Алгоритмическое и программное обеспечение задач автоматизации управления

Краткий курс лекций

Лекция 1

1. ПРОГРАММИРУЕМЫЕ КОНТРОЛЛЕРЫ

1.1 Определение ПЛК

Любая машина, способная автоматически выполнять некоторые операции, имеет в своем составе управляющий контроллер – модуль, обеспечивающий логику работы устройства. Контроллер – это мозг машины. Естественно, чем сложнее логика работы машины, тем «умнее» должен быть контроллер.

Контроллеры, выполненные на основе реле или микросхем с «жесткой» логикой, невозможно научить делать другую работу без существенной переделки. Очевидно, что такой возможностью обладают только *программируемые логические контроллеры* (ПЛК) [1,2,5].

Физически, типичный ПЛК представляет собой блок, имеющий определенный набор выходов и входов, для подключения датчиков и исполнительных механизмов (рис. 1.1). Логика управления описывается программно на основе микрокомпьютерного ядра. Абсолютно одинаковые ПЛК могут выполнять совершенно разные функции. Причем для изменения алгоритма работы не требуется каких-либо переделок аппаратной части. Аппаратная реализация входов и выходов ПЛК ориентирована на сопряжение с унифицированными приборами и мало подвержена изменениям [1,2,5].

Рис.1.1 – Принцип работы ПЛК.

Задачей прикладного программирования ПЛК является только реализация алгоритма управления конкретной машиной. Опрос входов и выходов контроллер осуществляет автоматически, внезависимости от способа физического соединения. Эту работу выполняет системное программное обеспечение. В идеальном случае прикладной программист совершенно не интересуется, как подсоединены и где расположены датчики и исполнительные механизмы. Мало того, его работа не зависит от того, с каким контроллером и какой фирмы он работает. Благодаря стандартизации языков программирования прикладная программа оказывается переносимой. Это означает, что ее можно использовать в любом ПЛК, поддерживающем данный стандарт [1,2,5].

Программируемый контроллер – это программно управляемый дискретный автомат, имеющий некоторое множество входов, подключенных посредством датчиков к объекту управления, и множество выходов, подключенных к исполнительным устройствам. ПЛК контролирует состояния входов и вырабатывает определенные последовательности программно заданных действий, отражающихся в изменении выходов (рис.1.2) [1,2,5].

Рис.2.1 – Внешний вид программируемого логического контроллера фирмы ОВЕН.

ПЛК предназначен для работы в режиме реального времени в условиях промышленной среды и должен быть доступен для программирования неспециалистом в области информатики.

1.2 Входы-выходы ПЛК

В ПЛК используются как бинарные, так и аналоговые входы и выходы [1,2,5].

Бинарный выход имеет два состояния – включен и выключен. Сфера применения бинарных выходов очевидна: электромагнитные реле, силовые пускатели, электромагнитные клапаны, световые сигнализаторы и т. д. ,

В современных ПЛК широко используются аналоговые входы и выходы. *Аналоговый* или *непрерывный сигнал* отражает уровень напряжения или тока, соответствующий некоторой физической величине в каждый момент времени. Этот уровень может относиться к температуре, давлению, весу, положению, скорости, частоте и т.д; то есть к любой физической величине.

Аналоговые входы контроллеров могут иметь различные параметры и возможности. Так, к их параметрам относятся: разрядность АЦП, диапазон входного сигнала, время и метод преобразования, несимметричный или дифференциальный вход, уровень шума и нелинейность, возможность автоматической калибровки, программная или аппаратная регулировка коэффициента усиления, фильтрация. Особые классы аналоговых входов представляют входы, предназначенные для подключения термометров сопротивления и термопар. Здесь требуется применение специальной аппаратной поддержки (трехточечное включение, источники образцового тока, схемы компенсации холодного спая, схемы линеаризации и т.д.) [5].

В сфере применения ПЛК бинарные входы и выходы называют обычно дискретными. Аналоговые сигналы в ПЛК обязательно преобразуются в цифровую, т.е. заведомо дискретную форму представления. Но в технических документах ПЛК любой фирмы указывается количество дискретных и аналоговых входов [3-5].

Помимо дискретных и аналоговых входов-выходов многие ПЛК имеют *специализированные входы-выходы*. Они ориентированы на работу с конкретными специфическими датчиками, требующими определенных уровней сигналов, питания и

специальной обработки. Например, квадратурные шифраторы, блоки управления шаговыми двигателями, интерфейсы дисплейных модулей и т.д.

Входы-выходы ПЛК не обязательно должны быть физически сосредоточены в общем корпусе с процессорным ядром. В последние годы все большую популярность приобретают технические решения, позволяющие полностью отказаться от прокладки кабелей для аналоговых цепей. Входы-выходы выполняются в виде миниатюрных модулей, расположенных в непосредственной близости от датчиков и исполнительных механизмов. Соединение подсистемы ввода-вывода с ПЛК выполняется посредством одного общего цифрового кабеля. Например, в интерфейсе Actuators/Sensors interface применяется плоский профилированный кабель («желтый кабель») для передачи данных и питания всего по двум проводам [5-9].

1.3 Режим реального времени и ограничения на применение ПЛК

Для математических систем характеристикой качества работы является правильность найденного решения. В системах реального времени помимо правильности решения определяющую роль играет время реакции. Логически верное решение» полученное с задержкой более допустимой, не является приемлемым.

Принято различать системы жесткого и мягкого реального времени. В системах жесткого реального времени существует выраженный временной порог. При его превышении наступают необратимые катастрофические последствия. В системах мягкого реального времени характеристики системы ухудшаются с увеличением времени управляющей реакции. Система может работать плохо или еще хуже, но ничего катастрофического при этом не происходит [3-5].

Классический подход для задач жесткого реального времени требует построения событийно управляемой системы. Для каждого события в системе устанавливается четко определенное время реакции и определенный приоритет. Практическая реализация таких систем сложна и всегда требует проработки и моделирования.

Для ПЛК существенное значение имеет не только быстродействие самой системы, но и время проектирования, внедрения и возможной оперативной переналадки.

Абсолютное большинство ПЛК работают по методу периодического опроса входных данных (сканирования). ПЛК опрашивает входы, выполняет пользовательскую программу и устанавливает необходимые значения выходов. Специфика применения ПЛК обусловливает необходимость одновременного решения нескольких задач. Прикладная программа может быть реализована в виде множества логически независимых задач, которые должны работать одновременно [1,2,5].

На самом деле ПЛК имеет обычно один процессор и выполняет несколько задач псевдопараллельно, последовательными порциями. Время реакции на событие оказывается зависящим от числа одновременно обрабатываемых событий. Рассчитать минимальное и максимальное значения времени реакции, конечно, можно, но добавление новых задач или увеличение объема программы приведет к увеличению времени реакции. Такая модель более подходит для систем мягкого реального времени. Современные ПЛК имеют типовое значение времени рабочего цикла, измеряемое единицами миллисекунд и менее. Поскольку время реакции большинства исполнительных устройств значительно выше, с реальными ограничениями возможности использования ПЛК по времени приходится сталкиваться редко [5].

В некоторых случаях ограничением служит не время реакции на событие, а обязательность его фиксации, например работа с датчиками, формирующими импульсы

малой длительности. Это ограничение преодолевается специальной конструкцией входов.

Так, счетный вход позволяет фиксировать и подсчитывать импульсы с периодом во много раз меньшим времени рабочего цикла ПЛК. Специализированные интеллектуальные модули в составе ПЛК позволяют автономно отрабатывать заданные функции, например модули управления сервоприводом [1,2,5].

1.4 Интеграция ПЛК в систему управления предприятием

Контроллеры традиционно работают в нижнем звене автоматизированных систем управления предприятием (АСУ) – систем, непосредственно связанных с технологией производства (ТП). ПЛК обычно являются первым шагом при построении систем АСУ. Это объясняется тем, что необходимость автоматизации отдельного механизма или установки всегда наиболее очевидна. Она дает быстрый экономический эффект, улучшает качество производства, позволяет избежать физически тяжелой и рутинной работы. Контроллеры по определению созданы именно для такой работы [5].

Далеко не всегда удается создать полностью автоматическую систему. Часто «общее руководство» со стороны квалифицированного человека – диспетчера необходимо. В отличие от автоматических систем управления такие системы называют автоматизированными.

В настоящее время подобные пульты применяются только в очень простых случаях, когда можно обойтись несколькими кнопками и индикаторами. В более «серьезных» системах применяются ПК [5].

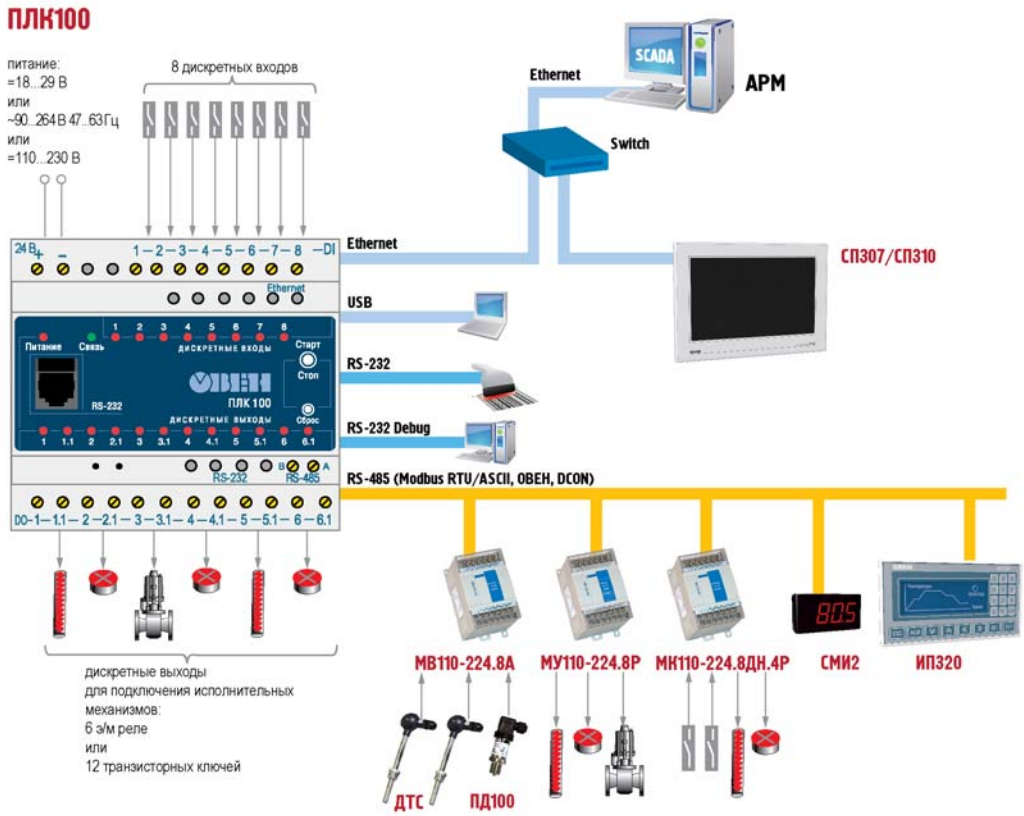

Рис.1.3 – Место ПЛК в АСУ ТП

Появился целый класс программного обеспечения реализующего интерфейс человек-машина (MMI). Это так называемые системы сбора данных и оперативного диспетчерского управления (Supervisory Control And Data Acquision System – SCADA). Современные SCADA-системы выполняются с обязательным применением средств мультимедиа. Помимо живого отображения процесса производства, хорошие диспетчерские системы позволяют накапливать полученные данные, проводят их хранение и анализ, определяют критические ситуации и производят оповещение персонала по каналам телефонной и радиосети, позволяют создавать сценарии управления (как правило, Visual Basic), формируют данные для анализа экономических характеристик производства [5-9].

Разделение производства ПЛК, средств программирования и диспетчерских систем привело к появлению стандартных протоколов обмена данными. Наибольшую, известность получила технология OPC (OLE for Process Control), базирующаяся на механизме DCOM Microsoft Windows. Механизм динамического обмена данными (DDE) применяется пока еще достаточно широко, несмотря на то что требованиям систем реального времени не удовлетворяет.

Средства системной интеграции являются составной частью базового программного обеспечения современного ПЛК (рис.1.3) [3-5].

Второй часто возникающей задачей является интеграция нескольких ПЛК с целью синхронизации их работы. Здесь появляются сети, обладающие рядом специфических требований. В целом это требования, аналогичные требованиям к ПЛК: режим реального времени, надежность в условиях промышленной среды, ремонтопригодность, простота программирования. Такой класс сетей получил название промышленных сетей (fieldbus). Существует масса фирменных реализаций и достаточно много стандартов таких сетей (Bitbus, Modbus, Profibus, CANopen, DeviceNet), позволяющих интегрировать аппаратуру различных фирм, но ни один из них нельзя признать доминирующим [1,2,5].

