит:

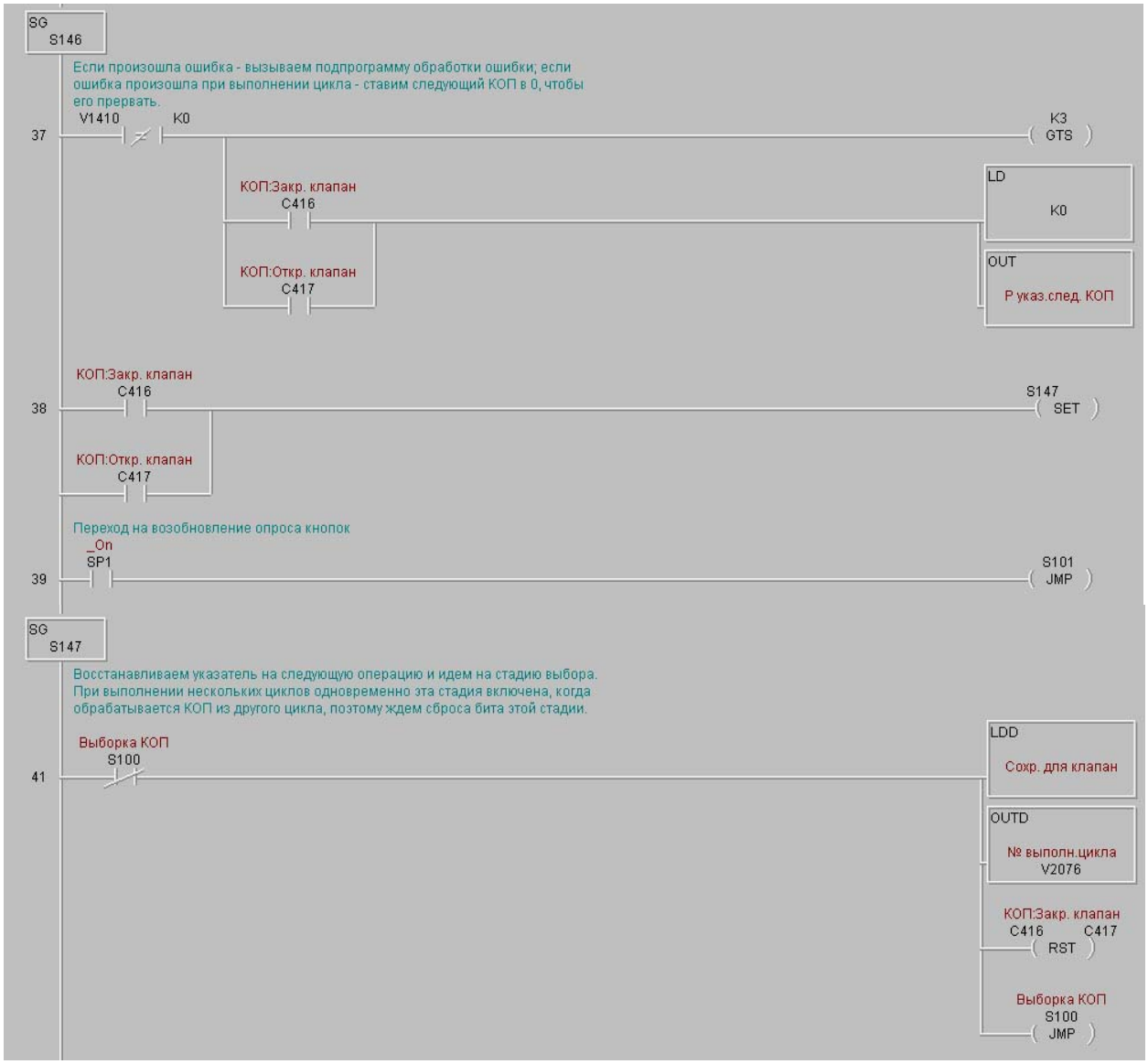


После установки C-бита кода операции стадия выборки КОП должна быть деактивирована. Это можно сделать командой RST S100, но рациональнее перейти на другую стадию, которую в большинстве случаев придется вводить, потому как в автоматическом цикле могут быть операции, для которых ручное управление вообще смысла не имеет; например, ожидание рабочего уровня вакуума в камере в течение

некоторого времени. Этим операциям также ставятся в соответствие свои коды и С-биты, которые и будут проверяется в стадии, активированной из стадии S100.

Если технологическая установка имеет несколько независимых производственных циклов (причем они могут выполняться одновременно), тогда для каждой операции зарезервируем две ячейки V-памяти, в которых будут храниться номер выполняемого цикла (эту ячейку еще можно использовать в качестве слова состояния цикла) и адрес следующего кода операции. Соответственно потребуется ввести ячейку, где будет храниться номер выполняемого в данный момент цикла. В стадию опроса кнопки ручного управления и установки бита КОП добавится новая ветвь, и стадия примет следующий вид:

Также нужно внести изменения в заключительную стадию обслуживания механизма: в случае ошибки нужно записать 0 вместо адреса кода следующей операции, чтобы прервать цикл, и если установлены биты КОП, активировать стадию, в которой будем ожидать готовности возобновить данный цикл. Сигналом для этого служит сброшенный бит стадии S100 – значит, она установила бит КОП и была деактивирована, теперь активируем ее из стадии ожидания и загружаем номер текущего цикла и адрес КОП, который должен быть выбран в стадии S100. Соответственно придется видоизменить обработку кнопки «Пуск»: просто установить флаг пуска и в следующей ветви в случае, если он установлен и сброшен бит стадии S100, загрузить в V-память коды операций цикла, номер цикла и адрес массива КОП, после чего активировать S100. Пример программы имеется выше, с той лишь разницей, что вместо бита «Кнопка Пуск» будет стоять бит «Флаг Пуска».

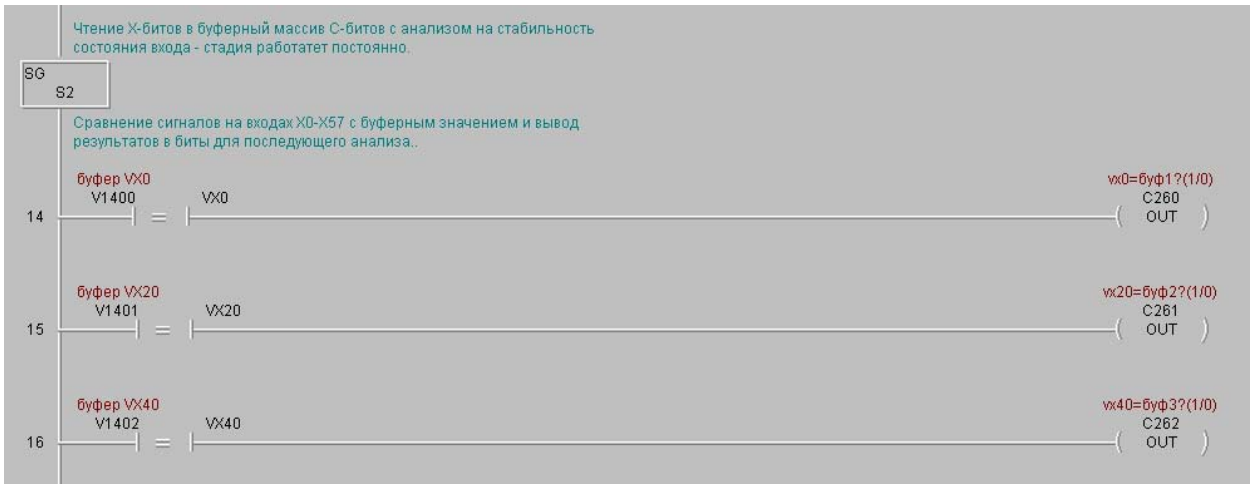


Сразу заметим, что данный подход к реализации одновременной отработки нескольких циклов не совсем безупречен: если в параллельных циклах задействованы одни и те же операции, то теоретически возможен момент, когда в текущем цикле будет установлен бит некоторой операции, а в настоящий момент не до конца выполнилась противоположная ей операция, т.е. выполняли «Закрыть», а текущая команда «Открыть». В этом случае текущая операция выполнена не будет, а выполнение текущего цикла остановится – он просто «потеряется». Однако здравый смысл подсказывает, что подобная ситуация вряд ли возможна – в самом деле, зачем запускать параллельно выполняющемуся циклу еще один, который выполняет противоположные операции. Если же подобные циклы предусмотрены в системе управления, тогда в программу необходимо добавить дополнительные блокировки, которые исключают одновременный запуск таких циклов.

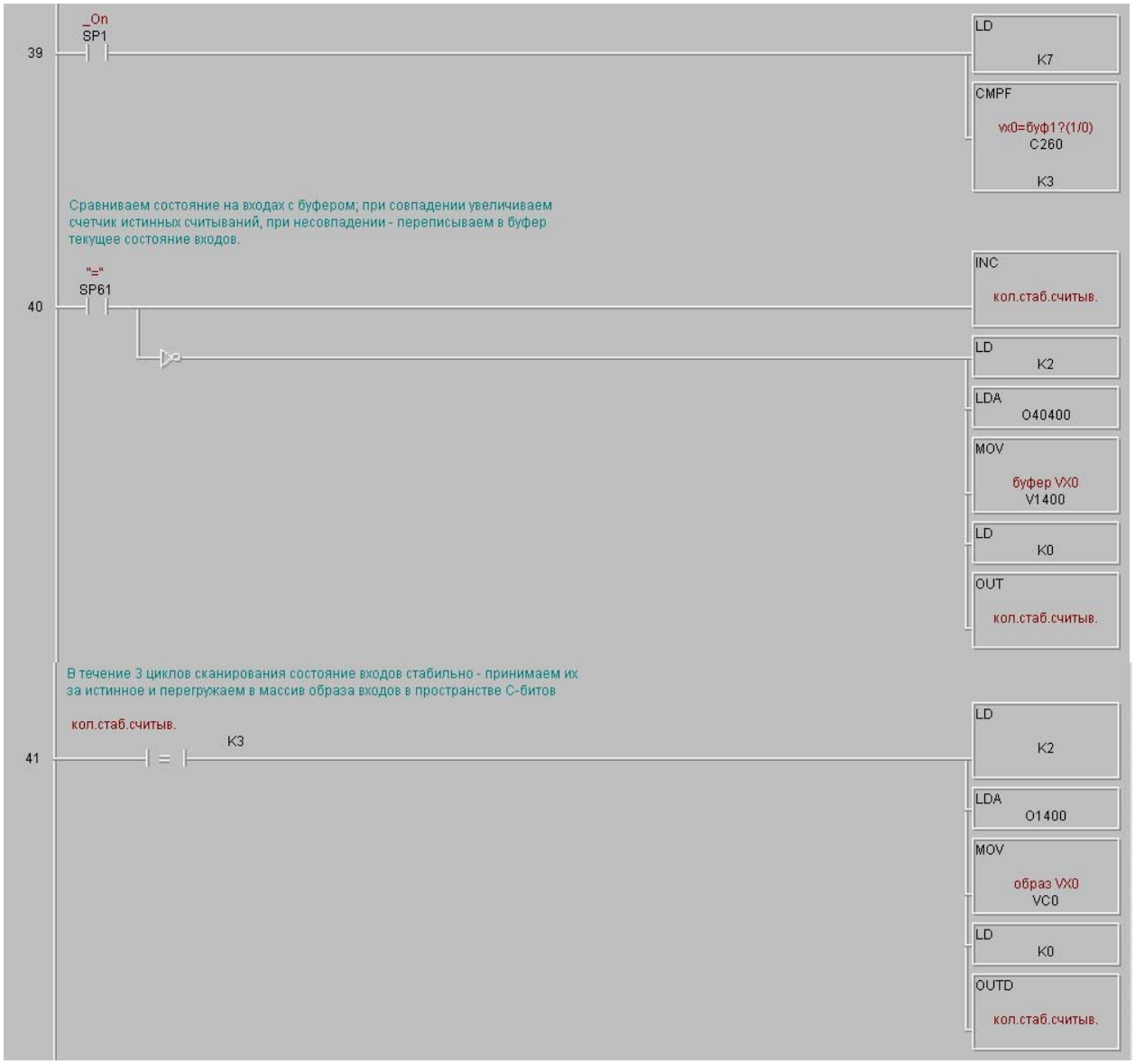
9. Чтение входов.

В системах управления многих технологических установок дискретные входы обычно отображают состояния переключателей, тумблеров, датчиков с состояниями «открыто» или «закрыто». Когда после выполнения операции показание датчика изменяется на противоположное, т.е механизм меняет свое положение, на практике имеет

место дребезг контактов, и по однократному считыванию дискретного входа еще нельзя с уверенностью констатировать то или иное состояние датчика. Для того, чтобы повысить достоверность считывания X-битов контроллера, можно воспользоваться так называемым виртуальным пространством входов, в котором отображаются состояния X-битов, не меняющиеся по крайней мере в течение 3 циклов сканирования. Для создания виртуального пространства входов задействуем C-биты – в них будем перегружать соответствующие X-биты, если их состояние не изменится в течение 3 циклов. Разумеется, использовать эти биты для других целей в программе уже нельзя. Сразу оговоримся, что этот подход применим для систем, не требующих мгновенного реагирования на изменение входа. Для считывания входов напишем стадию, которая всегда будет активна. Для примера будем считать, что в нашей системе 48 дискретных входов и адресация начинается с VX0. Зарезервируем 3 ячейки для хранения промежуточного состояния входов: V1400-V1402 и 3 C-бита в качестве флагов совпадения считанного значения и промежуточного: C260-C262. Сначала сравниваем состояние входов с промежуточными значениями и выставляем флаги совпадения (для удобства обращаемся к дискретным входам как к ячейкам V-памяти):



Если состояние входов не изменилось, увеличиваем счетчик стабильных считываний, иначе – принимаем текущее состояние входов за истинное и записываем его в буферные ячейки. Когда получим 3 стабильных считывания, будем считать, что состояние входов устойчивое, и можно перегрузить буферные ячейки в C-биты, соответствующие входам. Фрагменты программы, реализующей все вышесказанное, приведены ниже:



Теперь, чтобы проанализировать в программе состояние какого-либо дискретного входа, будем обращаться к С-битам.

10. Измерения.

Выбор модулей для аналогового ввода-вывода информации у контроллеров DirectLogic достаточно широкий. Работа с каждым из них подразумевает профессиональный подход.

10.1. Особенности работы с модулем F2-04THM.

Модуль F2-04THM предназначен для автоматического преобразования сигналов с термопар в показания температуры (в градусах Цельсия либо Фаренгейта), а также для преобразования вольтового (±5В) или милливольтового (±156мВ) сигнала в 16-битовые цифровые значения.

Модуль обеспечивает достаточно высокую точность измерений, не требует калибровки. Мы не будем подробно описывать его технические характеристики – все они приведены в документации, которую необходимо изучить перед применением модуля.

Данная глава посвящена особенностям работы с этим модулем, которые в документации не оговорены.

На первый взгляд, обработка показаний модуля не должна вызывать каких-либо сложностей, поскольку он поддерживает 9 типов термопар и еще два уровня потенциальных входов. Однако, не все встроенные градуировки соответствуют отечественным термопарам. Разумеется, можно применять и импортные термопары, но они весьма дороги.

Один из подходов к решению этой задачи – использование потенциальных входов. Без их применения не обойтись также, если нам нужно измерить не температуру, а, например, давление, и в качестве источника сигнала выступает вакуумная лампа. Полученное цифровое представление затем преобразуется в показания температуры (давления).

В штатной ситуации модуль функционирует безупречно, но время от времени происходит такая неприятная вещь как обрыв датчика, и вот тут начинаются проблемы. Несмотря на экранирование и изоляцию каналов, начинаются наводки на показания всех каналов всех модулей F2-04THM, используемых в системе управления.

Программное обеспечение, анализирующее показания датчиков, может в результате неадекватных показаний нарушить нормальную логику работы технологической установки: выдать сигналы несуществующих на самом деле ошибок, заблокировать какие-то механизмы и вообще спровоцировать аварийную ситуацию. К сожалению, в документации этой задаче внимание почти не уделяется, а в главе, посвященной написанию управляющей программы для чтения показаний модуля, настоятельно рекомендуется использовать для этого метод указателя. Этот метод существенно упрощает требования к программированию – процессор D2-250(D2-260) содержит специальные ячейки V-памяти, назначенные каждому слоту каркаса, в них заносится число обновляемых каналов и местоположение выходных данных, для получения результата достаточно прочитать ячейки с выходными данными. Никаких проверок на достоверность данных и обрыв датчика этот метод не предусматривает! Мы рекомендуем пользоваться методом мультиплексирования – во-первых, его алгоритм в каждом цикле сканирования позволяет определить, какой именно канал читается или в каком канале появился обрыв, а во-вторых, он более универсален, так как только его можно использовать для обработки показаний термопарных модулей, установленных в каркасе удаленного ввода-вывода.

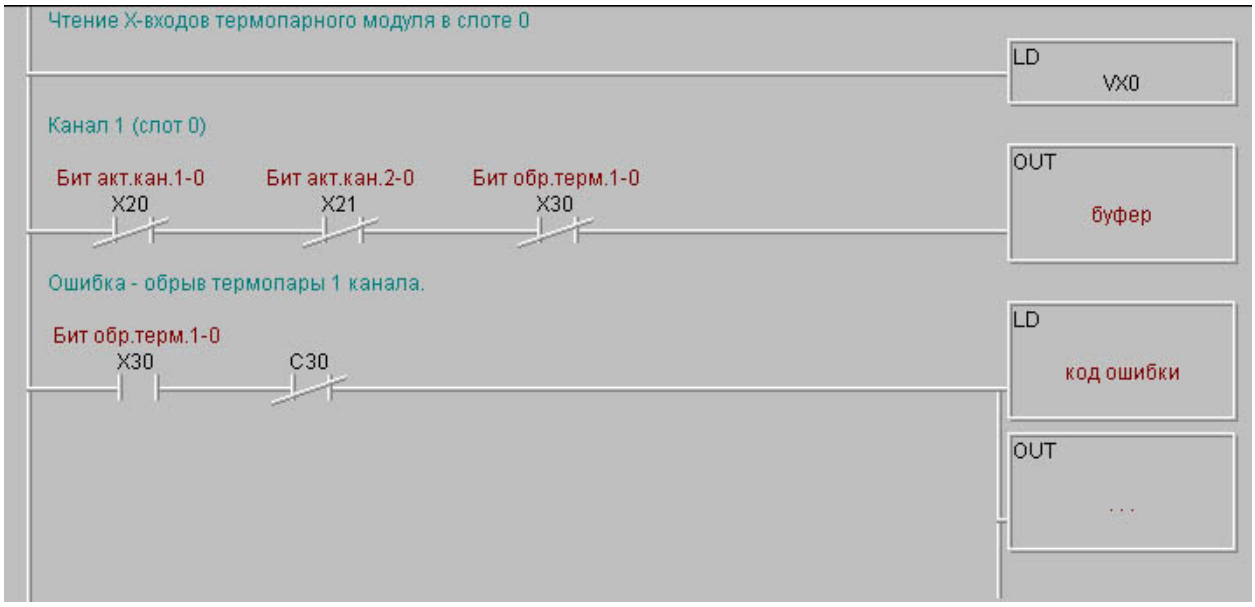
Алгоритм метода мультиплексирования достаточно прост. Как известно, модуль F2-04THM требует 32 дискретных входа процессора. Эти входы можно (и нужно!) использовать, чтобы получить: а) признак активного канала, б) цифровое представление аналогового сигнала, в) диагностическую информацию о состоянии модуля. В нашем примере предположим, что модуль установлен в 0-й слот каркаса; стало быть, входные точки имеют адреса V40400 (VX0) и V40401 (VX20). В ячейку V40400 записываются данные активного канала, в ячейке V40401 находится информация о состоянии модуля – номер активного канала и биты обрыва. Номер активного канала в данном случае будут определять биты X20 и X21:

X21	X20	Канал
0	0	1
0	1	2
1	0	3
1	1	4

Биты обрыва датчика:

Бит	Канал
X30	1
X31	2
X32	3
X33	4

Фрагмент программы, обеспечивающей чтение показаний датчика 1-го канала, будет выглядеть примерно так:



Остальные каналы обрабатываются аналогично.

Однако, если мы настроили модуль на потенциальные входы, бит обрыва устанавливается только в случае нарушения питания! Если имеет место обрыв датчика, в случае использования потенциальных входов обнаружить его можно лишь по косвенным признакам, которыми являются броски показаний или же их заведомая абсурдность. Бит обрыва диагностирует обрыв датчика только если модуль настроен на один из типов термопар!

Итак, для правильного диагностирования обрыва целесообразно отказаться от использования потенциальных входов и настроить модуль на один из типов термопар. Заметьте, «настроить» не означает «применить»!

Допустим, у нас имеется отечественная термопара типа ВР-2, и нам нужно получить значения температуры и избежать наводок вследствие обрыва. Как было сказано выше, единственный тип отечественных термопар, поддерживаемых модулем F2-04THM – это ТХА (тип К). Настроим модуль на этот тип термопары и сравним градуировочные таблицы термопар ТХА и ВР-2. Максимальному значению температуры, которую можно измерить термопарой ВР-2 – 1800°С – соответствует 27.226 мВ; нетрудно оценить, что для термопары ТХА такое напряжение соответствует примерно 660°С. Выпишем из градуировочной таблицы ВР-2 значения температур и напряжения:

Температура ВР2, °С	Напряже-ние, мВ
0	0
10	0,118
20	0,241
30	0,367

40	0,497
…	…

В градуировочной таблице ТХА найдем интервал напряжений, в котором находится напряжение, соответствующее градусам ВР-2. Наша таблица приобретет следующий вид:

Температура ВР2	Напряже-ние, мВ	Интервал напряжений ТХА, мВ		Градусы ТХА	
0	0		0	0	0
10	0,118	0,079	0,119	2	3
20	0,241	0,238	0,277	6	7
30	0,367	0,357	0,397	9	10
40	0,497	0,477	0,517	12	13
…	…	…	…	…	…

Теперь легко вычислить, сколько градусов ТХА будет соответствовать градусам ВР-2 (будем считать характеристику ТХА в интервале 1 градуса линейной):

Температура ВР2	Напряже-ние, мВ	Температура ТХА*10
0	0	**0**
10	0,118	**30**
20	0,241	**61**
30	0,367	**93**
40	0,497	**125**
…	…	**…**

Произведя вычисления для всех значений ВР-2 и преобразовав результат в 16-ричный код, получаем таблицу, по которой и будем рассчитывать температуру:

Таблица соответствия дискрет, принимаемых от термопары типа ВР, температуре, если термопарный блок F2-04THM настроен на термопару ТХА (тип К)

Температура ВР2	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	0	1E	3D	5D	7D	9E	C0	E3	106	129
100	14D	171	196	1BD	1E1	207	22D	253	27A	2A1
200	2C8	2F0	317	33F	367	38F	3B8	3E0	409	432
300	458	484	4AE	4D7	501	52B	555	580	5AA	5D5
400	5FF	62A	655	680	6AB	6D6	701	72C	757	782
500	7AD	7D8	803	82E	858	883	8AE	8D8	902	92C
600	956	980	9AA	9D4	9FD	A27	A50	A79	AA2	ACB
700	AF4	B1D	B45	B6E	B96	BBE	BE6	C0E	C36	C5E
800	C86	CAD	CD5	CFC	D23	D4A	D71	D98	DBF	DE6
900	E0C	E33	E59	E7F	EA5	ECB	EF1	F16	F3C	F61
1000	F86	FAB	FD0	FF5	101A	103E	1063	1087	10AB	10CF
1100	10F3	1117	113B	115E	1182	11A5	11C8	11EB	120E	1231
1200	1253	1276	1298	12BA	12DC	12FE	1320	1342	1363	1385
1300	13A6	13C8	13E9	140A	142A	144B	146C	148C	14AD	14CD
1400	14ED	150D	152D	154D	156D	158C	15AC	15CB	15EB	160A