1.5 Рабочий цикл ПЛК

Задачи управления требуют непрерывного циклического контроля. В любых цифровых устройствах непрерывность достигается за счет применения дискретных алгоритмов, повторяющихся через достаточно малые промежутки времени. Таким образом, вычисления в ПЛК всегда повторяются циклически, Одна итерация, включающая замер, обсчет и выработку воздействия, называется рабочим циклом ПЛК. Выполняемые действия зависят от значения входов контроллера, предыдущего состояния и определяются пользовательской программой.

По включению питания ПЛК выполняет самотестирование и настройку аппаратных ресурсов, очистку оперативной памяти данных (ОЗУ), контроль целостности прикладной программы пользователя. Если прикладная программа сохранена в памяти, ПЛК переходит к основной работе, которая состоит из постоянного повторения последовательности действий, входящих в рабочий цикл [1,2,5].

Рабочий цикл ПЛК состоит из нескольких фаз.

1. начало цикла;
2. чтение состояния входов;
3. выполнение кода программы пользователя;
4. запись состояния выходов;
5. обслуживание аппаратных ресурсов ПЛК;

6. монитор системы исполнения;
7. контроль времени цикла;
8. переход на начало цикла.

В самом начале цикла ПЛК производит физическое чтение входов. Считанные значения размещаются в области памяти входов. Таким образом, создается полная одномоментная зеркальная копия значений входов.

Далее выполняется код пользовательской программы. Пользовательская программа работает с копией значений входов и выходов, размещенной в оперативной памяти. Если прикладная программа не загружена или остановлена, то данная фаза рабочего цикла, естественно, не выполняется. Отладчик системы программирования имеет доступ к образу входов-выходов, что позволяет управлять выходами вручную и проводить исследования работы датчиков.

После выполнения пользовательского кода физические выходы ПЛК приводятся в соответствие с расчетными значениями (фаза 4) [3-5].

Обслуживание аппаратных ресурсов подразумевает обеспечение работы системных таймеров, часов реального времени, оперативное самотестирование, индикацию состояния и другие аппаратно-зависимые задачи.

Монитор системы исполнения включает большое число функций, необходимых при отладке программы и обеспечении взаимодействия с системой программирования, сервером данных и сетью. В функции системы исполнения обычно включается: загрузка кода программы в оперативную и электрически перепрограммируемую память, управление последовательностью выполнения задач, отображение процесса выполнения программ, пошаговое выполнение, обеспечение просмотра и редактирования значений переменных, фиксация и трассировка значений переменных, контроль времени цикла и т. д.

Пользовательская программа работает только с мгновенной копией входов. Таким образом, значения входов в процессе выполнения пользовательской программы не изменяются в пределах одного рабочего цикла. Это фундаментальный принцип построения ПЛК сканирующего типа. Такой подход исключает неоднозначность алгоритма обработки данных в различных его ветвях [5-9].

Кроме того, чтение копии значения входа из ОЗУ выполняется значительно быстрее, чем прямое чтение входа. Аппаратно чтение входа может быть связано с формированием определенных временных интервалов, передачей последовательности команд для конкретной микросхемы или даже запросом по сети [5].

Если заглянуть глубже, то нужно отметить, что не всегда работа по чтению входов полностью локализована в фазе чтения входов. Например, АЦП обычно требуют определенного времени с момента запуска до считывания измеренного значения. Часть работы системное программное обеспечение контроллера выполняет по прерываниям. Грамотно реализованная система исполнения нигде и никогда не использует пустые циклы ожидания готовности аппаратуры. Для прикладного программиста все эти детали не важны. Существенно только то, что значения входов обновляются автоматически исключительно в начале каждого рабочего цикла.

Общая продолжительность рабочего цикла ПЛК называется временем сканирования. Время сканирования в значительной степени определяется длительностью фазы кода пользовательской программы. Время, занимаемое прочими фазами рабочего цикла, практически является величиной постоянной. Для задачи среднего объема в ПЛК с системой исполнения CoDeSys время распределится примерно так: 98% – пользовательская программа, 2% – все остальное [1,2,5].

Существуют задачи, в которых плавающее время цикла существенно влияет на результат, например это автоматическое регулирование. Для устранения этой проблемы в развитых ПЛК предусмотрен контроль времени цикла. Если отдельные ветви кода управляющей программы выполняются слишком быстро, в рабочий цикл добавляется искусственная задержка. Если контроль времени цикла не предусмотрен, подобные задачи приходится решать исключительно по таймерам [1,2,5].

Время реакции – это время с момента изменения состояния системы до момента выработки соответствующей реакции. Очевидно, для ПЛК время реакции зависит от распределения моментов возникновения события и начала фазы чтения входов. Если изменение значений входов произошло непосредственно перед фазой чтения входов, то время реакции будет наименьшим и равным времени сканирования. Худший случай, когда изменение значений входов происходит сразу после фазы чтения входов. Тогда время реакции будет наибольшим, равным удвоенному времени сканирования минус время одного чтения входов. Иными словами, время реакции ПЛК не превышает удвоенного времени сканирования [5].

Вопросы для самоконтроля

1. Назначение и основные характеристики программируемых логических контроллеров.
2. Что является основной задачей прикладного программирования ПЛК?
3. Какие входы и выходы используются в ПЛК?
4. Назначение аналоговых входов и выходов ПЛК.
5. Назначение дискретных входов и выходов ПЛК.
6. Назначение специализированных входов и выходов ПЛК.
7. Режим реального времени и ограничения на применение ПЛК.
8. Программные обеспечения, реализующие интерфейс человек-машина.
9. Назначение и типы стандартных протоколов обмена данными.
10. Место программируемых логических контроллеров в АСУ ТП.
11. Последовательность рабочего цикла ПЛК.
12. Понятие времени реакции ПЛК.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. *Минаев, И.Г.* Программируемые логические контроллеры : практическое руководство для начинающего инженера [Текст] / И.Г. Минаев, В.В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.
2. *Парр, Э.* Программируемые контроллеры : руководства для инженера. – М.: Бином; Лаборатория знаний, 2007. – 516 с.
3. *Костров Б.В.* Микропроцессорные системы и микроконтроллеры [Текст] / Б.В. Костров, В.Н. Ручкин. – М.: «ТехБук», 2007. – 320 с.
4. *Мелехин, В.Ф.* Вычислительные машины, системы и сети [Текст]: учебник для студ. высш. учеб. заведений / В.Ф. Мелехин, Е.Г. Павловский. – 2-е изд. , стер. – М.: Издательский центр «Академия», 2007. – 560 с.

Дополнительная

5. *Петров, И.В.* Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-Пресс, 2004. – 246 с.

6. *Басалин, П.Д.* Архитектура вычислительных систем [Текст]: Учебник. – Нижний Новгород: Изд-во Нижегородского госуниверситета, 2003. – 243 с.
7. *Иванов, И.Ю.* Микропроцессорные устройства систем управления [Текст]: Учебное пособие / Ю.И. Иванов, В.Я. Ягай. – Таганрог: Изд-во ТРТУ, 2005. – 133.
8. *Бойко, В.И.* Схемотехника электронных систем. Микропроцессоры и микроконтроллеры [Текст]: Учебник. – БХВ-Петербург, 2004. – 464 с.
9. *Корнеев, В.В.* Современные микропроцессоры [Текст] / В.В. Корнеев, А.В. Киселёв. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003. – 448 с.

Лекция 2

2. ИНСТРУМЕНТЫ ПРОГРАММИРОВАНИЯ ПЛК

2.1 Комплексы проектирования МЭК

Контроллеры, программирование которых осуществляется со встроенного или выносного пульта, встречаются сегодня достаточно редко. Как правило» это простые специализированные ПЛК, предназначенные для управления освещением по расписанию, регулировки температуры и т.д. Все программирование таких контроллеров сводится обычно к заданию набора констант [1,2,4].

Для программирования ПЛК универсального назначения применяются ПК. Процесс разработки и отладки программного обеспечения происходит при помощи специализированных комплексов программ, обеспечивающих комфортную среду для работы программиста

Традиционно все ведущие изготовители программируемых ПЛК имеют собственные фирменные наработки в области инструментального программного обеспечения. Безусловно, большинство из них представляют удобные инструменты, оптимизированные под конкретную аппаратуру. Понятно, что в разработке универсальных систем программирования, приемлемых для своих ПЛК и для ПЛК конкурентов, изготовители не заинтересованы. Кроме того, это достаточно сложная задача. Системы программирования ПЛК небольших фирм в лучшем случае реализуют один из языков МЭК с некоторыми расширениями, призванными сохранить совместимость со своими же более ранними (нестандартными) системами. Крупнейшие лидеры рынка ПЛК предлагают сегодня очень мощные комплексы с поддержкой МЭК-языкав, также сохраняющие преемственность и фирменные традиции («Concept» Schneider Electric, «S7» Siemens) [1-4].

Открытость МЭК-стандарта – с одной стороны и сложность реализации высококлассных комплексов программирования – с другой, привели к появлению специализированных фирм, занятых исключительно инструментами программирования ПЛК.

Наибольшей известностью в мире пользуются комплексы CoDeSys. CoDeSys – это один из самых развитых функционально полных инструментов программирования МЭК 61131-3 [4].

2.2 Инструменты комплексов программирования ПЛК

Главная задача инструментов комплекса программирования ПЛК состоит в автоматизации работы разработчика прикладной системы. Хорошо организованная среда программирования сама толкает к созданию надежного, читабельного и пригодного для повторного применения кода.

В интегрированных комплексах программирования ПЛК сложился определенный набор возможностей, позволяющий относить их к средствам быстрой разработки. Многие приемы являются общими и для систем программирования компьютеров и, вероятно, покажутся вам знакомыми. Сервисные функции систем программирования не являются требованием стандарта. Но от полноты набора доступных программисту инструментов существенно зависит скорость и качество его работы [4-8].

2.2.1 Встроенные редакторы

Классические (с командной строкой) ассемблеры и компиляторы обрабатывают текст файла, содержащего программный модуль, и формируют объектный код. Исходный текст программы записывается в любом текстовом редакторе. Интегрированная среда предполагает наличие встроенного редактора [1-4].

2.2.2 Текстовые редакторы

Интеграция в единую среду программирования предполагает наличие у текстовых редакторов нескольких существенных свойств [4]:

- возможность быстрого ввода стандартных текстовых элементов. Комбинаций клавиш быстрого ввода, или контекстно-зависимые меню команд, предлагают мгновенную вставку в текст операторов, функций, функциональных блоков. При чем речь идет не только о стандартных элементах, но и о созданных программистом в текущем проекте;
- автоматическое объявление переменных. Если при вводе текста программы вы используете новую переменную, система автоматически поместит необходимое описание в разделе объявлений. Тип переменной и начальное значение задаются в диалоговом окне. В этом помогают меню, весь ввод обычно выполняется мышью, без помощи клавиатуры;
- представление раздела объявлений переменных в виде текста или картотеки таблиц, разделенных и отсортированных по функциональному значению (входные переменные» локальные и т.д.);
- проверка синтаксиса и автоматическое форматирование ввода. Редактор автоматически контролирует введенный текст и выделяет цветом ключевые слова, константы и комментарии. В результате текст не только легко читается» но и оказывается синтаксически проверенным еще до трансляции

2.2.3 Графические редакторы

Графические редакторы еще более тесно связаны с контекстом конкретных языков. Они должны обеспечивать следующие возможности [1-4]:

- автоматическая трассировка соединений компонентов. Программисту вообще не приходится рисовать соединения. При вставке и удалении компонентов система автоматически проводит графические соединительные линии (рис.2.1);

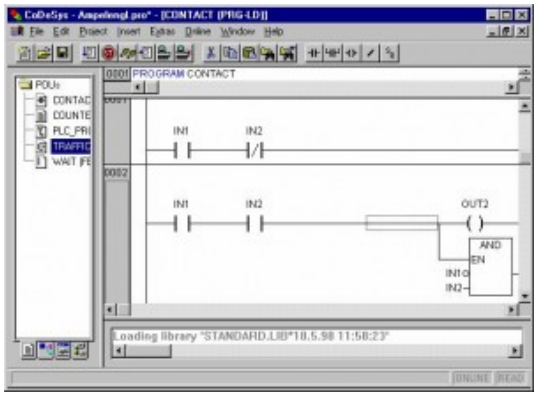

Рис.2.1 – Графический редактор релейных схем

– автоматическая расстановка компонентов. Местоположение компонента на экране определяется автоматически с учетом порядка выполнения. Этим свойством обладают графические редакторы CoDeSys и OpenPCS. В других комплексах программист выбирает местоположение компонента вручную, координаты компонента сохраняются при записи проекта (рис.2.2). Команда индикации порядка выполнения добавляет в изображение компонента порядковый номер [4].