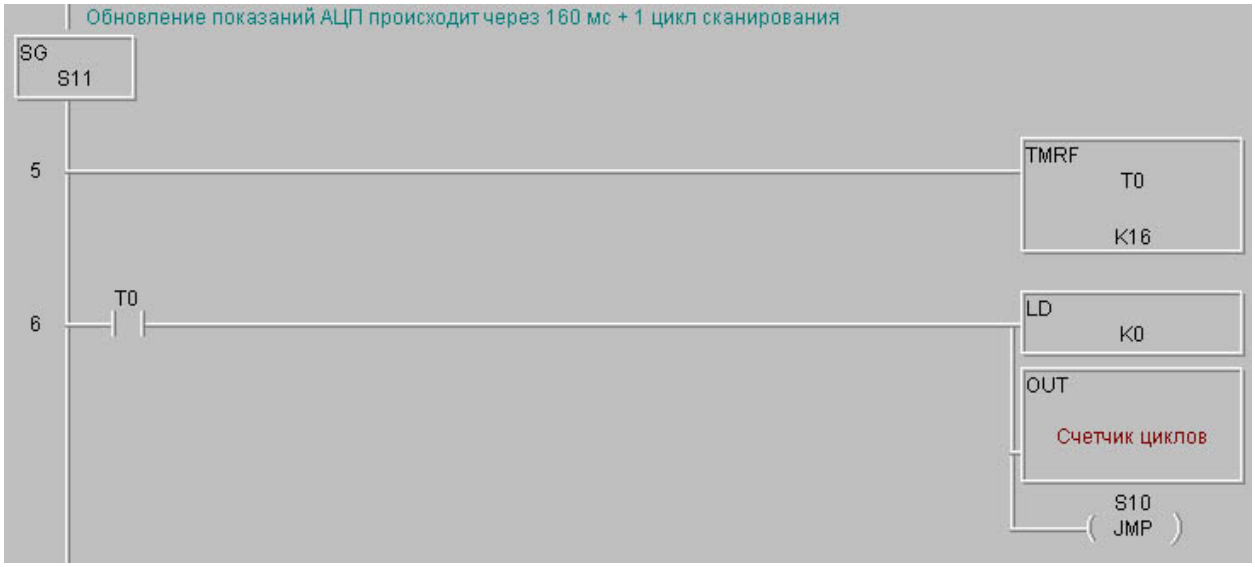
1500	1629	1648	1667	1686	16A4	16C3	16E1	1700	171E	173C
1600	175A	1778	1795	17B3	17D1	17EE	180B	1828	1845	1862
1700	187F	189B	18B8	18D4	18F0	190C	1927	1943	195E	1979
1800	1994									

Итак, мы настроили модуль таким образом, чтобы при обрыве термопары устанавливался бит, получили таблицу пересчета показаний термопары одного типа в значения другого типа, а что дальше? Если обрыв произошел, продолжать измерения невозможно: из-за наводок результаты будут неверны. Единственный выход из такой ситуации – замкнуть накоротко канал, в котором произошел обрыв. При разработке электрической схемы установки нужно встроить в нее цепь, которая при установлении бита обрыва либо автоматически замкнет канал, либо один из выходов, имеющихся на каркасе контроллера, задействовать для выполнения функции замыкания:

После замены неисправного датчика канал необходимо вернуть в цепь измерений. Для этого наиболее рациональным решением представляется добавление на панель оператора технологической установки специальной кнопки, соединенной с одним из X-входов. По нажатию этой кнопки бит замыкания сбрасывается:

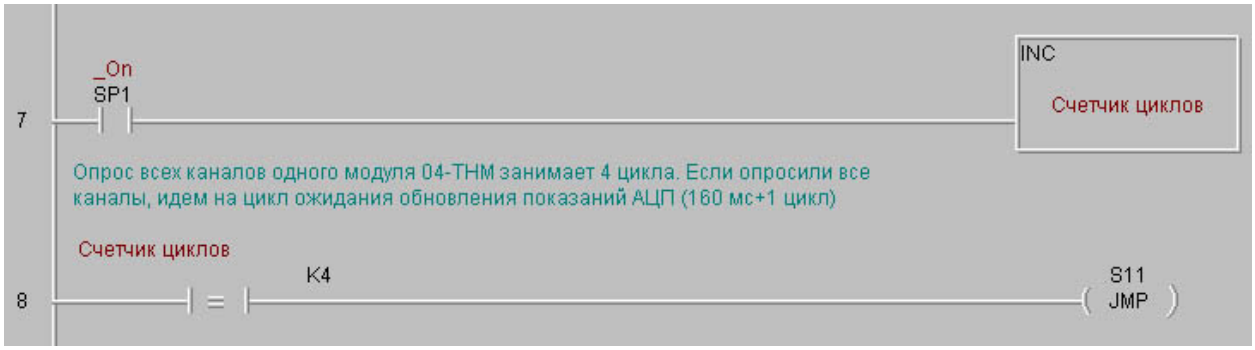
10.2. Обработка измерений

Теперь, когда мы решили проблемы с «нестандартными» для модуля термопарами и с обрывом, займемся непосредственно измерениями. Согласно документации, модулю для считывания показаний и копирования их в V-память требуется 160 мс на канал плюс один период сканирования, т.е. нам нет необходимости получать результаты измерений в каждом цикле сканирования (если, конечно, цикл сканирования не превышает 160 мс), иначе мы будем считывать одно и то же значение. Поскольку для считывания данных мы будем пользоваться методом мультиплексирования, построим алгоритм считывания следующим образом: через 160 мс плюс один цикл включаем стадию опроса всех задействованных каналов; проверяя биты активного канала, за 4 цикла получим данные 4-х каналов.



Переход на Стадию 10 произойдет с следующем цикле сканирования, так мы выполняем условие обновления показаний каналов.

В Стадии 10 проверяем биты активности каналов и биты обрыва датчика, как было показано выше, в конце стадии проверяем счетчик циклов:

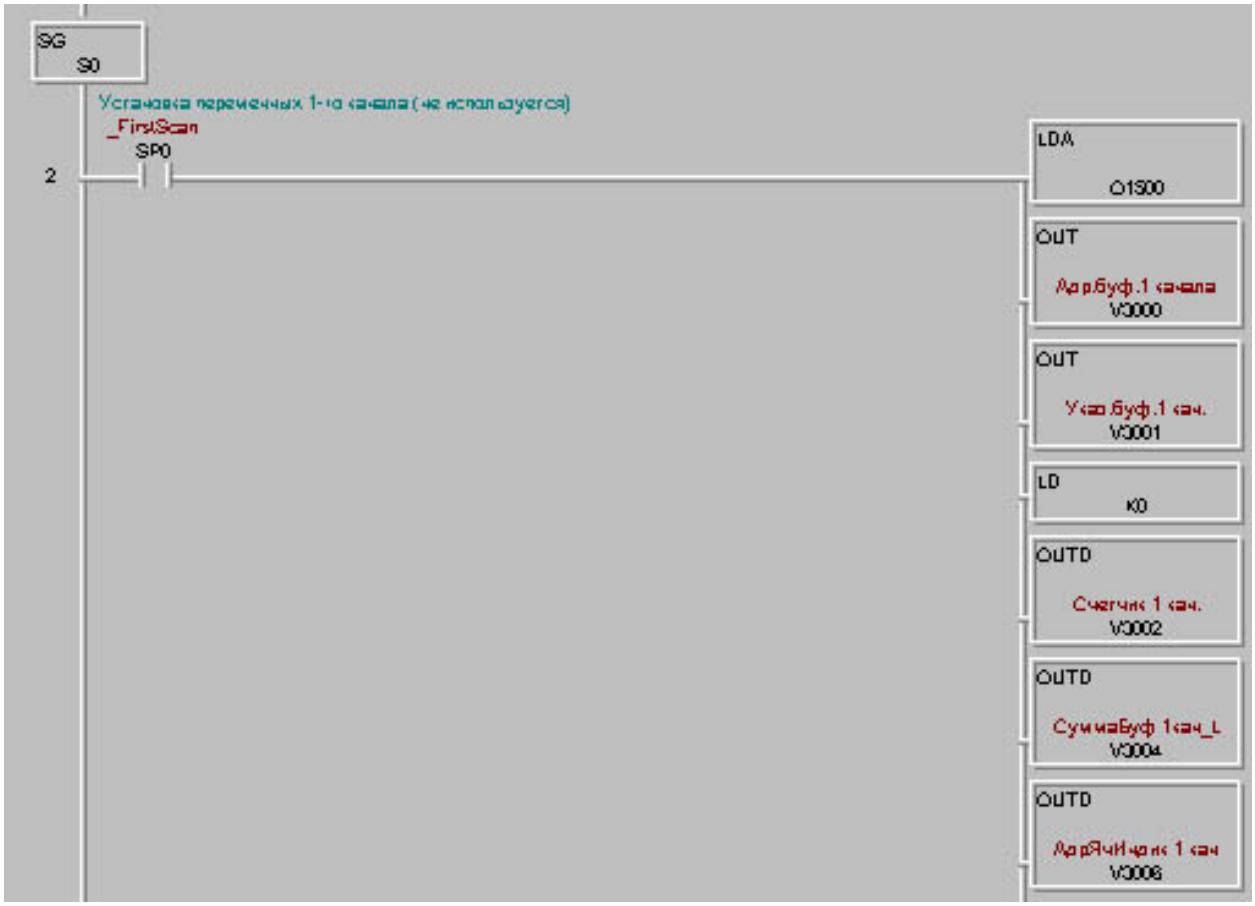


Следующий вопрос – что делать с результатами измерений? Использовать только что полученный результат для индикации либо каких-то дальнейших расчетов и выводов представляется нерациональным, поскольку, во-первых, результаты всегда отличаются последней цифрой вследствие высокой чувствительности модуля; во-вторых, при частой индикации мы будем наблюдать неудобное для глаза мерцание результата; в-третьих, мы не сможем правильно определить тенденцию к изменению показаний; в-четвертых, мы не сможем отфильтровать ошибочные показания, возникшие по причине помех. Обычный способ обработки показаний (и здесь мы не открываем ничего нового) – это накопление полученных результатов в буфере и последующее их усреднение. Обычно оптимальное количество значений в буфере – 8 или 10, в этом случае цикл обновления индикации будет примерно полторы секунды, что вполне приемлемо для визуального контроля. Этот интервал также вполне приемлем для решения задач регулирования.

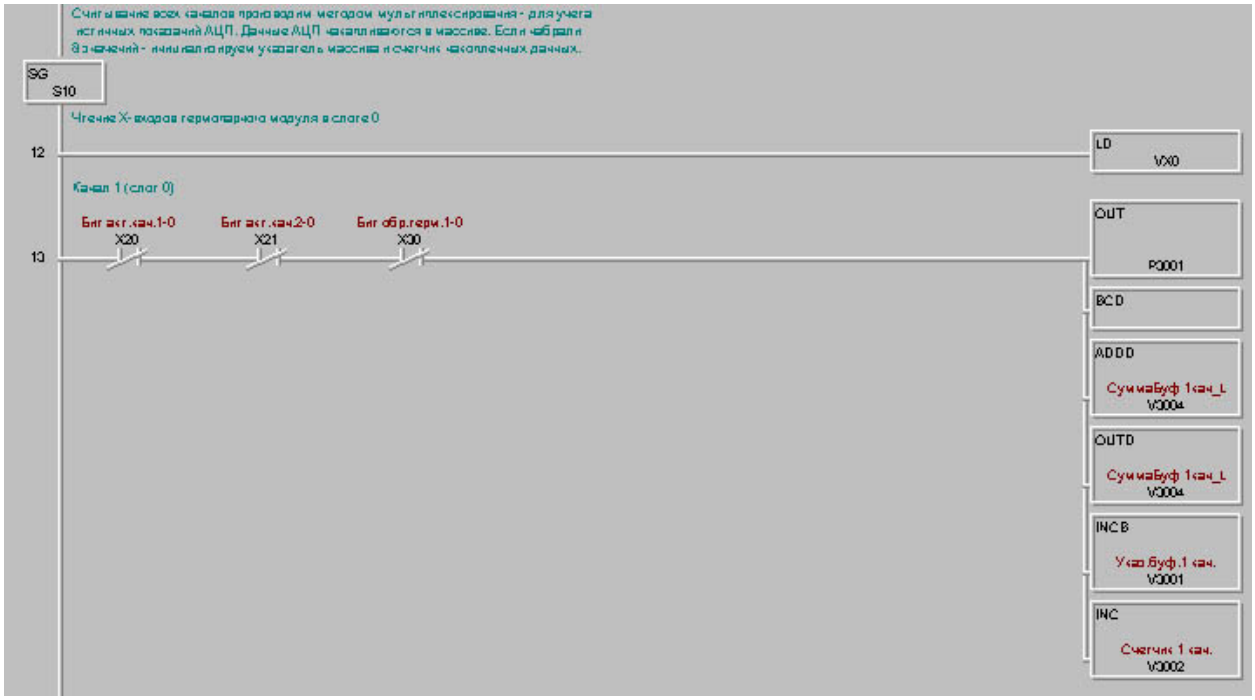
Для организации буфера нам понадобятся 8 последовательных ячеек V-памяти, ячейка-указатель на элемент буфера, ячейка-счетчик количества считываний, ячейка-адрес буфера, а также две ячейки, в которых будет накапливаться сумма всех 8 считываний по данному каналу. По аналогии с объектно-ориентированным программированием можно сказать, что мы создали класс под названием Канал считывания, а ячейки V-памяти – переменные этого класса. Это не то чтобы дань моде, просто практика показывает, что так удобнее – поскольку принципы считывания

показаний каналов и их последующей обработки одинаковы, программирование становится более простым и несколько приближенным к объектно-ориентированным языкам высокого уровня.