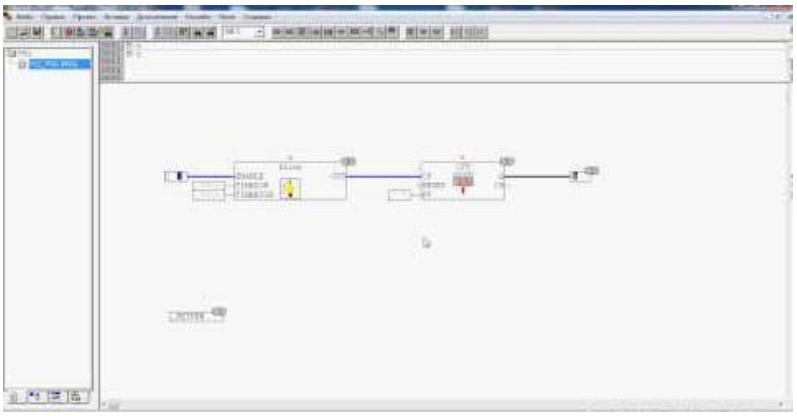

Рис.2.2 – Ручное размещение компонента

– автоматическая нумерация цепей;
– копирование и перемещение выделенной графической группы компонентов с учетом их индивидуальной специфики (рис.2.3);

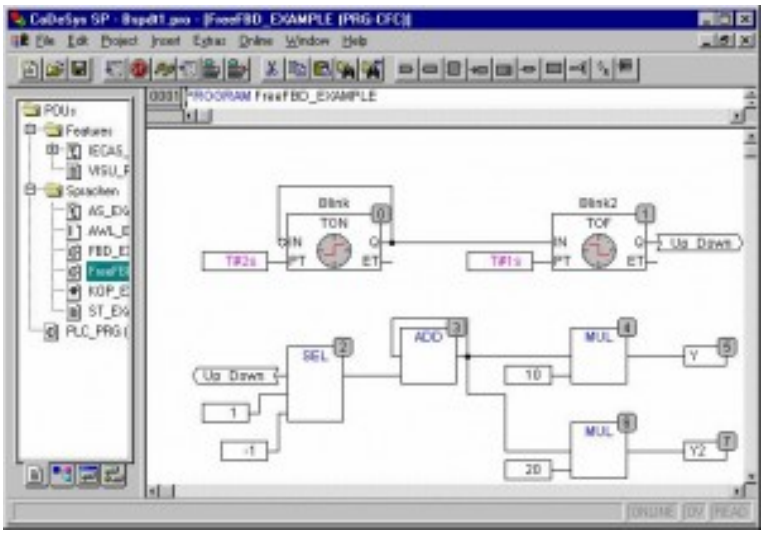

Рис.2.3 Графический редактор функциональных блоковых диаграмм

– произвольное масштабирование изображения с целью наилучшего представления или отдельное окно общего вида. Для анализа больших разветвленных графических диаграмм удобно иметь возможность увидеть всю диаграмму или достаточно релевантную ее часть целиком.

В режиме исполнения встроенные редакторы отображают тексты и графические диаграммы. При этом [4-8]:

– мгновенные значения переменных видны непосредственно в окне редактора и доступны для изменения;
– активные цепи выделены жирными линиями и цветом. Для графических диаграмм наглядно отражается последовательность выполнения.

2.2.4 Средства отладки

Стандартный набор отладочных функций включает в себя [1,2,4]:
– унифицированный механизм соединения с ПЛК. Работа инструментов отладки не зависит от способа соединения контроллера с отладчиком. Не имеет значения, эмулируется ли контроллер на том же самом компьютере (рис.2.4), подключен ли через последовательный порт ПК или даже расположен в другой стране и связан через Интернет;

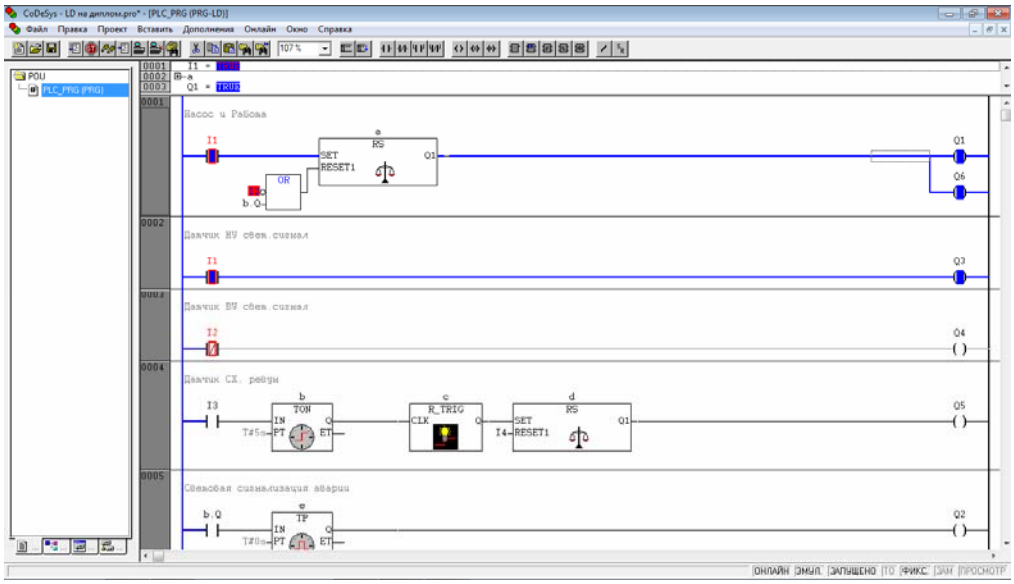

Рис.2.4 – Эмуляция ПЛК

– загрузку кода управляющей программы в оперативную память и электрически перепрограммируемую память ПЛК;
– автоматический контроль версий кода. Проверка соответствия кода содержащегося в памяти ПЛК и кода полученного после текущей компиляции;
– выполнение управляющей программы в режиме реального времени;
– режим останова. Останов означает прекращение выполнения только кода управляющей программы. Все прочие фазы Рабочего цикла выполняются. Способность наблюдать значения входов и управлять выходами ПЛК вручную сохраняется. В этом режиме можно проводить тестирование и настройку датчиков и механизмов объекта управления;
– сброс ПЛК. Может быть несколько видов сброса. В стандарте МЭК предусмотрено два вида сброса «горячий» и «холодный». Первый включает перевод управляющей программы в исходное состояние и выполнение начальной инициализации переменных. Во втором виде сброса добавляется начальная инициализация переменных, размещенных в энергонезависимой области памяти. В CoDeSys предусмотрен еще и «заводской» сброс (original), удаляющий пользовательскую

программу и восстанавливающий состояние контроллера, в котором он поступает с завода изготовителя. Кроме того, в ПЛК может произойти аппаратный сброс путем выключения питания или перезапуска микропроцессора. Система программирования должна адекватно реагировать в случае аппаратного сброса» Детальная реакция на команды сброса определяется системой исполнения. Поэтому здесь возможны некоторые отличия для разных ПЛК, даже в одной среде программирования;

- мониторинг и изменение мгновенных значений всех переменных проекта» включая входы-выходы ПЛК. Для удобства работы значения представляются в заданной пользователем системе счисления [4];
- фиксацию переменных, включая входы-выходы. Фиксированные переменные будут получать заданные значения в каждом рабочем цикле независимо от реального состояния ПЛК и действий управляющей программы. Данная функция позволяет имитировать элементарные внешние события в лабораторных условиях и избегать нежелательной работы исполнительных механизмов при отладке на «живом» объекте управления. Неуправляемая работа механизмов может привести к поломке и представлять опасность для окружающих людей;
- выполнение управляющей программы шагами по одному рабочему циклу. Применяется при проверке логической правильности алгоритма;
- пошаговое выполнение команд программы и задание точек останова;
- просмотр последовательности вызовов компонентов в точке останова;
- графическую трассировку переменных. Значения нужных переменных запоминаются в циклическом буфере и представляются на экране ПК в виде графиков. Запись значений можно выполнять в конце каждого рабочего цикла либо через заданные периоды времени. Трассировка запускается вручную или синхронизируется с заданным изменением значения определенной (триггерной) переменной [4];

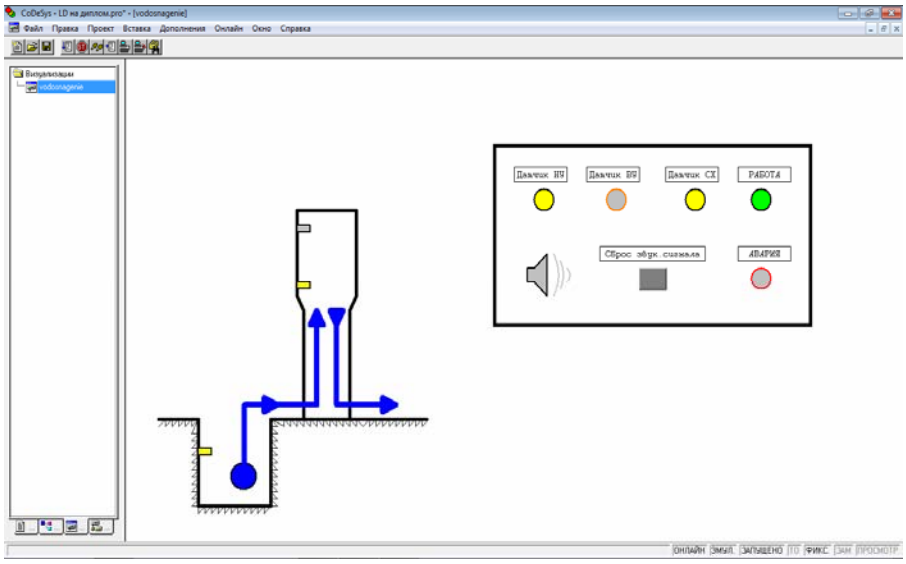

Рис.2.5 – Визуализация проекта

- визуализацию – анимационные картинки, составленные из графических примитивов, связанных с переменными программы. Значение переменной может

определять координаты, размер или цвет графического объекта. Графические объекты включают векторные геометрические фигуры или произвольные растровые изображения. Визуализация может содержать элементы обратной связи» например кнопки, ползунки и т. д. (рис.2.5). С помощью визуализации создается изображение, моделирующее объект управления или систему операторского управления [4-8].

2.2.5 Средства управления проектом

Все программные комплексы обязательно содержат средства управления проектом. Эту задачу решает менеджер проекта, в обязанности которого входит [4]:

- представление всех элементов проекта и общей его структуры в удобном виде. Создание, удаление, переименование и копирование компонентов. Автоматический вызов соответствующих редакторов для любой глубины вложения программных компонентов. Настройка ресурсов;
- управление процессом трансляции и сборки кода. Настройка опций транслятора;
- сравнение и выборочное слияние нескольких проектов или их версий;
- управление библиотеками. Здесь существуют две задачи. Первая – это включение необходимых библиотек в состав проекта, а вторая — это создание и сопровождение новых библиотек;
- документирование проекта. Документирование проекта в комплексах МЭК-программирования предусматривает распечатку всех данных проекта, включая [4]:
  - текстовое описание, дата создания и авторские права;
  - описание переменных и реализацию всех компонентов проекта;
  - ресурсы проекта — конфигурацию ПЛК, описание глобальных переменных, настройки задач, список и состав библиотечных модулей;
  - таблицу перекрестных ссылок и стек вызовов;
  - окно трассировки.

Естественно, нельзя ожидать от системы программирования полного комплекта документации в соответствии с требованиями ЕСКД. Под словами «полная документация» в руководстве по применению системы понимается только то, что по данному печатному документу можно полностью и однозначно восстановить проект.

CoDeSys позволяет составить специальные файлы комментариев на разных языках (русский, английский и т.д.).

Средства восстановления проекта. В реальной жизни нельзя исключать ситуацию, что исходные файлы проекта окажутся утраченными. В это время обязательно возникнет необходимость внести поправки в работу готовой программы. Эта задача имеет три решения.

1. *Декомпиляция кода*. Исполняемый код считывается из памяти ПЛК и преобразуется в МЭК-программы. Для систем генерирующих машинный код эта задача практически невыполнима. Безусловно, можно дизассемблировать код в IL или ST. Но это ненамного лучше, чем обычное машинно-зависимое дизассемблирование. Структура программы получится отличной от исходного представления. Как правило, разобраться в такой программе сложнее, чем написать заново. Для интерпретирующих систем ситуация значительно лучше. Так, OpenPCS способен восстановить программу из исполняемого кода IL абсолютно адекватно, естественно, с потерей комментариев. Декомпиляция – это крайняя мера. Важное практическое значение она имела во времена преобладания автономных пультов программирования ПЛК и при отсутствии надежных устройств хранения информации [4-8].

2. *Сжатие всех файлов проекта и сохранение в памяти ПЛК*. Современные мощные алгоритмы компрессии и существенное удешевление памяти делают такой подход все более популярным (MULTIPROG, CoDeSys), Безусловно, при наличии достаточного объема памяти ПЛК это наиболее удобный способ архивации.

3. *Правильная организация работы*. В комплекс разработчика включается утилита для периодической архивации проектов и сохранения на сервере, сменных носителях, в печатном виде и отправки по электронной почте. В архив помещаются исходные файлы, включенные в проект библиотеки, объектные файлы, текстовое описание архива и любые другие нужные файлы. Промежуточные версии проекта не перезаписываются, а хранятся независимо, что позволяет осуществить быстрый откат при выборе неудачного решения. В связи с появлением накопителей большой емкости и надежных перезаписываемых оптических носителей такой подход не имеет технических препятствий [4].

Средства обеспечения безопасности. Возможность просмотра и модификации проекта закрывается парольным доступом или аппаратным ключом. Посторонний человек не должен иметь возможности читать, распечатывать и модифицировать проект.

Сквозной (по всем программам проекта, разделам объявлений, конфигурации и др.) *контекстный поиск и замена*.

Средства тестирования «разумности» проекта. Вспомогательные средства, позволяющие отыскать странные и потенциально опасные моменты в программах. Например, объявленные, но не использованные переменные, использование одной области памяти разными переменными или в разных параллельных задачах, присваивание разных значений выходу ПЛК в одном рабочем цикле и т. д. Подобные «трюки» сами по себе не являются ошибками. Но они часто приводят к сложно обнаруживаемым паразитным эффектам. Средства тестирования помогают отыскать тонкие места в программах, не создавая препятствий там, где эти приемы применены осмысленно.

Средства импорта и экспорта проектов в другие комплексы программирования.

Перечисленные выше средства управления проектами позволяют создавать высококачественные проекты с минимумом затрат времени [1,2,4].

2.3 Комплекс CoDeSys

Комплекс CoDeSys разработан фирмой 3S (Smart Software Solutions). Это универсальный инструмент программирования контроллеров и встраиваемых систем на языках МЭК 61131-3, не привязанный, к какой-либо аппаратной платформе и удовлетворяющий современным требованиям быстрой разработки программного обеспечения [4-8].

Ядро системы исполнения CoDeSys написано на языке С. Существует несколько модификаций оптимизированных для различных микропроцессоров (включая PC-совместимые). Для привязки к конкретному ПЛК требуется адаптация, касающаяся низкоуровневых ресурсов – распределение памяти, интерфейс связи и драйверы ввода-вывода.

Среди особенностей данного пакета можно отметить следующее [4].

– Прямая генерация машинного кода. Генератор кода CoDeSys – это классический компилятор, что обеспечивает очень высокое быстродействие программ пользователя.

- Полноценная реализация МЭК-языков, в некоторых случаях даже расширенная.
- «Разумные» редакторы языков построены таким образом, что не дают делать типичные для начинающих МЭК программистов ошибки.
- Встроенный эмулятор контроллера позволяет проводить отладку проекта без аппаратных средств. Причем эмулируется не некий абстрактный контроллер, а конкретный ПЛК с учетом аппаратной платформы- При подключении реального контроллера (режим online) отладчик работает аналогичным образом.
- Встроенные элементы визуализации дают возможность создать модель объекта управления и проводить отладку проекта без изготовления средств имитации. Существует «операционная» версия CoDeSys. Это компактное приложение, выпблнякИцее только визуализацию» без средств разработки. Во многих простых случаях нет необходимости приобретать отдельно SCADA-систему. Серверы данных (DDE и OPC) также входят в стандартный пакет поставки.
- Очень широкий набор сервисных функций, ускоряющих работу программиста.
- В настоящее время создано более 150 адаптации комплекса CoDeSys: Фирма 3S не скрывает своих клиентов. Все они открыто взаимодействуют друг с другом и совместно работают над совершенствованием программного инструментария [4].

Вопросы для самоконтроля

1. Понятие комплексы проектирования МЭК.
2. Инструменты комплексов программирования ПЛК.
3. Встроенные редакторы комплексов программирования ПЛК.
4. Основные свойства текстовых редакторов комплексов программирования ПЛК.
5. Возможности графических редакторов комплексов программирования ПЛК.
6. Стандартный набор отладочных функций комплексов программирования ПЛК.
7. Средства управления проектом комплексов программирования ПЛК.
8. Комплекс CoDeSys для программирования ПЛК.
9. Особенности пакета комплекса CoDeSys.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. *Минаев, И.Г.* Программируемые логические контроллеры : практическое руководство для начинающего инженера [Текст] / И.Г. Минаев, В.В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.
2. *Парр, Э.* Программируемые контроллеры : руководства для инженера. – М.: Бином; Лаборатория знаний, 2007. – 516 с.
3. *Костров Б.В.* Микропроцессорные системы и микроконтроллеры [Текст] / Б.В. Костров, В.Н. Ручкин. – М.: «ТехБук», 2007. – 320 с.

Дополнительная

4. *Петров, И.В.* Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-Пресс, 2004. – 246 с.
5. *Басалин, П.Д.* Архитектура вычислительных систем [Текст]: Учебник. – Нижний Новгород: Изд-во Нижегородского госуниверситета, 2003. – 243 с.
6. *Иванов, И.Ю.* Микропроцессорные устройства систем управления [Текст]: Учебное пособие / Ю.И. Иванов, В.Я. Ягай. – Таганрог: Изд-во ТРТУ, 2005. – 133.

7. *Бойко, В.И.* Схемотехника электронных систем. Микропроцессоры и микроконтроллеры [Текст]: Учебник. – БХВ-Петербург, 2004. – 464 с.
8. *Корнеев, В.В.* Современные микропроцессоры [Текст] / В.В. Корнеев, А.В. Киселёв. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003. – 448 с.

Лекция 3

3. СЕМЕЙСТВО ЯЗЫКОВ МЭК

3.1 Релейные диаграммы (LD)

3.1.1 Цепи

Релейная схема представляет собой две вертикальные шины питания, между ними расположены горизонтальные цепи, образованные контактами и обмотками реле. Количества контактов в цепи произвольно, реле одно. Если последовательно соединенные контакты замкнуты, ток идет по цепи и реле включается (в примере на рис.3.1 Lampl). При необходимости можно включить параллельно несколько реле, последовательное включение не допускается.

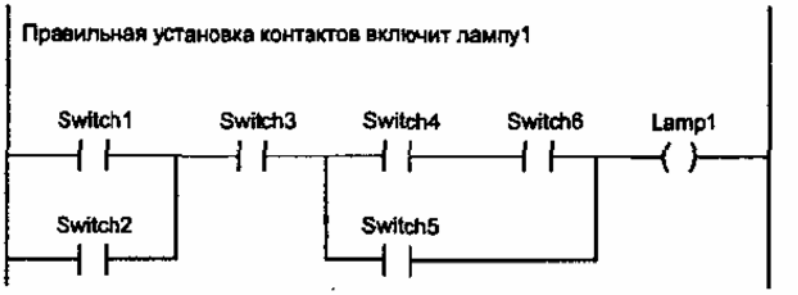

Рис.3.1 Схема LD из одной цепи

В LD каждому контакту ставится в соответствие логическая переменная» определяющая его состояние. Если контакт замкнут, то переменная имеет значение ИСТИНА. Если разомкнут – ЛОЖЬ. Имя переменной пишется над контактом и фактически служит его названием.

Последовательное соединение контактов или цепей равноценно логической операции И. Параллельное соединение образует монтажное ИЛИ.

Цепь может быть либо замкнутой (ON), либо разомкнутой (OFF). Это как раз и отражается на обмотке реле и соответственно на значении логической переменной обмотки (ИСТИНА/ЛОЖЬ).

Зрительное восприятие LD-диаграмм должно быть интуитивно понятным. Для России этому несколько мешает принятая система условных графических обозначений, базирующаяся на американском стандарте NEMA. Преимущество таких обозначений состоит в возможности применения символов псевдографики для построения LD-диаграмм.

Табл.3.1 – Сопоставление обозначений базовых элементов LD и обозначений ЕСКД

LD	ЕСКД	Обозначение
		Нормально разомкнутый контакт
		Нормально замкнутый контакт
		Обмотка реле

Сопоставление обозначений базовых элементов LD и обозначений ЕСКД приведено в таблице 3.1.

Контакт может быть инверсным – нормально замкнутым. Такой контакт обозначается с помощью символа |/| и замыкается, если значение переменной ЛОЖЬ, Инверсный контакт равнозначен логической операции НЕ.

Переключающий контакт образуется комбинацией прямого и инверсного контактов (рис.3.2).

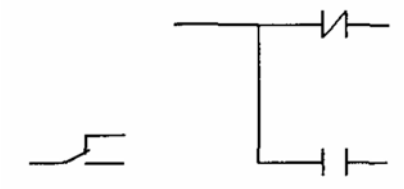

Рис.3.2 Переключающий контакт

Обмотки реле также могут быть инверсными, что обозначается символом (/). Если обмотка инверсная, то в соответствующую логическую переменную копируется инверсное значение состояния цепи.

3.1.2 Реле с самофиксацией

Помимо обычных реле, в релейных схемах часто применяются поляризованные реле. Такое реле имеет две обмотки, переключающие его из одного положения в другое. Переключение производится импульсами тока. При отключении тока питания поляризованное реле остается в заданном положении, что реализует элементарную ячейку памяти.

В LD такое реле реализуется при помощи двух специальных обмоток SET и RESET. Обмотки типа SET обозначаются буквой S внутри круглых скобок (S). Обмотки типа RESET обозначаются буквой R. Если соответствующая обмотке (S) переменная принимает значение ИСТИНА, то сохраняет его бесконечно. Вернуть данную переменную в ЛОЖЬ можно только обмоткой (R).

Очевидно, что полной аналогии с поляризованным реле программно достичь невозможно. Даже если значение логического выхода сохраняется в энергонезависимой памяти, состояние самой электрической цепи при выключенном питании ПЛК определяется его схематикой. Фиксация безопасного положения аппаратуры при аварии питания системы управления может быть достигнута только аппаратно.

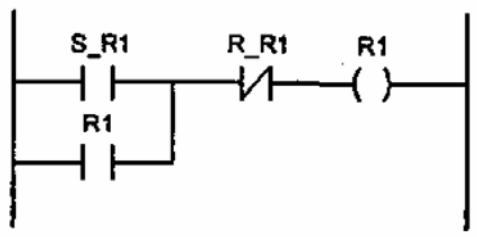

Рис.3.3. Реле с самофиксацией

Условие выключения реле не всегда равносильно отсутствию условия включения. Благодаря (R) и (S) обмоткам условия включения и выключения реле можно

формировать совершенно независимо, причем в любой цепи и сколько угодно раз. Обмотки (R) и (S) обеспечивают фиксацию условий управления, что необходимо при реализации автоматов с памятью.

Самофиксацию несложно организовать и на простом реле, используя дополнительный контакт в цепи питания. Пример этого представлен на рис.3.3.

Контакт S_R1 включает, а R_R1 выключает реле R1, благодаря контакту R1 реле получает питание после размыкания SR1. Применение SET/RESET-обмоток не дает ничего принципиально нового, но делает LD-диаграмму проще.

3.1.3 Порядок выполнения и обратные связи

Идеология релейных схем подразумевает параллельную работу всех цепей. Ток во все цепи подается одновременно.

В LD решение диаграммы выполняется последовательно слева направо и сверху вниз. В каждом рабочем цикле однократно выполняются все цепи диаграммы, что и создает эффект параллельности работы цепей. Любая переменная в рамках одной цепи всегда имеет одно и то же значение. Если даже реле в цепи изменит переменную, то новое значение поступит на контакты только в следующем цикле. Цепи расположенные ниже, получат новое значение переменной сразу. Цепи расположенные выше – только в следующем цикле. Строгий порядок выполнения схемы очень важен. Случайный или даже истинно параллельный порядок выполнения цепей мог бы приводить к эффекту «гонок», встречающемуся в электронных схемах с триггерами. Благодаря жесткому порядку выполнения LD-диаграммы сохраняют устойчивость при наличии обратных связей.

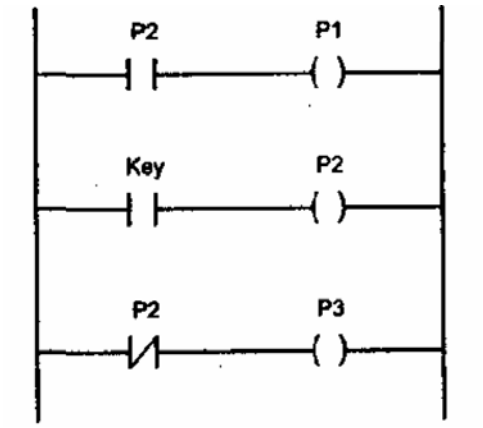

Рис.3.4. LD диаграмма с обратной связью

В приведенной на рис.3.4 схеме включение Key вызовет мгновенное (в том же цикле) включение P2 и отключение P3. Реле P1 будет включено только в следующем цикле, причем даже если Key уже в обрыве (ЛОЖЬ).