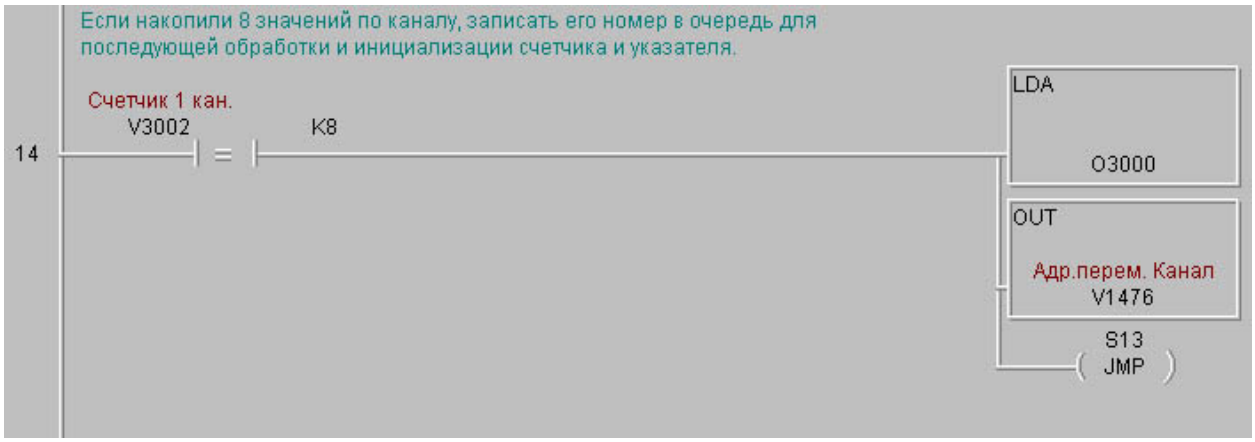
Создание переменной объекта типа Канал считывания мы проводим в Стадии 0, которая обычно используется для настройки и инициализации переменных, используемых в программе. Для примера используем один канал; пусть его буфер начинается с ячейки V1500, а переменные этого канала разместим в группе ячеек, начинающейся с адреса V3000. Для усиления тезиса объектно-ориентированного подхода добавим к переменным класса еще одну – ячейку, в которую будет помещаться результат обработки для индикации. Вот так будет выглядеть создание одной переменной типа Канал считывания:



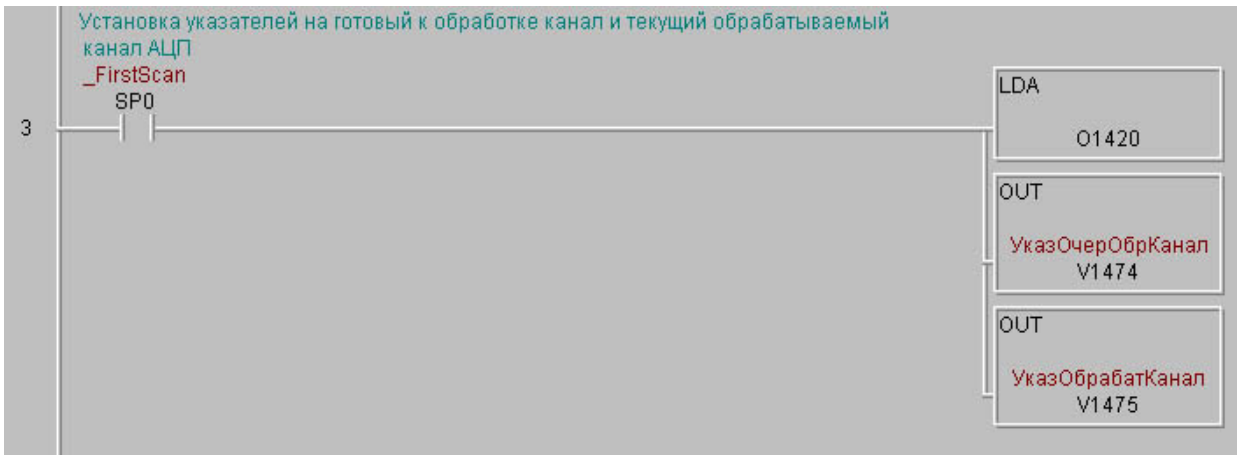
Тогда процедура считывания показаний по этому каналу приобретет следующий вид:



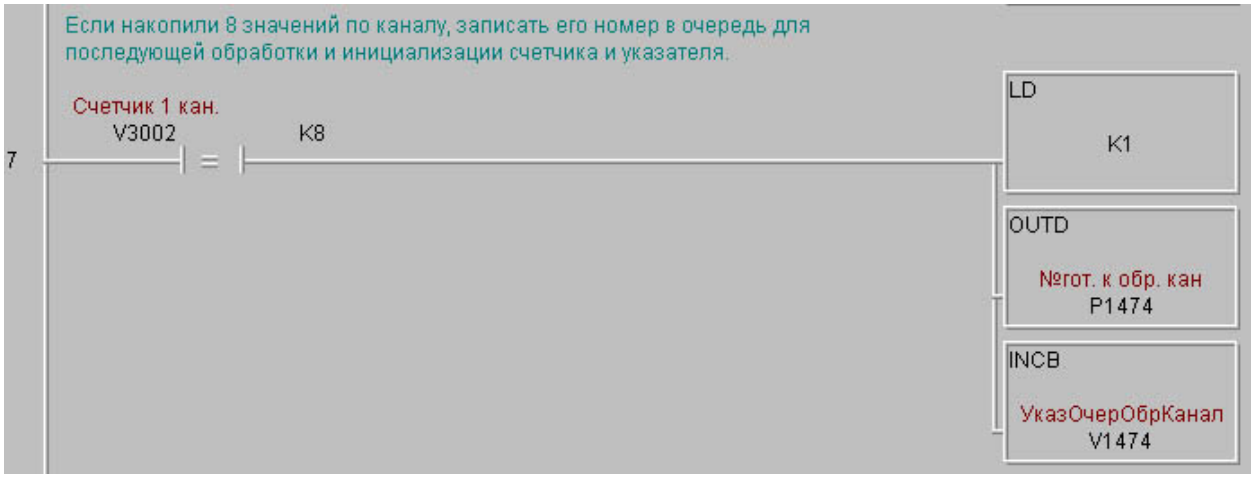
Если произвели 8 считываний – идем на обработку показаний, т.е. усреднение, выбраковку и пересчет из дискрет, например, в градусы (если измеряем температуру):



Если в составе контроллера имеется несколько модулей F2-04THM, то накопить 8 значений в буфере мы можем для нескольких каналов в одном цикле сканирования, что сделает невозможным использование ветви программы, приведенной выше. Выйти из этой ситуации можно, если разделить циклы измерения и циклы обработки, т.е пусть измерения идут своим чередом, а обрабатывать будем только те каналы, для которых произвели 8 считываний. Адреса переменной типа Канал считывания при этом заносятся с специальную очередь, а цикл обработки проверяет очередь и если обнаруживает в ней адрес переменной Канал считывания, то производит обработку. Очередь организуется тоже в Стадии 0:



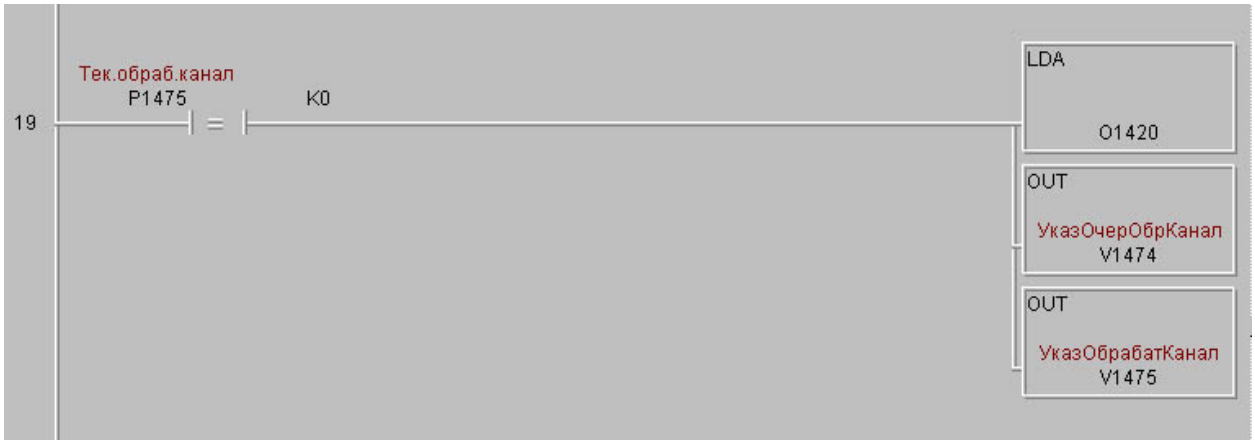
С учетом этого при накоплении 8 значений в буфере записываем адрес переменной Канал считывания в очередь:



Пусть просмотр очереди осуществляется в Стадии 12. Выбираем из очереди текущий элемент, который представляет собой номер канала, готового для обработки, сохраняем его в специальной ячейке, и очищаем этот элемент очереди:

Теперь легко получить адрес объекта Канал считывания:

Если в очереди нет отличного от нуля элемента, указатели устанавливаются на начало очереди:

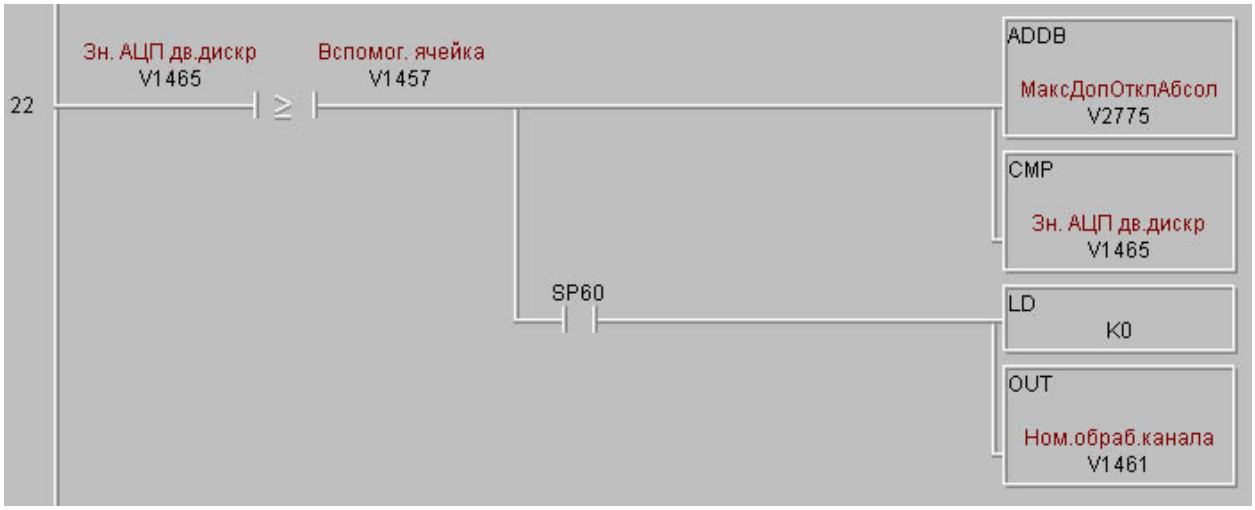


Сумма всех 8 считываний находится по адресу ($V_{\text{адрес массива переменных канала}}$+4). Используем ячейку V1460 в качестве вспомогательного указателя для расчета среднего значения цикла считывания, результат сохраним в V1465:

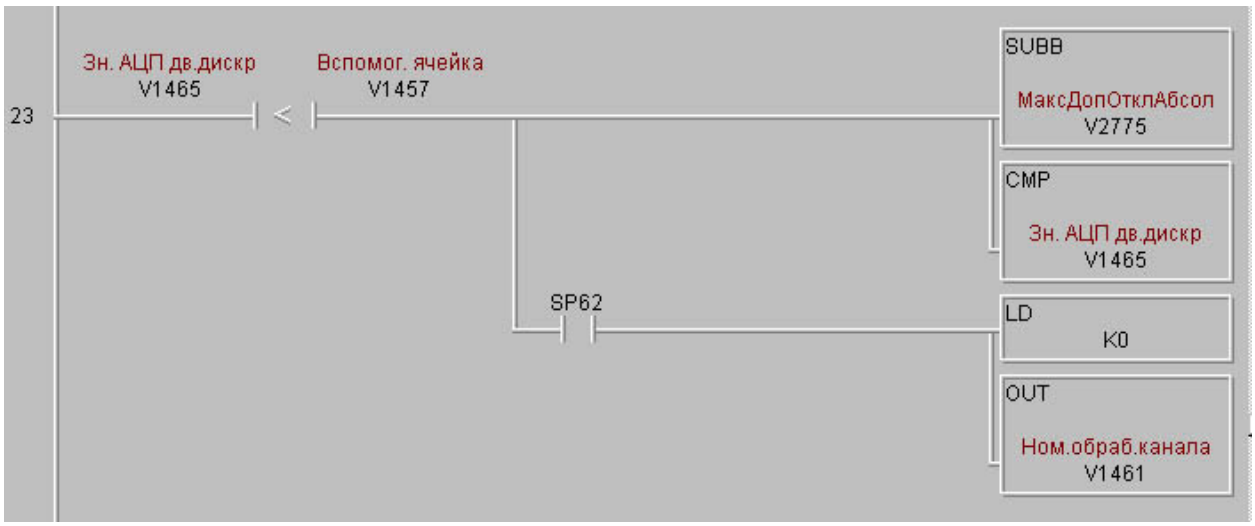
Итак, в ячейке V1465 мы получили усредненный результат измерений в течение 8 циклов. Как определить, насколько он правильный? По адресу ($V_{\text{адрес массива переменных канала}}$+3) будем хранить предыдущий результат усреднения, а в ячейке, например, v2775 – максимально (или минимально) допустимое значение, которое будем рассчитывать, исходя из здравого смысла: пусть для простоты расчетов текущее значение отличается от предыдущего не более чем на 50%, т.е. в V2775 будем записывать половину предыдущего:

Теперь, если текущий результат больше предыдущего, он не должен превышать суммы предыдущего значения и содержимого ячейки V2775, в которой находится величина максимально допустимой абсолютной разницы между текущим и предыдущим показаниями, иначе считаем текущее показание ошибкой и не обрабатываем, хотя сохраняем для сравнения в следующем цикле. Это необходимо для того, чтобы

зафиксировать скачок показаний, который может иметь место в случае обрыва, аварийной ситуации и т.п.:



Аналогично проверяем правильность результата, если он меньше предыдущего:

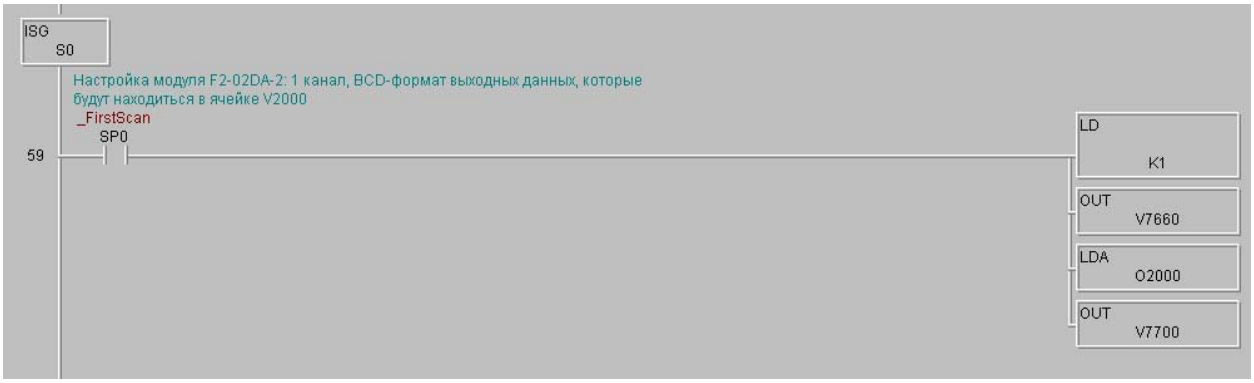


Теперь перед следующим циклом считывания необходимо проинициализировать переменные объекта Канал считывания:

В заключение этой главы – рекомендации по поводу пересчета дискрет в значение температуры. Пересчет не требуется, если модуль настроен на один из стандартных типов термопар, например, тип К. Если же необходимо пересчитать показания модуля в милливольты, а затем по таблице соответствующей термопары определить температуру, можно воспользоваться алгоритмом двоичного поиска: сравнивая напряжение с серединой интервала, в котором находится искомое значение температуры, находим более короткий интервал, и так до определения интервала напряжений, внутри которого характеристику термопары можно считать линейной (обычно это 1 градус), и на этом интервале проводим линеаризацию.

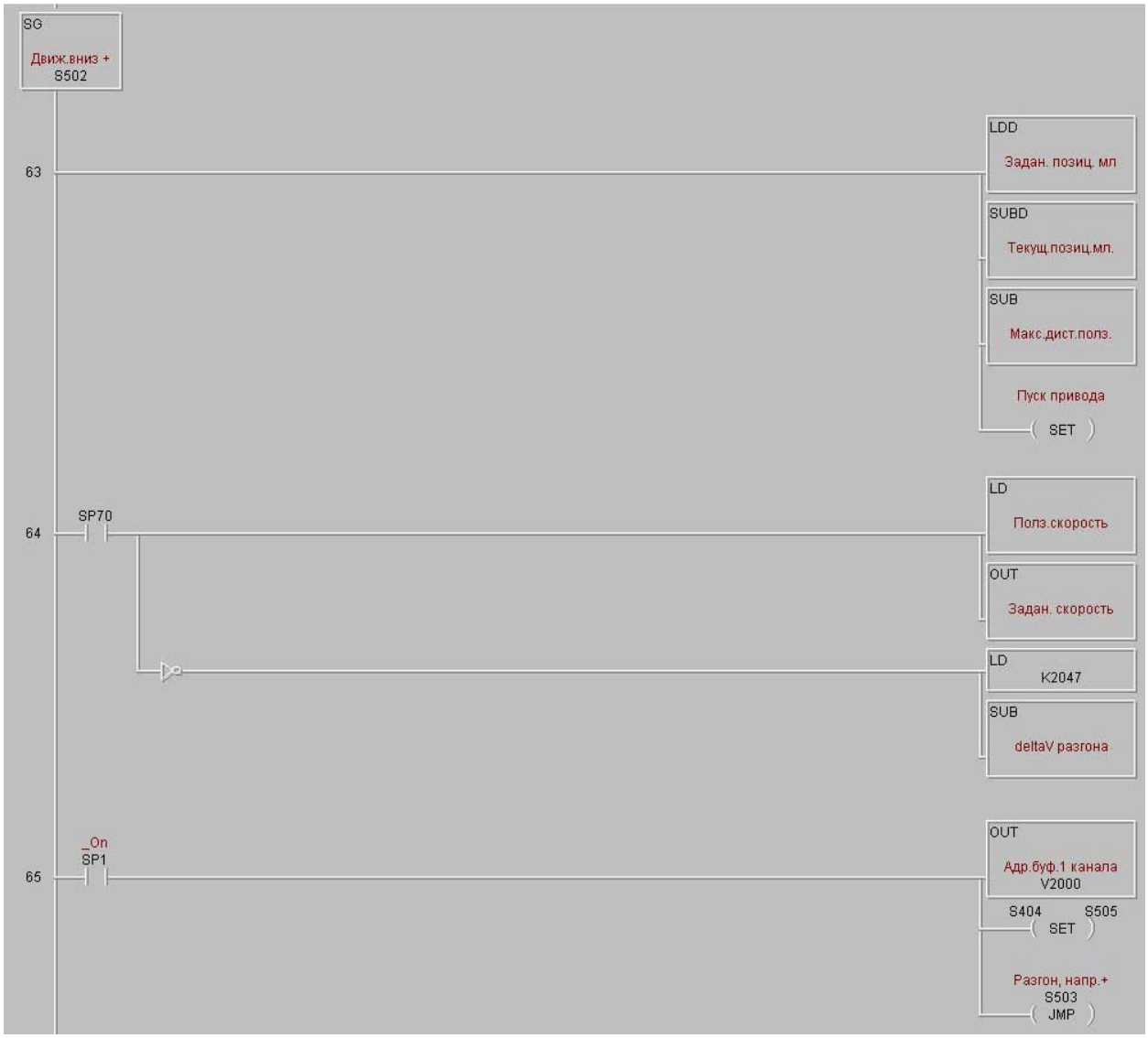
11. Алгоритм «Выход в точку с заданной скоростью».

Пусть в нашей системе управления имеется механизм, привод которого может перемещаться в положительном и отрицательном направлениях (вперед-назад, вверх-вниз). Для управления приводом используем модуль ЦАП «F2-02DA-2», который установлен в слоте 0. Для настройки модуля на выходное напряжение ±10В и формат выходных данных 0-4095 соответствующим образом устанавливаем перемычки на корпусе. Описание положения перемычек подробно приводится в документации. При работе с модулем воспользуемся методом указателя, поэтому для настройки модуля требуется написать команды, которые заносят в специальные ячейки процессора количество каналов и адреса V-памяти, в которые будем записывать выходные значения. Команды настройки, как уже не раз говорилось, помещаются в начальной стадии:

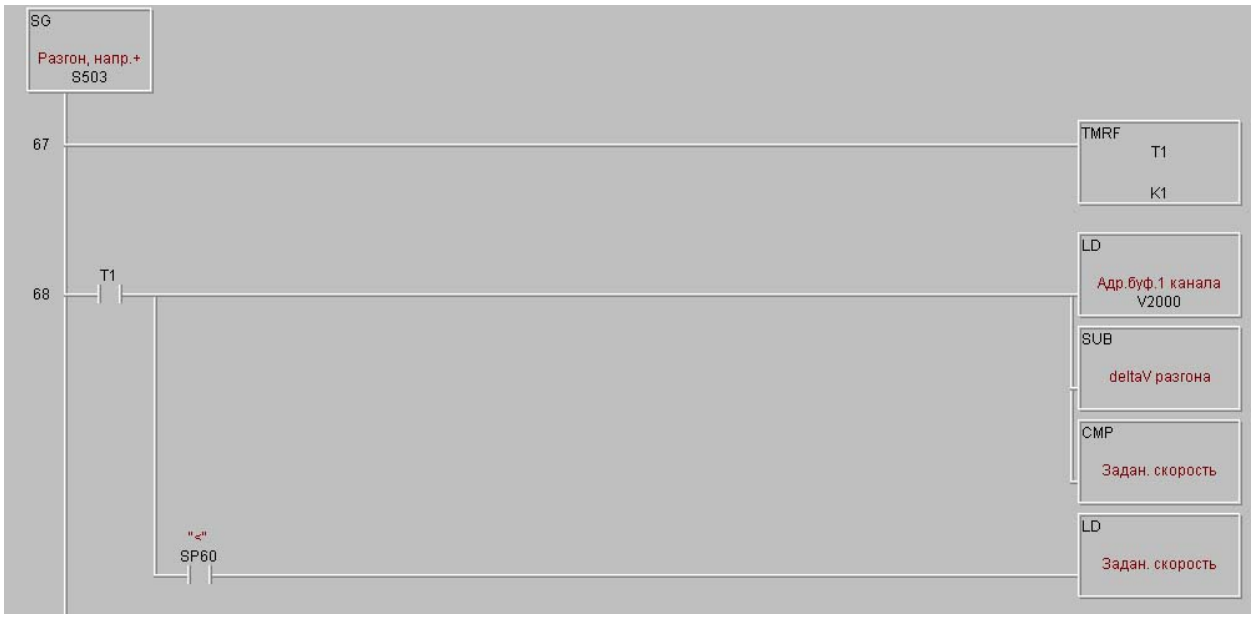


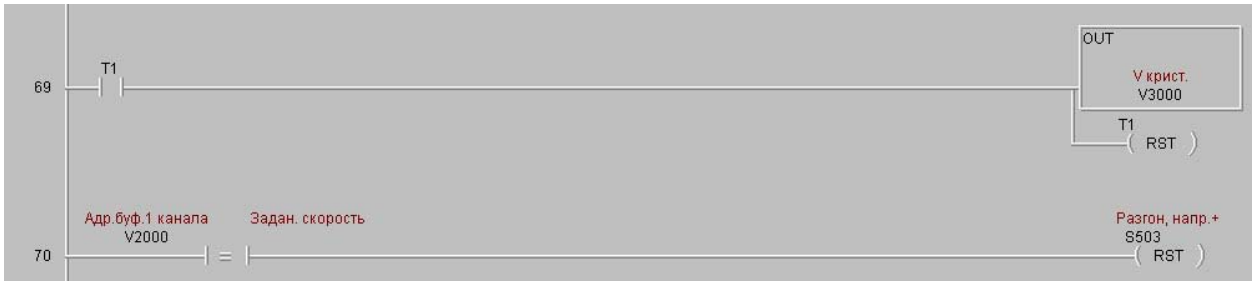
Чтобы на перегружать пример излишними подробностями, опустим некоторые детали, такие как получение текущей координаты привода и задание новой, пересчет ее в дискреты, а также обсуждение варьируемых величин – обычно их значение подбирается эмпирическим путем. Итак, сначала определяем, в какую сторону будет двигаться привод:

Теперь нужно разогнать привод до заданной скорости, а при приближении к заданной координате затормозить, т.е. выполнить традиционную траекторную задачу управления движением с участками ускорения и замедления. Сразу на ЦАП величину заданной скорости выставить из-за перегрузок нельзя, так же нельзя мгновенно остановить привод, поэтому подберем такое значение уставки ЦАП, при которой привод будет двигаться достаточно медленно, чтобы его можно было сразу остановить или сразу привести в движение без последствий для его конструкции. Назовем эту величину «ползучей скоростью». На этой скорости будем двигаться, если заданный путь меньше участков разгона и торможения. Переходим на стадию разгона, а также активируем стадии, которые осуществляют проверку, не достигли ли мы заданной координаты или же точки начала торможения:

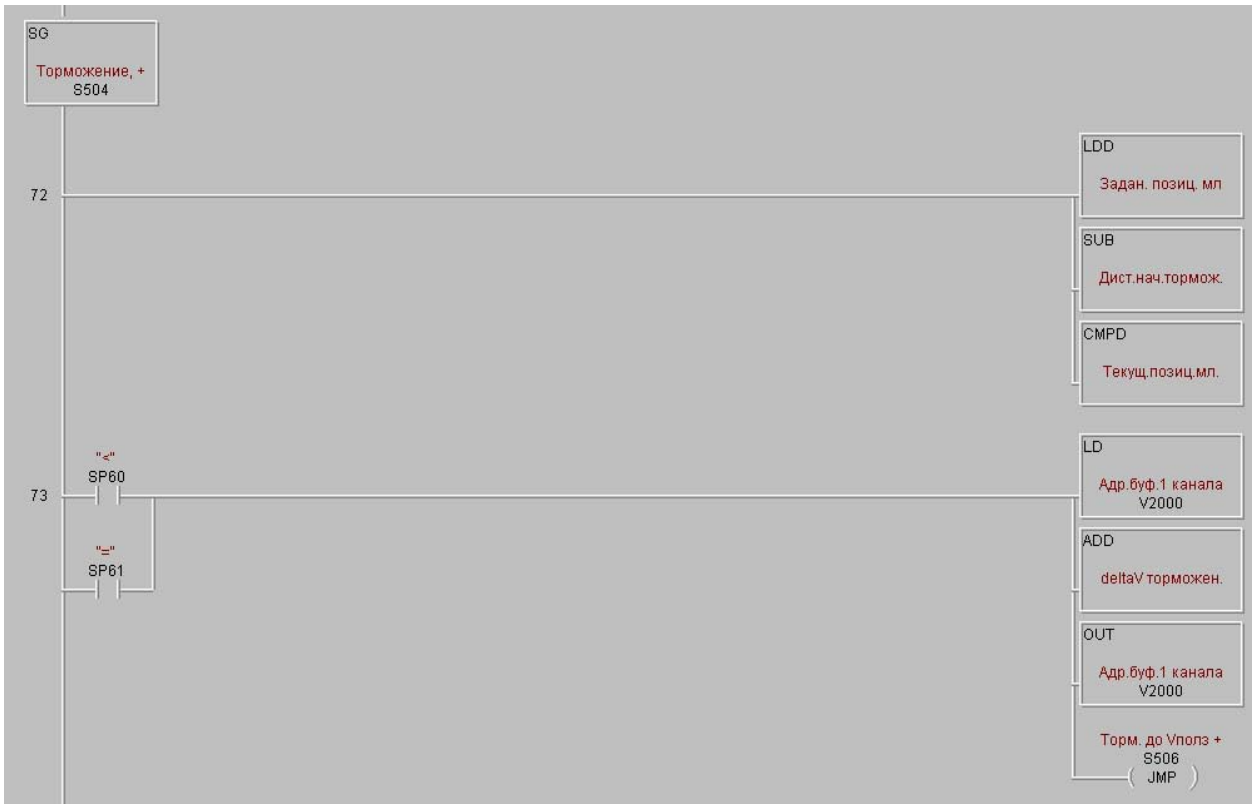


При разгоне каждые 10 мс увеличиваем значение уставки на ЦАП на величину ΔV разгона, пока не достигнем заданной скорости; тогда эта стадия деактивируется:

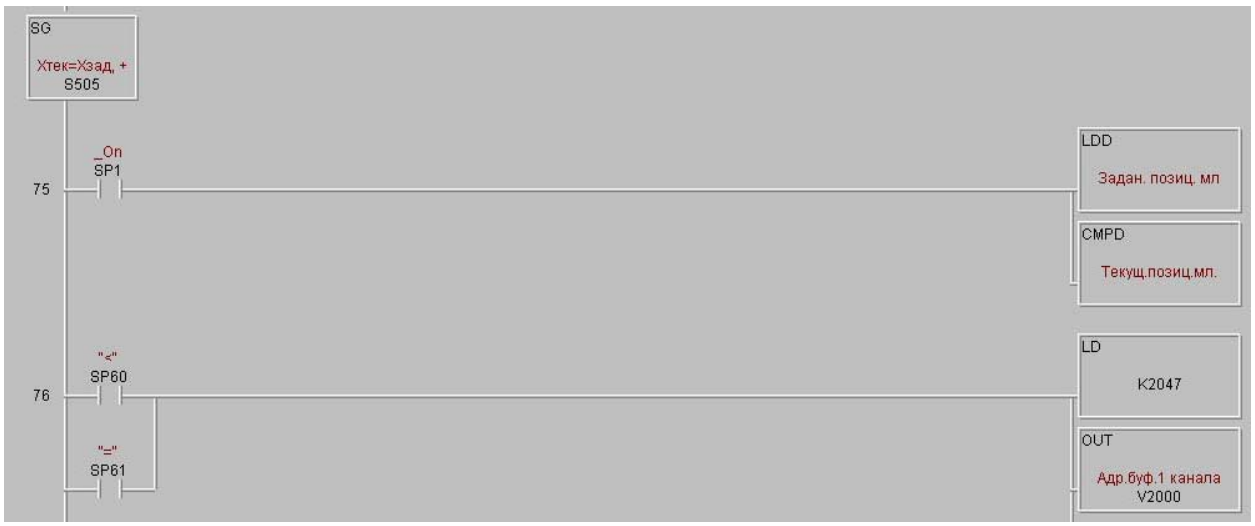




При движении определяем также, не находимся ли мы на участке торможения. В этом случае начинаем снижать скорость:



Если достигли заданной координаты, останавливаем движение:



Пока не достигли заданной точки, снижаем скорость до значения «ползучей», при которой привод можно безопасно остановить:

Алгоритм движения в противоположную сторону очень легко можно сделать из приведенного выше, внеся небольшие изменения, связанные с тем, что для увеличения скорости уставку на ЦАП нужно не уменьшать, а увеличивать.

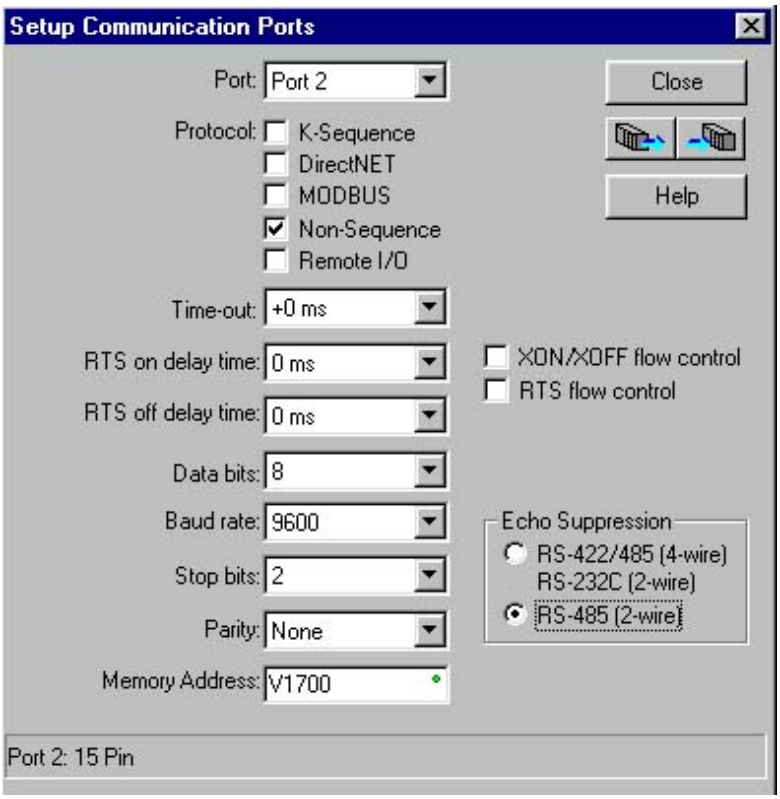
Заметим, что в приведенном примере стадии деактивируют сами себя. Мы не считаем это примером хорошего стиля программирования на RLL Plus, однако здесь имеем как раз тот случай, когда отказ от следования стилю даёт достаточно простое и эффективное решение поставленной задачи.

12. Обмен с периферийными устройствами

Долгое время контроллеры DirectLogic представляли собой замкнутую систему, способную обмениваться информацией только с аналогичными контроллерами и с «родственными» периферийными устройствами, например, с панелями оператора. Можно было также использовать коммуникационный порт 2 процессора для вывода на печать. С появлением процессора D2-260 стало возможным осуществлять через порт 2 обмен

информацией с любыми периферийными устройствами по известному протоколу. Правда, в описании команд, отвечающих за посылку и прием байтов, оговаривается, что эти команды не могут применяться в одном приложении, т.е. предлагается в зависимости от поставленной задачи либо только посылать в порт строку, либо только читать из порта (в качестве примера приводятся электронные весы). На самом деле это не так. Фирма-изготовитель контроллера предлагает использовать только либо запись в порт, либо чтение из порта потому, что отсутствует аппаратная синхронизация процедур чтения и записи. Тем не менее организовать обмен реально, если удастся реализовать синхронизацию программно. В этой главе мы покажем, как это делается. Но сразу сделаем оговорку: пример с обменом был реализован на одном конкретном устройстве. Для любого другого устройства необходим индивидуальный подход. Особенность работы того или иного устройства в режиме обмена выясняется по мере написания нескольких тестовых программ. Основная же задача этой главы - предложить методологию, которая поможет справиться с задачей обмена быстрее и поможет правильному пониманию работы с портом 2.