Рис.3.5. Генератор единичных импульсов

Используя вышеописанный принцип цикличности выполнения LD-диаграмм, очень легко построить генератор единичных импульсов. Пример такого построения дан на рис.3.5 Период импульсов на реле P1 будет равен удвоенной длительности рабочего цикла ПЛК.

3.1.4 Управление порядком выполнения

Порядок выполнения цепей диаграммы можно принудительно изменять, используя метки (labels) и переходы (jumps).

Метку можно ставить только в начало цепи. Имена меток подчинены правилам наименования переменных. Для наглядности можно закончить метку двоеточием. Двоеточие не образует новой метки. Так, Ml: и Ml это одно и то же. Цепь может иметь только одну метку и один переход. Переход равнозначен выходному реле и выполняется, если выходная переменная имеет значение ИСТИНА. Переход может быть инверсным, в этом случае он выполняется при значении цепи ЛОЖЬ. Используя переход» можно пропустить выполнение части диаграммы. Пропущенные цепи не сбрасываются, а именно не выполняются – замирают в том положении, в котором были ранее. Переход вверх допускается и позволяет создавать циклы (рис.3.6).

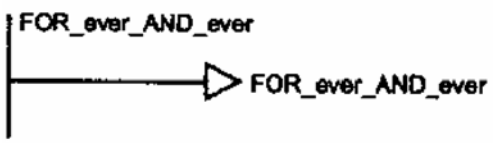

Рис.3.6 Простейший бесконечный цикл

Идеологически переходы противоречат аналогии LD с релейными схемами, нарушая законы электрических цепей. В схеме LD с переходами разобраться бывает сложно. Желательно не заниматься управлением порядком выполнения LD-диаграммы в ней самой, а использовать для этого более выразительные средства. Например, разделить LD-диаграмму на модули (действия), а порядок выполнения описать в SFC.

Специальный переход RETURN прекращает выполнение LD диаграммы. Если RETURN встречается в основной программе (PLC_PRG), рабочий цикл прерывается. В функциях и функциональных блоках происходит возврат в место вызова. Иными словами, использование перехода RETURN аналогично по смыслу оператору RETURN в текстовых языках.

3.1.5 Расширение возможностей LD

В LD-диаграмму можно вставить функции и функциональные блоки. Функциональные блоки должны иметь логические вход и выход. На рис.3.7 показан пример организации цикла на 10 повторов на базе функционального блока декрементный счетчик.

Первая цепь загружает счетчик числом повторов. Вторая цепь – генератор единичных импульсов. Третья – декрементный счетчик с проверкой условия окончания цикла. Тело цикла на рисунке 3.7 не показано.

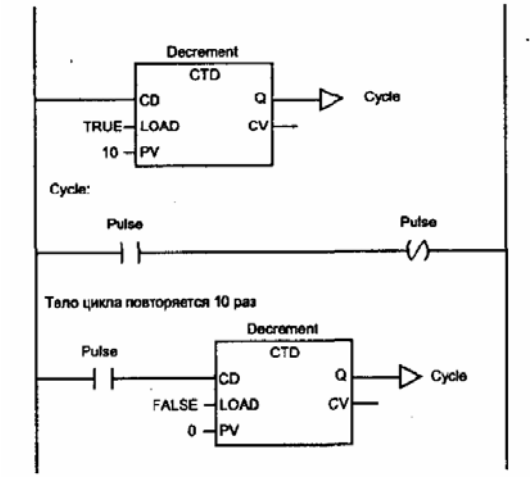

Рис.3.7. Цикл на 10 повторов на базе функционального блока декрементный счетчик

Для включения в диаграмму функций в них искусственно вводится добавочный логический вход, обозначаемый EN (Enable) (рис.3.8). Логическое значение на входе EN разрешает или запрещает выполнение функции. Сама функция не терпит никаких изменений при добавлении входа EN.

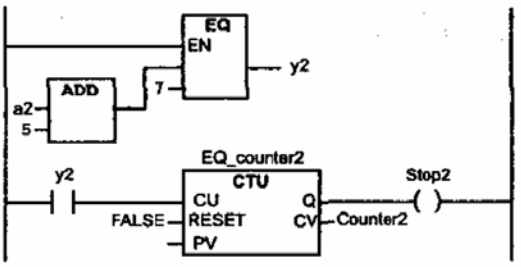

Рис.3.8. Использование оператора EQ, управляемого по входу EN

В первой редакции стандарт МЭК определял контакты и обмотки» управляемые фронтами импульсов: контакт |P| и обмотка (P) переднего фронта» контакт |N| и обмотка (N) заднего фронта. В настоящее время поддержка таких контактов и обмоток не является обязательной» так как аналогичную цепь легко можно построить при помощи функциональных блоков R_TRIG и F_TRIG.

3.1.6 LD-диаграммы в режиме исполнения

В режиме Online обмотки реле, контакты и проводники, находящиеся в состоянии On (под током), окрашены голубым (цвета по умолчанию) цветом. CoDeSys позволяет менять значения логических переменных (ИСТИНА/ЛОЖЬ) непосредственно в графической диаграмме двойным щелчком мыши на имени переменной. Значения входов-выходов функциональных блоков отображаются числовыми значениями.

Точка останова может устанавливаться только целиком на цепь. Для установки или сброса точки останова необходимо щелкнуть мышью по номеру цепи. В режиме останова номер цепи подсвечен красным. Пошаговое – по одной цепи выполнение достигается командами «Step over» и «Step in».

3.2 Функциональные блоковые диаграммы (FBD)

3.2.1 Отображение POU

Диаграмма FBD строится из компонентов, отображаемых на схеме прямоугольниками. Входы POU изображаются слева от прямоугольника, выходы справа. Внутри прямоугольника указывается тип POU и наименования входов и выходов. Для экземпляра функционального блока его наименование указывается сверху, над прямоугольником. В графических системах программирования прямоугольник компонента может содержать картинку, отражающую его тип. Размер прямоугольника зависят от числа входов и выходов и устанавливается графическим редактором автоматически. Пример графического представления экземпляра Blinker функционального блока BLINK дан на рис.3.9.

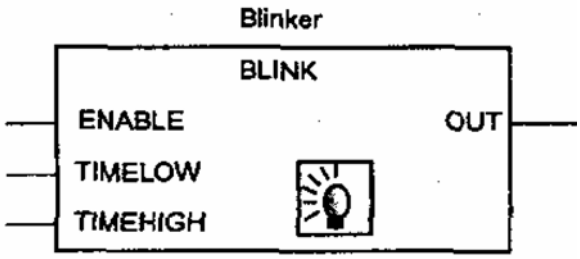

Рис.3.9 Графическое представление экземпляра функционального блока

Программа в FBD не обязательно должна представлять большую единую схему. Как и в LD, диаграмма образуется из множества цепей» которые выполняются одна за другой. В CoDeSys все цепи одного POU отображаются в едином графическом окне, пронумерованные и разделенные горизонтальными линиями. Значения переменных, вычисленные в одной цепи, доступны в последующих цепях сразу в том же рабочем цикле.

3.2.2 Соединительные линии

Прямоугольники POU в FBD соединены линиями связи. Соединения имеют направленность слева направо. Вход блока может быть соединен с выходом блока, расположенного слева от него. Помимо этого, вход может быть соединен с переменной или константой. Соединение должно связывать переменные или входы и выходы одного типа. В отличие от компонента переменная изображается на диаграмме без прямоугольной рамки. Ширина соединительной линии в PBD роли не играет. Стандарт допускает использование соединительных линий разной ширины и стиля для соединений разного типа.

3.2.3 Порядок выполнения FBD

Выполнение FBD-цепей идет слева направо, сверху вниз. Блоки, расположенные левее, выполняются раньше. Блок начинает вычисляться только после вычисления значений всех его входов. Дальнейшие вычисления не будут продолжены до вычисления значений на всех выходах. Другими словами, значения на всех выходах графического блока появляются одновременно. Вычисление цепи считается

законченным только после вычисления значений на выходах всех входящих в нее элементов.

В некоторых системах программирования пользователь имеет возможность свободно передвигать блоки с сохранением связей. В этом случае ориентироваться нужно исходя из порядка соединений. Редактор FBD CoDeSys автоматически расставляет блоки в порядке выполнения.

3.2.4 Инверсия логических сигналов

Инверсия логического сигнала в PBD изображается в виде окружности на соединении, перед входом или переменной. Инверсия не является свойством самого блока и может быть легко добавлена или отменена непосредственно в диаграмме- В CoDeSys это делается командой «Negate».

3.2.5 Соединители и обратные связи

Соединители (connectors) представляют собой поименованное соединение, которое можно разорвать и перенести в следующую цепь. Такой прием может понадобиться при ограниченной ширине окна редактора FBD. В CoDeSys ширина окна не ограничена, поэтому соединители здесь не нужны.

Стандарт не запрещает соединения, идущие с выхода блока на свой вход или вход ранее исполняемых блоков. Обратная связь не образует цикл, подобный FOR, просто некоторое вычисленное значение поступит на вход при следующем вызове диаграммы. Фактически это означает неявное создание переменной, которая сохраняет свое значение между вызовами диаграммы. Для устранения неоднозначности необходимо присвоить безопасное начальное значение переменной обратной связи.

В редакторе FBD CoDeSys обратные соединения запрещены. Для создания обратной связи используйте явно объявленную внутреннюю переменную.

При необходимости переноса или разветвления соединения в другие цепи также необходимо использовать промежуточные локальные переменные.

3.2.6 Метки, переходы и возврат

Порядок выполнения FBD-цепей диаграммы можно принудительно изменять, используя метки и переходы, точно так же, как и в релейных схемах.

Метка ставится в начале любой цепи, являясь, по сути, названием данной цепи- Цепь может содержать только одну метку. Имена меток подчинены общим правилам наименования идентификаторов МЭК. Графический редактор автоматически нумерует цепи диаграммы. Эта нумерация применяется исключительно для документирования и не может заменять метки.

Переход обязательно связан с логической переменной и выполняется, если переменная имеет значение ИСТИНА. Для создания безусловного перехода используется константа ИСТИНА, связанная с переходом. Метки и переходы в FBD представлены в примере, показанном на рис.3.10. Последняя цепь является пустой. Пустая цепь обозначается единственной константой TRUE.

Оператор возврата RETURN можно использовать в FBD так же, как и переход на метку» т. е. в связке с логической переменной. Возврат приводит к немедленному

окончанию работы программного компонента и возврату на верхний уровень вложений. Для основной программы это начало рабочего цикла ПЛК.

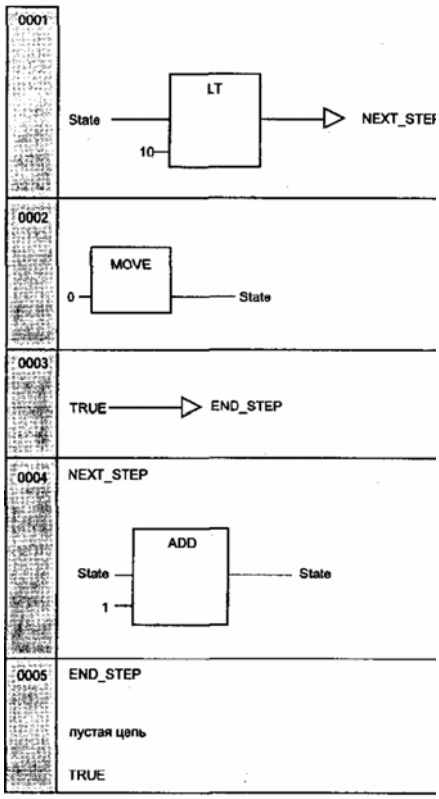

Рис. 7.17. Метки и переходы в FBD

Вопросы для самоконтроля

1. Семейство языков программирования ПЛК.
2. Язык программирования релейные диаграммы (LD)
3. Порядок выполнения и обратные связи в языке программирования релейные диаграммы.
4. Управление порядком выполнения в языке программирования релейные диаграммы.
5. Расширение возможностей языка программирования релейные диаграммы.
6. LD-диаграммы в режиме исполнения.
7. Язык программирования функциональные блоковые диаграммы (FBD).
8. Отображение POU в языке программирования функциональные блоковые диаграммы.
9. Порядок выполнения FBD
10. Соединители и обратные связи в языке программирования функциональные блоковые диаграммы.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. *Минаев, И.Г.* Программируемые логические контроллеры : практическое руководство для начинающего инженера [Текст] / И.Г. Минаев, В.В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.
2. *Парр, Э.* Программируемые контроллеры : руководства для инженера. – М.: Бином; Лаборатория знаний, 2007. – 516 с.

3. *Костров Б.В.* Микропроцессорные системы и микроконтроллеры [Текст] / Б.В. Костров, В.Н. Ручкин. – М.: «ТехБук», 2007. – 320 с.

Дополнительная

4. *Петров, И.В.* Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-Пресс, 2004. – 246 с.
5. *Басалин, П.Д.* Архитектура вычислительных систем [Текст]: Учебник. – Нижний Новгород: Изд-во Нижегородского госуниверситета, 2003. – 243 с.
6. *Иванов, И.Ю.* Микропроцессорные устройства систем управления [Текст]: Учебное пособие / Ю.И. Иванов, В.Я. Ягай. – Таганрог: Изд-во ТРТУ, 2005. – 133.
7. *Бойко, В.И.* Схемотехника электронных систем. Микропроцессоры и микроконтроллеры [Текст]: Учебник. – БХВ-Петербург, 2004. – 464 с.
8. *Корнеев, В.В.* Современные микропроцессоры [Текст] / В.В. Корнеев, А.В. Киселёв. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003. – 448 с.

Лекция 4

4. СТАНДАРТНЫЕ КОМПОНЕНТЫ КОМПЛЕКСОВ МЭК-ПРОГРАММИРОВАНИЯ

4.1 Операторы и функции

4.1.1 Арифметические операторы

Почти все арифметические операторы имеют символьную форму для записи в выражениях языка ST. В других языках МЭК используются вызовы операторов в виде функции. Арифметические операторы являются перегружаемыми: тип результата операции определяется типом операндов (табл.4.1).

Табл.4.1 – Стандартные арифметические операторы

Оператор	Символ	Действие	Типы параметров
ADD	+	Сложение	ANY_NUM, TIME
SUB	–	Вычитание	ANY_NUM, TIME
MUL	*	Умножение	ANY_NUM, TIME
DIV	/	Деление	ANY_NUM, TIME
MOD	MOD	Остаток от деления	ANY_INT
EXPT		Возведение в степень	INI ANY_NUM IN2 ANYJNT
MOVE	:=	Присваивание	ANY

В графических языках блоки MUL и ADD можно расширять, т.е. добавлять произвольное число параметров. Реализация примера в FBD представлена на рис.4.1.

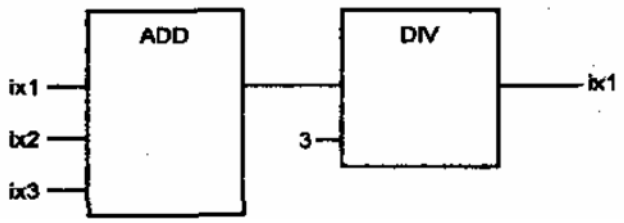

Рис.4.1 Пример графического представления арифметических блоков

Переменные типа TIME можно складывать между собой, вычитать. Одну переменную типа TIME можно умножать и делить на число. Результат во всех случаях будет иметь тип TIME.

Операция MOD применима только на множестве целых чисел. Смысл выражения OUT := INI MOD IN2 можно раскрыть на языке ST так:

```
IF (IN2 = 0) THEN        OUT := 0;
ELSE                     OUT : -IN1 – (IN1/IN2) * IN2 ;
END IF
```

Операция OUT : - EXPT(IN1/IN2) означает OUT = $IN1^{IN2}$. Параметр IN2 должен быть целого типа.

Операция MOVE может иметь только один параметр совместимого типа. В явном виде MOVE встречается только в графических языках. В IL присваивание значения

одной переменной или константы другой переменной выполняется парой инструкций LD, ST.

4.1.2 Операторы битового сдвига

Операторы сдвига применимы для типов ANY_BIT. Все они имеют 2 параметра:
OUT := ОПЕРАТОР(IN,N);

Табл.4.2 – Операции битового сдвига

Оператор	Действие
SHL	Побитный сдвиг операнда IN влево на N бит, с дополнением нулями справа
SHR	Побитный сдвиг операнда IN вправо на N бит» с дополнением нулями слева
ROR	Циклический сдвиг операнда IN вправо на N бит» старшие биты замещаются младшими
ROL	Циклический сдвиг операнда IN влево на N бит» младшие биты замещаются старшими

Пример применения операций сдвига в графической схеме представлен на рис.4.2.

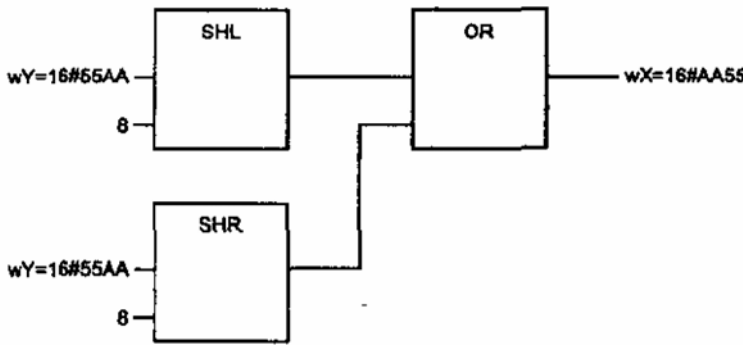

Рис.4.2. Перестановка байт в слове (режим online)

4.1.3 Логические битовые операторы

Битовые операторы применимы для типов ANY_BIT (табл.4.3).

Табл.4.3 – Логические битовые операторы

Оператор	Действие
AND	Побитное И
OR	Побитное ИЛИ
XOR	Побитное исключающее ИЛИ
NOT	Побитное НЕ

Оператор NOT имеет только один параметр.

В FBD блоки AND, OR и XOR можно расширять, т.е. добавлять произвольное число входных параметров. Операция NOT для проводников типов BOOL (инверсия) обозначается в виде окружности.

В языке LD логические операции И, ИЛИ для типа BOOL заменяются монтажными соединениями. Операция AND представляется последовательным соединением контактов, а операция OR параллельным соединением (монтажное ИЛИ) (рис.4.4).

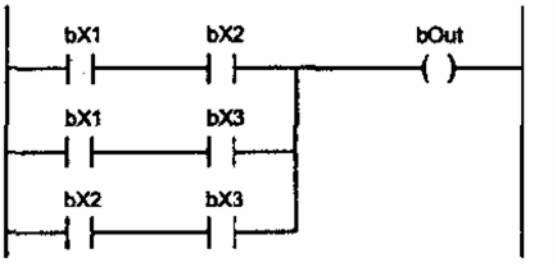

Рис.4.4. Блок голосования два из трех.

4.2 Стандартные функциональные блоки

4.2.1 Таймеры

Таймеры ПЛК принципиально отличаются от таймеров, применяемых в языках общего применения, В языках программирования компьютеров существуют функции задержки (delay, sleep), которые приводят к приостановке выполнения программы на заданное время. Таймера, способного приостановить работу ПЛК, в стандарте МЭК нет. Представьте себе, что на один вход контроллера поступает некоторый сигнал. На второй вход поступает тот же сигнал, но через аппаратный модуль задержки. Именно так работают стандартные таймеры. Временная задержка влияет только на формирование выходных сигналов и не вызывает никакого замедления в программе.

Для правильной работы таймеров необходима аппаратная поддержка. Все экземпляры функциональных блоков таймеров «засекают» время (в CoDeSys во внутренней локальной переменной StartTime), пользуясь общими часами. При проектировании ПЛК достаточно иметь один аппаратный таймер-счетчик, увеличивающийся с постоянной частотой. Аппаратный счетчик должен иметь достаточную разрядность, чтобы исключить возможность переполнения за один рабочий цикл ПЛК.

Нельзя полагаться на то, что повторный вызов экземпляра функционального блока в одном рабочем цикле даст различные результаты. Значения программных таймеров могут обновляться при вызове экземпляра функционального блока или синхронно с обновлением входов. Это зависит от реализации системы исполнения. Не используйте в своих программах циклы (WHILE, REPEAT) с условием окончания итераций по таймеру.

TP генератор импульса

TP			
IN	BOOL	Q	BOOL
PT	TIME	ET	TIME

Запуск таймера происходит по фронту импульса на входе IN. Вход PT задает длительность формируемого импульса. После запуска таймер не реагирует на изменение значения входа IN. Выход ET отсчитывает прошедшее время. При достижении ET значения PT счетчик останавливается, и выход Q сбрасывается в 0.

Временная диаграмма работы таймера TP показана на рис.4.5.

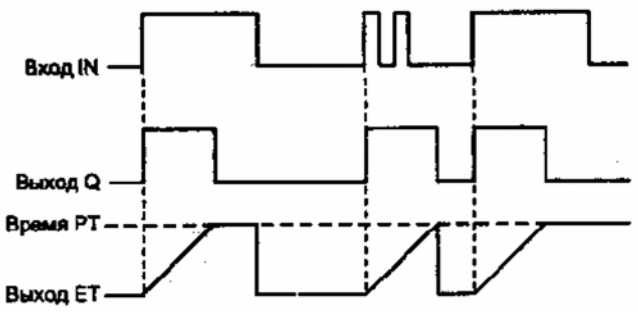

Рис.4.5. Временная диаграмма работы таймера TP

На рис.4.6 показано простейшее применение блока TP в качестве генератора коротких прямоугольных импульсов. Длительность паузы задается таймером. Начальное состояние bx = 0. В первом цикле bx получит значение 1 благодаря инвертору NOT. Так формируется фронт запуска, который поступает на вход IN таймера в третьем цикле. Инвертор формирует фронт запуска по каждому спаду выхода таймера.

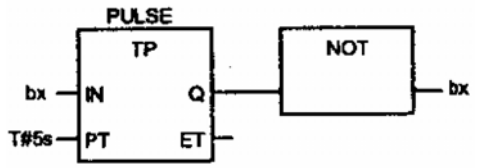

Рис.4.6. Пример использования блока TP

TOF таймер с задержкой выключения

TOF			
IN	BOOL	Q	BOOL
PT	TIME	ET	TIME

По фронту входа IN выход Q устанавливается в TRUE. Сброс счетчика ET и начало отсчета времени происходит по каждому спаду входа IN. Выход Q будет сброшен через заданное PT время после спада входного сигнала. Если во время отсчета вход IN будет установлен в TRUE, то отсчет приостанавливается. Таким образом, выход Q включается по фронту» а выключается логическим нулем продолжительностью не менее PT.

Временная диаграмма работы таймера TOF показана на рис.4.7.

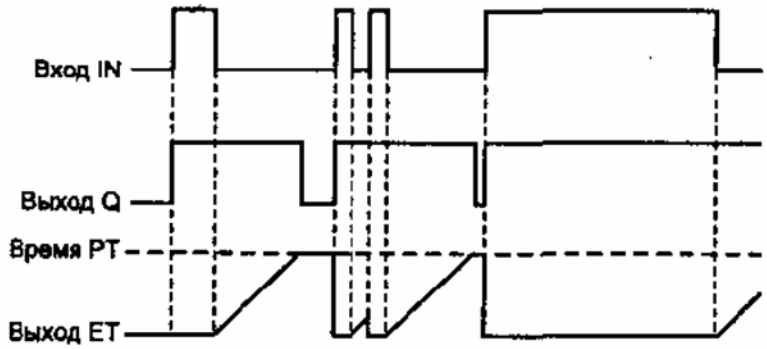

Рис.4.7 Временная диаграмма работы таймера TOF

TON таймер с задержкой включения

TOF			
IN	BOOL	Q	BOOL
PT	TIME	ET	TIME

По фронту входа IN выполняется обнуление счетчика и начинается новый отсчет времени. Выход Q будет установлен в TRUE через заданное PT время, если IN будет продолжать оставаться в состоянии TRUE. Спад входа IN останавливает отсчет и сбрасывает выход Q в FALSE. Таким образом, выход Q включается логической единицей продолжительностью не менее PT, а выключается по спаду входа IN.

Временная диаграмма работы таймера TON показана на рис.4.8.

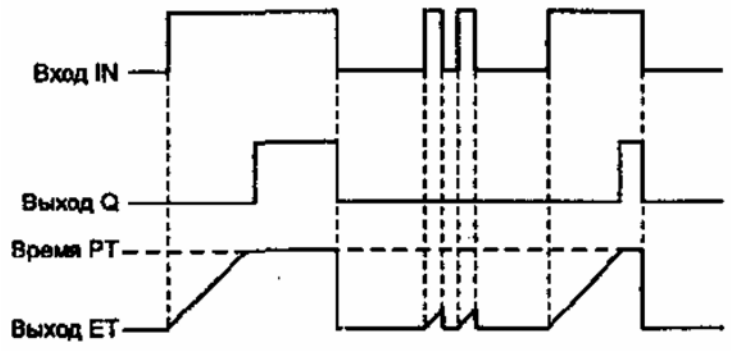

Рис.4.8. Временная диаграмма работы таймера TON

RTC часы реального времени

RTC			
EN	BOOL	Q	BOOL
PDT	DATE_AND_TIME	CDT	DATE_AND_TIME

При создании экземпляра блока, пока вход EN равен FALSE, выход Q равен FALSE, а выход CDT равен DT#1970-01-01-00-00:00:00. По переднему фронту EN в часы загружается начальное время PD и начинается отсчет. Пока часы работают, выход Q = FALSE. Если EN перейдет в FALSE, CDT сбросится в начальное значение.