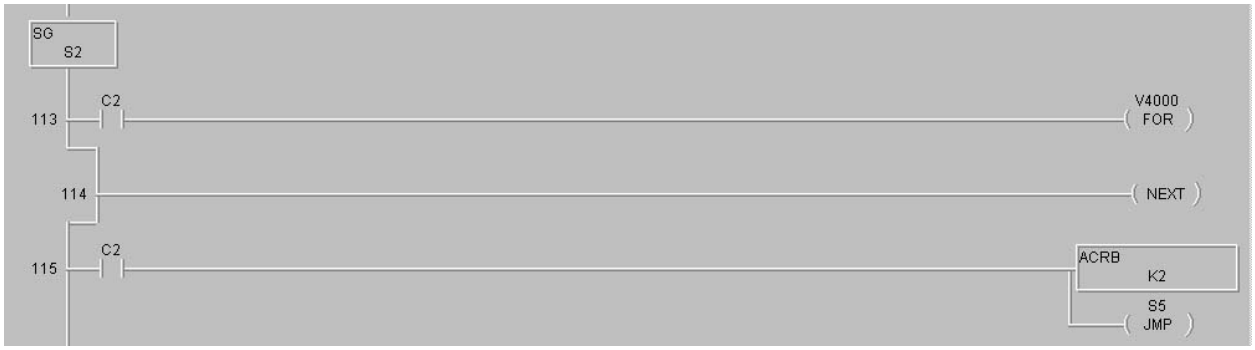
Прежде всего нужно правильно сконфигурировать порт 2. Делается это в среде Direct Soft: меню PLC->Setup->Setup Sec. Comm port:



Значения таймаута, количество служебных битов и байтов данных, а также тип интерфейса устанавливаются согласно техническим характеристикам устройства.

Предположим, что обмен осуществляется 1 раз в секунду. С отправкой информации в порт проблем нет: строка посылки формируется в V-памяти, для анализа результата посылки резервируются два С-бита: бит «порт занят» (Busy) и бит «выполнено» (Complete), также требуется задать количество посылаемых байтов; при необходимости можно добавить заключительные символы типа «Конец строки» и «Возврат каретки» и переставить байты при посылке их в порт:

Отправка не происходит мгновенно; команда PRINTV лишь инициирует этот процесс. О завершении его можно судить по факту установления бита “Complete” (в нашем примере – бита C2). Состояние C2 проверяем в следующей стадии и, если он установился, идем на стадию приема байтов из порта. Нелишне будет перед этим очистить буфер приема, хотя большой необходимости в этом нет:



Поясним назначение этой стадии. Мы не знаем, сколько времени нужно для установления бита C2 – может, несколько миллисекунд или несколько десятков миллисекунд? У устройства, с которым мы хотим наладить обмен, есть свои временные параметры для отправки ответного сообщения, и если мы опоздаем с анализом бита C2 и не начнем вовремя принимать байты из порта, ответная посылка просто-напросто потеряется. А такое вполне может произойти, если время цикла сканирования относительно велико – скажем, до 10 миллисекунд. С другой стороны, факт установки бита C2 означает лишь то, что посылка отправлена, и нужно подождать какое-то время, пока наше устройство примет информацию, обработает ее и сформирует ответную посылку. В этом случае нам на помощь придет цикл FOR, благодаря которому мы незначительно увеличим цикл сканирования, но зато, правильно подобрав параметр цикла (значение в ячейке V4000), начнем прием сообщения примерно в то время, когда оно и должно появиться на входе порта 2, т.е. фактически эта стадия и отвечает за синхронизацию. Абсолютная точность, к счастью, здесь не требуется: команда приема позволяет задать время таймаута на прием первого символа, и нам лишь нужно, чтобы прием начался до того, как это время истечет.

Для приема строки из порта 2 нужно задать буфер для принимаемых символов, длину посылки (если длина произвольна, то ввести максимально возможную, но не более 128 байт), определить C-биты «занято» и «выполнено». Крайне важно задать время таймаута для первого символа - как сказано выше, благодаря этому параметру в сочетании с параметром цикла FOR и удается синхронизировать прием и отправку сообщений. Можно также задать таймаут между символами. Если задан таймаут, то резервируется C-бит, который будет установлен в случае, когда время таймаута истекло:

Далее мы ждем установки одного из битов и только тогда уходим из этой стадии и производим соответствующие действия. Так, если установился бит C4 ("Complete"), анализируем принятое сообщение; если установлен один из битов таймаута - сообщаем об ошибке и т.п. Биты завершения посылки и результата приема должны быть сброшены.

Вообще природа команды AIN довольно сложна, и даже в ее описании почти ничего не говорится о том, как она работает. Выводы, которые приведены ниже, сделаны исключительно опытным путем, но, повторимся, получены они в результате обмена с одним определенным устройством, а в случае использования другого устройства команда AIN может вести себя по-другому. Выяснить, как правильно ее применить, можно только экспериментально.

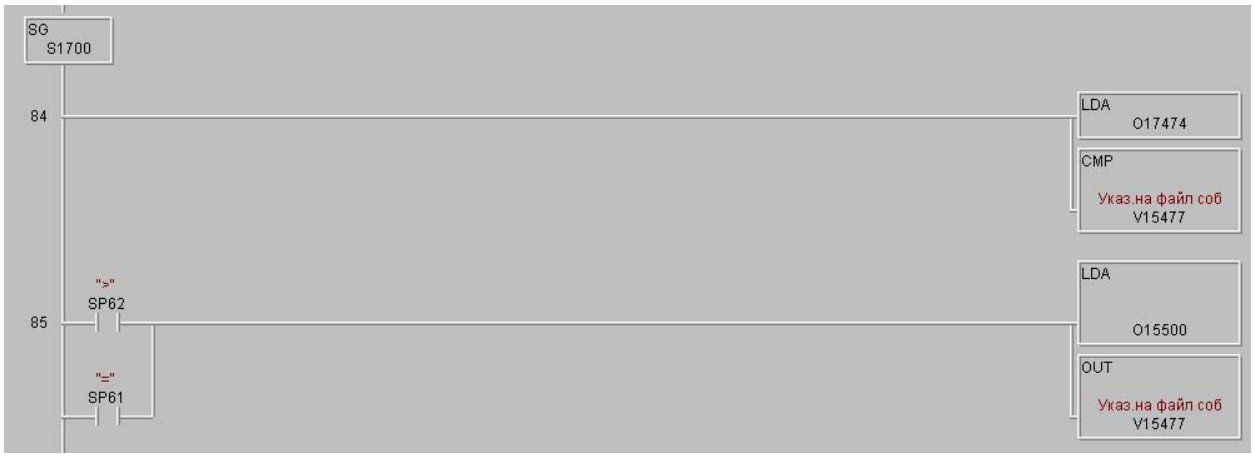
Итак, стадия, в которой используется команда приема, и сама команда приема должны быть активны! Когда команда AIN активируется впервые, инициируется процесс приема и одновременно запускаются ее внутренние таймеры, отсчитывающие таймауты (если приема нет). В последующих циклах сканирования команда AIN фактически не работает, прием идет аппаратно, а активность AIN означает лишь то, что мы как бы "даем понять" процессору, что процедура приема нас все еще интересует - до тех пор, пока не установился один из битов ("Complete" или "Timeout"). Желательно, чтобы во время приема цикл сканирования был как можно короче. Для этого, если программа довольно велика по объему и времени выполнения, возможно, придется на время обмена отключать все или почти все остальные стадии.

Для работы с полученным сообщением в систему команд процессора D2-260 добавлены новые команды выделения подстроки AEX, выделения подстроки AFIND, сравнения строк CMPV и создания строки в V-памяти VPRINT.

13. Создание «файла событий».

Объем V-памяти контроллера DirectLogic достаточно велик. Поэтому свободные ячейки памяти можно использовать для создания так называемого «файла событий» – т.е. фиксации всех событий в СУ (изменение состояния дискретных входов, флагов или возникновение ошибки с указанием времени события). Эти данные можно затем проанализировать, распечатав на принтере или передав на верхний уровень управления. Задействуем верхние адреса V-памяти – с V15500 по V17474. Когда количество событий превысит размер отведенной памяти, запись снова начнется с начала массива. Массив (т.е. указатель на него), как обычно, сформируем в начальной стадии:

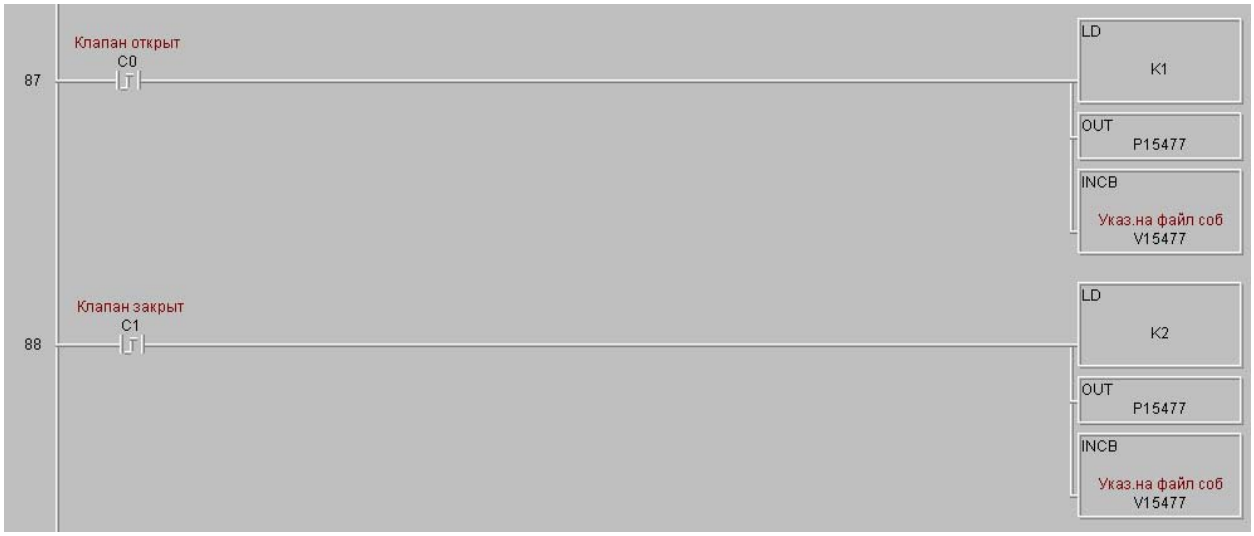
Теперь создадим стадию, которая и будет формировать файл событий. Перед тем, как записать событие, проверим, не вышли ли мы за пределы нашего массива:



Запишем в массив текущее время из ячеек V7766(секунды), V7767(минуты), V7770(часы):

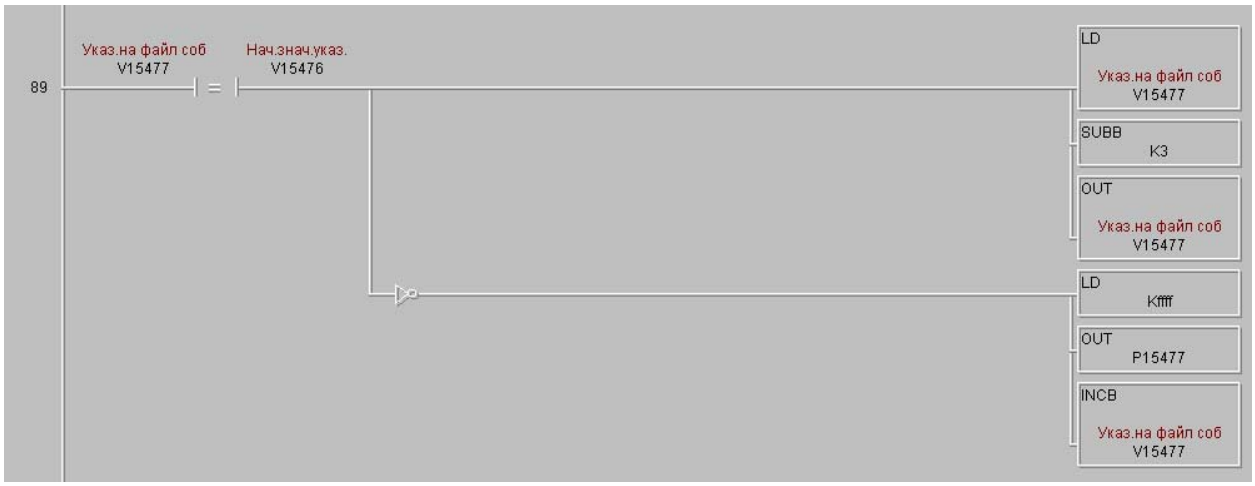
Сохраним текущее значение указателя массива:

Факт события будем записывать в массив таким образом:



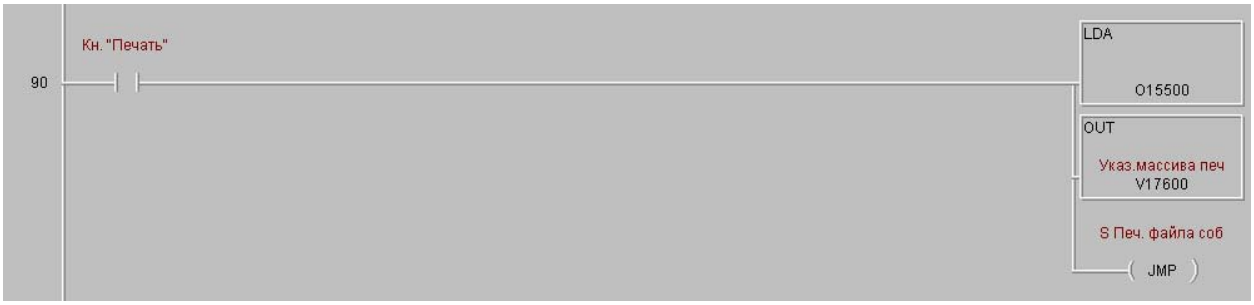
Чуть короче это выглядело бы, если запись в массив и модификацию указателя организовать в виде подпрограммы, однако это будет иметь эффект, только если в нашей системе может произойти не более 128 разных событий.

Если в текущем цикле сканирования событий не произошло, возвращаем указатель на место для записи времени в новом цикле, если же события были – запишем в массив число FFFF в виде признака окончания списка событий в данном цикле:



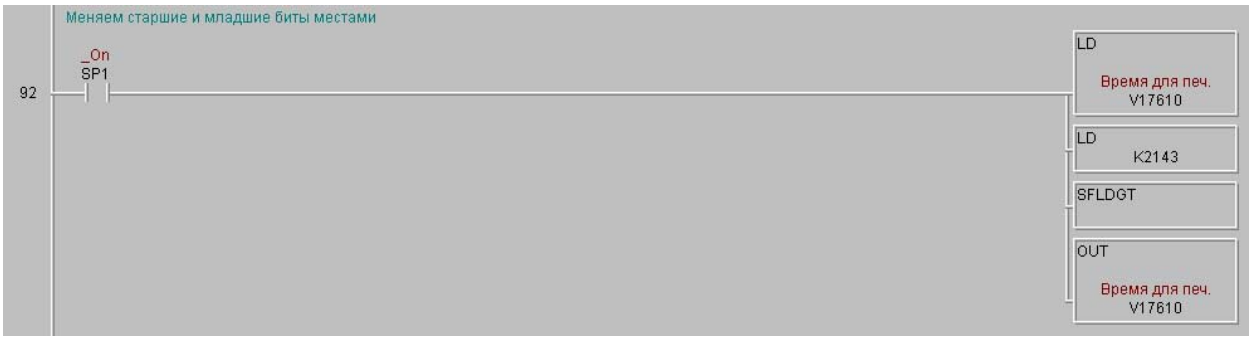
При желании можно этот алгоритм сделать более совершенным: например, ввести проверку изменения времени, чтобы при печати не повторялось одно и то же время. Произвести быстрое сравнение таблиц позволяет система команд процессора D2-260.

Теперь покажем, как файл событий распечатать на принтере. Принтер должен иметь последовательный порт, порт 2 контроллера необходимо сконфигурировать, как указано в документации (см. описание команды PRINT). Пример будет интересен еще и тем, что в нем используются команды, которые специфичны по своему действию и потому применяются относительно редко. Итак, пусть распечатка файла событий производится по нажатию кнопки. Если никаких событий в системе к моменту начала распечатки не произошло, распечатается только текущее время. Для печати элементов массива событий создадим новый указатель на этот массив:



Значения времени, которые занимают 3 ячейки V-памяти, нужно преобразовать в ASCII-коды. Для этого в языке RLL Plus имеется специальная команда HTA:

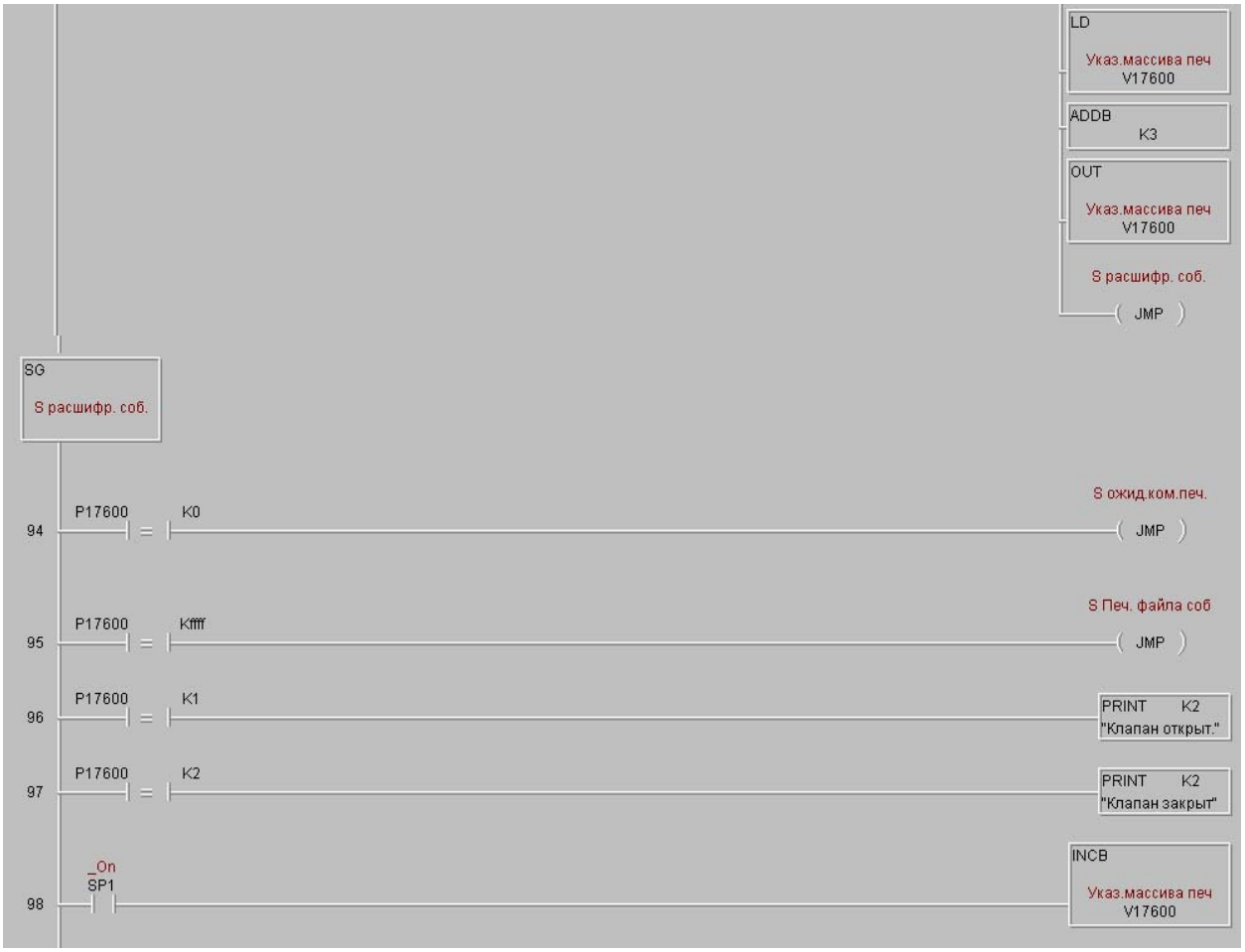
Теперь в результате выполнения этой команды мы имеем массив из 6 ячеек, начинающийся с адреса 17610, в котором находится время в ASCII-коде. Но распечатывать время пока что рано – необходимо выполнить еще одно преобразование. Дело в том, что печать производится младшими байтами вперед, а в результате преобразования числа в ASCII-код старшие разряды времени оказались, соответственно, в старшем байте, хотя на печать должны выводиться раньше. Итак, перед нами стоит задача переставить местами байты в 3 ячейках: V17610, V17612, V17614. В ячейках с адресами V17611, V17613, V17615 находятся ASCII-коды нулей, которые не нужно ни преобразовывать, ни печатать вообще. К сожалению, в системе команд процессора D2-250 отсутствует команда перестановки байтов, поэтому воспользуемся тем, что есть, а именно командой перестановки знаков SHFLDGT. Для того, чтобы поменять местами старший и младший байты, порядок перестановки должен быть 2143 (разряды 16-31 не трогаем):



То же самое делаем с двумя другими ячейками. Можно было бы, конечно, организовать цикл FOR, но в данном случае сэкономить на командах не получилось бы: пришлось бы заводить указатели и модифицировать их. Мы сделали проще и понятнее. А процессор DL-260 выполняет это преобразование одной командой:

Теперь печатаем время (лучше делать это в отдельной стадии):

Формат команды PRINT позволяет выводить содержимое ячеек V-памяти напрямую, но если бы мы поступили так, то получили бы по два нуля перед значениями часов, минут и секунд. Теперь распечатаем события, которые произошли в этот момент времени. Если весь список событий распечатан, возвращаемся на ожидание команды печати, а если изменилось время, возвратимся на стадию преобразования времени:



В заключение заметим, как обычно, что совершенству предела нет, и предложенный алгоритм можно при желании улучшить.

14. О том, чего нет в этой книге.

Как уже не раз подчеркивалось, эта книга написана на основе нашего опыта программирования на RLL Plus, и так получилось, что на практике применялись не все

команды и возможности языка RLL Plus, а также не все разнообразие модулей, которые могут работать в составе системы управления на базе контроллеров DirectLogic. В частности, не было повода использовать в наших программах счетчики, барабанный командоаппарат, модули прерываний и, соответственно, процедуры их обслуживания, сдвиговые регистры и еще ряд команд и элементов языка. С одной стороны, это говорит о неполноте материала, представленного в нашей работе, но с другой – это отображение того факта, что не все богатые возможности языка RLL Plus обязательно должны иметь применение в системном программном обеспечении, что качественная и надежная программа управления технологической установкой обычно состоит из наиболее общеупотребительных и простых команд. Решение о применении той или иной команды принимается программистом в зависимости от конкретного назначения оборудования, которое ему предстоит автоматизировать, и мы не сомневаемся, что на многих предприятиях, выпускающих автоматизированное технологическое оборудование под управлением контроллеров DirectLogic, активно используются счетчики, преобразователи Грэй-кода и 7-сегментые преобразователи, барабанные командоаппараты, тригонометрические расчеты и операции со строками. Мы не ставили перед собой цель создать учебник языка с примерами, а просто хотели поделиться опытом программирования. Описания «экзотических» команд с примерами имеются в Руководстве пользователя, и на их основе профессиональный программист легко разберется в том, как работает та или иная команда, чтобы использовать ее в своей программе.

Кроме того, мы не описали работу с периферийными устройствами ввода-вывода, которые можно подключить к контроллеру через верхний порт. Имеются в виду пульты оператора и панели отображения. Дело в том, что для настройки этих устройства, а также для модуля H2-CTRIO существует специальное программное обеспечение, вследствие чего для работы с ними не требуется практически никакого программирования – обычно достаточно определить ячейки V-памяти, в которые будут вводиться данные или содержимое которых будет индицироваться.

Поскольку, как было сказано в начале книги, ее особенностью является незаконченность, мы будем признательны нашим читателям, которые предложат новые темы, незатронутые нами, новые интересные алгоритмы, концепции программирования, а также сделают свои замечания по содержанию книги.