Описанная реализация блока RTC вызывает массу сложностей. Очевидно, часы должны продолжать свою работу при выключенном питании ПЛК. Коррекция хода часов отдельных экземпляров функционального блока RTC должна осуществляться на прикладном уровне. Многие ПЛК имеют аппаратно реализованные часы реального времени. Доступ к аппаратным часам гораздо проще организовать через прямо адресуемые ячейки памяти, чем поддерживать блок RTC.

4.2.2 Триггеры

Работу триггеров SR и RS легче всего понять по аналогии с электрическими устройствами, например, электрическим пускателем. Для переключения ему нужны 2 кнопки «ПУСК» и «СТОП». Кнопки не имеют механической фиксации, переключение выполняется коротким нажатием кнопок. Пускатель сам фиксирует свое состояние.

Именно так работают триггеры SR и RS. Их поведение отличается только при одновременном нажатии обеих кнопок. В блоке доминантной установки SR побеждает «ПУСК». В блоке доминантного сброса RS побеждает «СТОП».

SR переключатель с доминантой включения

SR			
SET1	BOOL	Q1	BOOL
RESET	BOOL		

Блок SR имеет два устойчивых состояния Ql = TRUE и Ql = FALSE. На языке ST работа блока описывается выражением:

Q1 = (NOT RESET AND Ql) OR SET1,

Вход SET1 включает выход, вход RESET – выключает. При одновременном воздействии обоих входов вход SET1 является доминантным.

RS переключатель с доминантой выключения

SR			
SET1	BOOL	Q1	BOOL
RESET	BOOL		

Блок SR имеет два устойчивых состояния Ql = TRUE и Ql = FALSE. На языке ST работа блока описывается выражением:

Ql = NOT RESET1 AND (Ql OR SET).

Вход SET включает выход, вход RESET1 – выключает. При одновременном воздействии обоих входов вход RESET1 является доминантным.

4.2.3 Детекторы импульсов

Детекторы импульсов предназначены для применения в случае, когда требуется реакция не на состояние дискретного сигнала, а на его изменение.

R_TRIG детектор переднего фронта

R_TRIG			
CLK	BOOL	Q	BOOL

Функциональный блок R_TRIG генерирует единичный импульс по переднему фронту входного сигнала.

Реализация блока требует одной внутренней переменной M: BOOL := FALSE. На языке ST блок реализуется так:

```
Q := CLK AND NOT M;
M := CLK;
```

Выход Q устанавливается в TRUE, если в предыдущем цикле вход CLK был равен FALSE, а в текущем цикле он уже имеет значение TRUE. При следующем вызове функционального блока выход сбрасывается в FALSE. Переменная M запоминает значение CLK в предыдущем цикле.

Если на вход CLK подать константу TRUE, то при перезапуске ПЛК на выходе Q будет сформирован единичный импульс. Аналогично, если вход CLK связан с аппаратурой и уже имеет значение TRUE, экземпляр R_TRIG сформирует ложный единичный импульс при первом вызове. Если бы переменная M имела начальное значение TRUE, то ложного импульса не возникало бы. В случае, когда это явление не желательно, можно создать собственный безопасный детектор фронта или применить

пустой вызов экземпляра при начальной инициализации. Такое поведение детекторов фронтов не является ошибкой, поскольку во многих случаях начальный импульс оказывается желательным.

F_TRIG детектор заднего фронта

R_TRIG			
CLK	BOOL	Q	BOOL

Функциональный блок F_TRIG генерирует единичный импульс по заднему фронту входного сигнала.

Реализация блока требует одной внутренней переменной M: BOOL := FALSE. На языке ST блок реализуется так:

```
Q := NOT CLK AND NOT M;
M := NOT CLK;
```

Из сравнения двух реализаций очевидно, что блок F_TRIG превращается в R_TRIG включением на входе инвертора NOT.

4.2.4 Счетчики

Рассмотрим реализацию счетчиков.

CTU инкрементальный счетчик

CTU			
CU	BOOL	Q	BOOL
RESET	BOOL		
PV	WORD	CV	WORD

По каждому фронту на входе CU значение счетчика (выход CV) увеличивается на 1, Выход Q устанавливается в TRUE, когда счетчик достигнет или превысит заданный PV порог. Логическая единица на входе сброса (RESET = TRUE) останавливает счет и обнуляет счетчик (CV := 0).

CTD декрементный счетчик

CTD			
CU	BOOL	Q	BOOL
LOAD	BOOL		
PV	WORD	CV	WORD

По каждому фронту на входе CD счетчик (выход CV) уменьшается на 1. Выход Q устанавливается в TRUE, когда счетчик достигнет нуля. Счетчик CV загружается начальным значением, равным PV по входу LOAD = TRUE.

CTUD инкрементный / декрементный счетчик

CTUD			
CU	BOOL	QU	BOOL
CD	BOOL	QD	BOOL
RESET	BOOL		
LOAD	BOOL		
PV	WORD	CV	WORD

По значению входа RESET = TRUE счетчик CV сбрасывается в 0. По значению входа LOAD = TRUE счетчик CV загружается значением равным PV.

По фронту на входе CU счетчик увеличивается на 1. По фронту на входе CD счетчик уменьшается на 1 (до 0).

Выход QU равен TRUE, если CV >= PV, иначе FALSE.
Выход QD равен TRUE, если CV = 0, иначе FALSE.

Вопросы для самоконтроля

1. Стандартные компоненты комплексов МЭК-программирования.
2. Назначение и примеры стандартных арифметических операторов программирования.
3. Назначение и примеры стандартных операторов битового сдвига.
4. Назначение и примеры логических битовых операторов.
5. Назначение и примеры стандартных функциональных блоков.
6. Назначение и временные диаграммы работы таймеров TP, TOF, TON, RTC.
7. Назначение и особенности работы триггеров SR и RS.
8. Назначение и особенности работы детекторов импульсов R_TRIG и F_TRIG.
9. Назначение и особенности работы счетчиков CTU, CTD и CTUD.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. *Минаев, И.Г.* Программируемые логические контроллеры : практическое руководство для начинающего инженера [Текст] / И.Г. Минаев, В.В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.
2. *Парр, Э.* Программируемые контроллеры : руководства для инженера. – М.: Бином; Лаборатория знаний, 2007. – 516 с.
3. *Костров Б.В.* Микропроцессорные системы и микроконтроллеры [Текст] / Б.В. Костров, В.Н. Ручкин. – М.: «ТехБук», 2007. – 320 с.

Дополнительная

4. *Петров, И.В.* Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-Пресс, 2004. – 246 с.
5. *Басалин, П.Д.* Архитектура вычислительных систем [Текст]: Учебник. – Нижний Новгород: Изд-во Нижегородского госуниверситета, 2003. – 243 с.
6. *Иванов, И.Ю.* Микропроцессорные устройства систем управления [Текст]: Учебное пособие / Ю.И. Иванов, В.Я. Ягай. – Таганрог: Изд-во ТРТУ, 2005. – 133.
7. *Бойко, В.И.* Схемотехника электронных систем. Микропроцессоры и микроконтроллеры [Текст]: Учебник. – БХВ-Петербург, 2004. – 464 с.
8. *Корнеев, В.В.* Современные микропроцессоры [Текст] / В.В. Корнеев, А.В. Киселёв. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003. – 448 с.

Лекция 5

5. СИСТЕМЫ АВТОМАТИЗАЦИИ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ В ЖИВОТНОВОДСТВЕ С ИСПОЛЬЗОВАНИЕМ ПЛК

5.1 Автоматизация первичной обработки молока

Процесс первичной обработки молока включает в себя операции его очистки, пастеризации и охлаждения. При этом цель пастеризации состоит в уничтожении содержащихся в молоке микроорганизмов. Последующее за пастеризацией охлаждение позволяет увеличить срок хранения продукта. Охлаждение применяют и как самостоятельную операцию при хранении молока на молочных фермах и комплексах.

Пастеризатор молока представляет собой многосекционный пластинчатый теплообменник, подогреваемый горячей водой. На практике используют разные режимы пастеризации: мгновенный (при t=85…90 ^{o}C), кратковременный (T=20 с, при t=72…76 ^{o}C) и длительный (T=300 с, при t =90 ^{o}C).

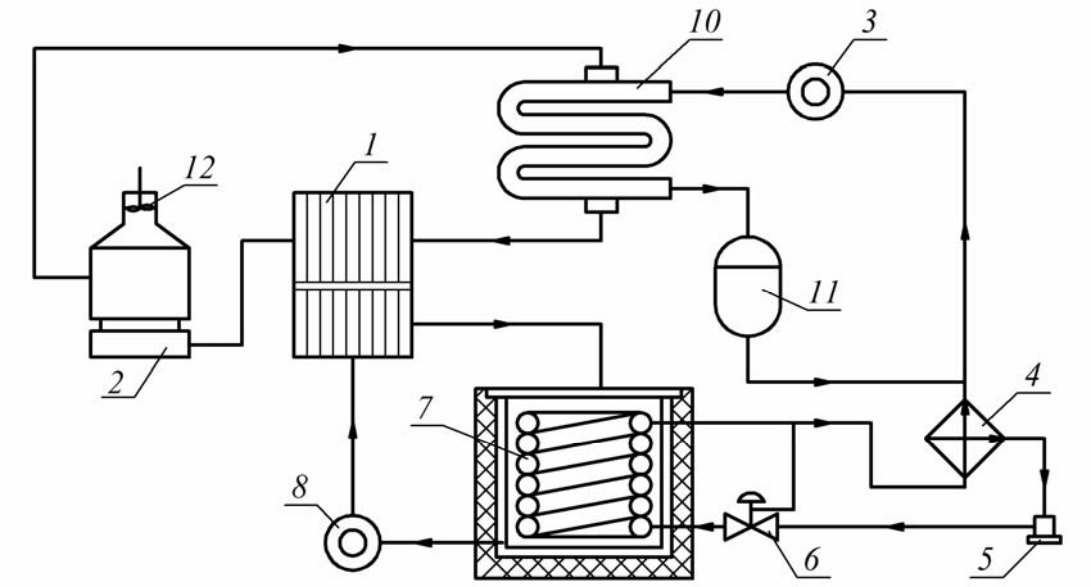

Рис.5.1 Функциональная схема водоохладительной установки

Установка для охлаждения молока (рис.5.1) работает по замкнутому циклу. Пары хладагента поступают в компрессор 3, сжимаются и попадают в конденсатор 10, где превращаются в жидкость, стекающую в ресивер 11. Из ресивера жидкий хладагент поступает в испаритель 7, проходя последовательно через теплообменник 4, фильтр-осушитель 5 и терморегулирующий вентиль 6. В терморегулирующем вентиле давление хладагента падает, он оказывается перегретым относительно нового давления и потому вскипает, отбирая теплоту у воды, орошающей поверхность испарителя. Эта вода насосом 8 перекачивается в охладитель молока 1, после которого возвращается в испаритель.

Для охлаждения воды, омывающей трубки конденсатора, применяется малогабаритная градирня 2 с вентилятором 12.

В настоящее время для управления клапанами и задвижками с электроприводом по температуре теплоносителя в водоохладительных и пастеризационных установках получили широкое распространение ПИД-регуляторы, в частности, «ОВЕН ТРМ-212» с интерфейсом RS-485, функциональная схема которого представлена на рис.5.2.

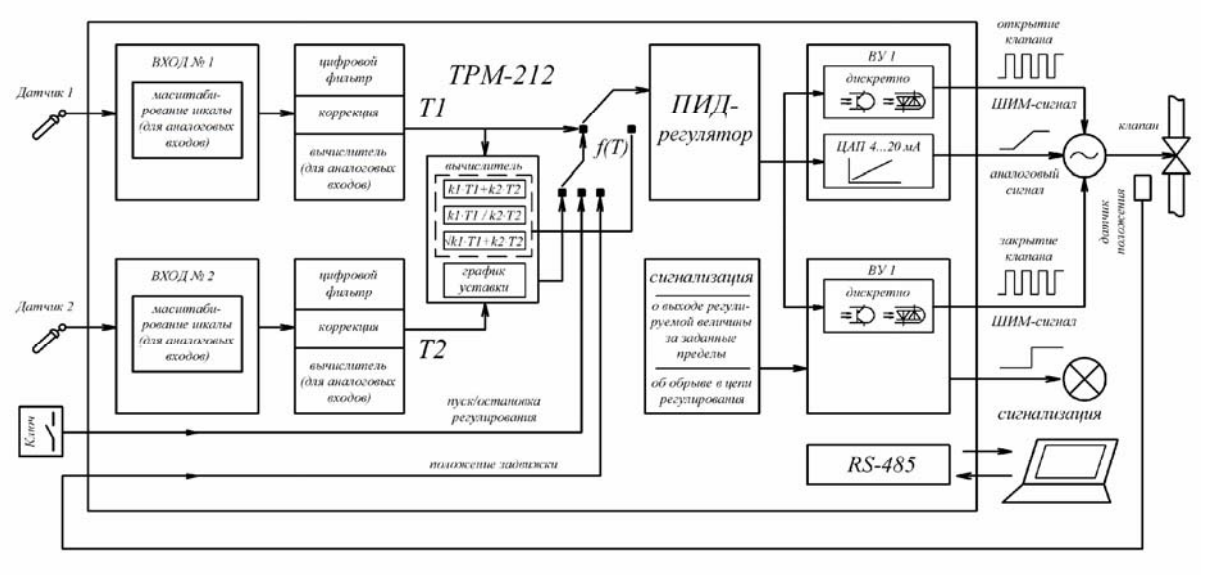

Рис.5.2 Функциональная схема САУ трехходовым клапаном

Применение данного регулятора позволяет управлять клапанами как в функции температуры, так и в функции давления, расхода, уровня, при поступлении сигнала датчика на универсальные входы. На соответствующих выходах ВУ1 и ВУ2 формируется аналоговый или ШИМ-сигнал открывающий (закрывающий) клапан или задвижку. При этом происходит постоянный контроль положения рабочего органа с помощью датчика положения.

При возникновении аварийных режимов, например, обрыв в цепи регулирования или выход регулируемой величины за заданные пределы, подается соответствующий звуковой или световой сигнал.

5.2 Автоматическое управление системами обеспечения микроклимата в животноводческих помещениях

В число параметров микроклимата, определяющих эффективность животноводства, входят температура воздуха, относительная влажность, содержание в воздухе вредных для животных компонентов, скорость движения воздуха и др.

Нормы технологического проектирования определяют температуру в коровнике на уровне 10 ^{o}C, отклонение в которой от указанного значения приводит к снижению продуктивности. При этом влияние повышенных и пониженных температур особенно заметно в условиях повышенной влажности воздуха.

Системы вентиляции выполняют с естественным и искусственным побуждением воздуха. Автоматизация систем с искусственным побуждением определяется, в том числе, конструкцией вентиляционной системы, которая может быть приточной, вытяжной и сбалансированной, когда производительность приточной и вытяжной систем одинакова. По способу раздачи воздуха в вентилируемом помещении различают равномерную, сосредоточенную и децентрализованную с помощью нескольких отопительно-вентиляционных агрегатов.

Для регулирования температуры в системах отопления с приточной вентиляцией в животноводческих помещениях в настоящее время разработано большое количество устройств. В частности, применение контроллера типа «ОВЕН ТРМ-33-Щ4» на фермах КРС позволяет повысить точность поддержания требуемой температуры воздуха в

системах приточной вентиляции с водяным или паровым калорифером на заданном уровне и снизить эксплуатационные затраты. САР, функциональная схема которой представлена на рис.5.3, обеспечивает поддержание заданной температуры приточного воздуха путем управления приводами вентилятора, жалюзи и запорно-регулирующего клапана.

К входам 1…3 подключаются температурные датчики, например, терморезисторы типа ТСМ 50М или ТСП 50П, для контроля температуры наружного воздуха $T_н$, приточного воздуха $T_{прит}$, обратной воды в контуре теплоносителя $T_{обр}$.

К входам 4…6 подключаются датчики для диагностики работоспособности системы: $C1$ – коммутирующее устройство, например, таймер, тумблер и др., для дистанционного перевода системы в дежурный режим работы; $C2$ – датчик контроля работы вентилятора для автоматического перевода системы в дежурный режим при неисправности вентилятора; $C3$ – датчик контроля протока воды через калорифер для автоматического перевода системы в режим защиты от замораживания при прекращении протока.

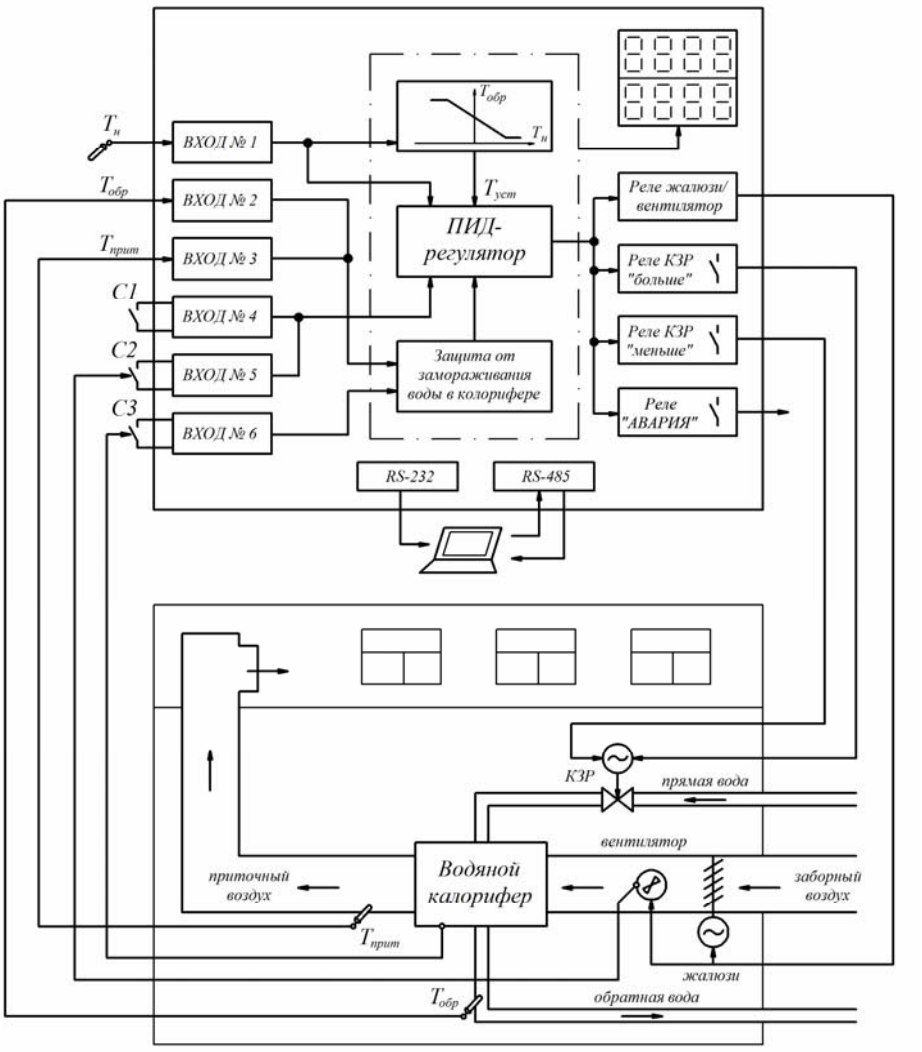

Рис.9.3 Функциональная схема САР температуры воздуха в коровнике

Температуру приточного воздуха в системе $T_{уст.прит}$, нагрев которого осуществляется теплоносителем, проходящим через калорифер, задается оператором при программировании контроллера. Регулятор по температуре уставки $T_{уст.прит}$ и по

результатам измерений и опроса входных датчиков *C*1, *C*2 и *C*3 с помощью выходных реле управляет работой вентилятора и жалюзи, а также положением запорно-регулирующего клапана для поддержания заданной температуры в системе отопления. Управление клапаном осуществляется кратковременными импульсами (ШИМ-сигналами), что позволяет поддерживать заданную температуру с высокой точностью.

Перед началом работы контроллер осуществляет прогрев калорифера, время которого устанавливается оператором при наладке исходя их эксплуатационных параметров системы. При этом для более быстрого разогрева регулятор формирует команду задержки включения вентилятора и открытия жалюзи при полном открытии клапана.

Контроллер осуществляет также защиту системы от превышения температуры обратной воды, возвращаемой в теплоцентраль и защиту калорифера от замораживания в нем воды.

Управление температурой обратной воды осуществляется в зависимости от температуры наружного воздуха в соответствии с графиком $T_{обр}=f(T_н)$ (рис.5.4), строящимся прибором автоматически по заданным оператором программирования координатам точек *A*, *B* и *C*. При превышения заданного значения температуры обратной воды $T_{обр.i} > T_{обр.\max.i}$ регулятор прерывает управление запорно-регулирующим клапаном по $T_{прит}$ и переходит на управление по сигналу рассогласования $E_i = T_{обр.i} - T_{обр.\max.i}$. После возврата $T_{обр.i}$ в допустимые пределы регулирование продолжается по $T_{прит}$.

При падении температуры приточного воздуха или температуры обратной воды ниже заданной критической температуры, либо возникновении неисправностей входных датчиков, например, обрыв или короткое замыкание, система переходит в режим защиты от замораживания воды в калорифере. При этом для максимально быстрого повышения температуры регулятор формирует команду на полное открытие клапана, выключение вентилятора и закрытие жалюзи.

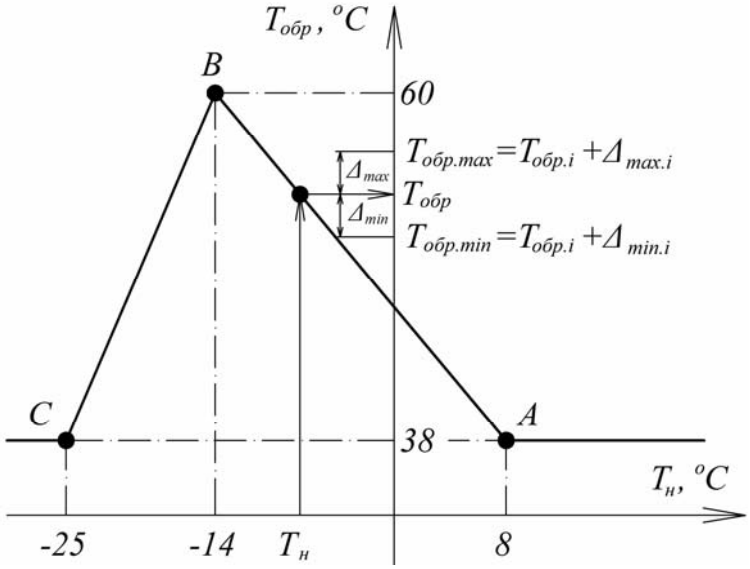

Рис.5.4 График температуры обратной воды

При превышении температурой наружного воздуха значения $T_{летн}$, заданного при программировании контроллера, система автоматически переходит на летний режим, отключение которого происходит при достижении $T_н$ значения $T_{н.}$точки *A* (рис.5.4).

Вопросы для самопроверки

1. Автоматизация первичной обработки молока.
2. Автоматическое управление системами обеспечения микроклимата в животноводческих помещениях.
3. Перспективы применения программируемых логических контроллеров для автоматизации процессов в животноводстве.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. Бородин, И.Ф. Автоматизация технологических процессов и систем автоматического управления [Текст] / И.Ф. Бородин, С.А. Андреев – М.: КолосС, 2005. – 352с.: ил. – 1000 экз. – ISBN: 5-9532-0140-0.
2. Змеев, А.Я. Проектирование систем электрификации [Текст]: учеб. пособие / А.Я. Змеев, К.М. Усанов, В.А. Каргин. – 2-е изд., перераб. и доп. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010 – 152 с. : ил. – 100 экз. – ISBN 978-5-7011-0694-7.
3. Усанов, К.М. Автоматизация технологических процессов [Текст]: учеб. пособие / К.М. Усанов, А.Я. Змеев, А.В. Волгин, В.А. Каргин, Е.А. Четвериков, Т.В. Улыбина. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010. – 108 с.: ил. – 100 экз. – ISBN 978-5-7011-0691-6.

Дополнительная

1. Усанов, К.М. Автоматика [Текст]: учеб. пособие для вузов/ К.М. Усанов, А.Я. Змеев, А.В. Волгин. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2008 – 108 с. : ил. – 100 экз. – ISBN 978-5-7011-0545-2.

Лекция 6

6. СИСТЕМЫ АВТОМАТИЗАЦИИ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ В РАСТЕНИЕВОДСТВЕ С ИСПОЛЬЗОВАНИЕМ ПЛК

6.1 САР температуры в помещении для хранения сельхоз продукции

Технология хранения сельскохозяйственной продукции включает в себя процессы подогрева, охлаждения и увлажнения для предохранения продукции от переохлаждения, самосогревания и обезвоживания. При этом автоматизация данных процессов имеет важное значение, поскольку нарушения в технологии хранения оборачиваются большими потерями продукции. Сохранность продукта зависит главным образом от температурного режима в хранилище.

Функциональная схема САР температуры в помещении для хранения сельхоз продукции представлена на рис.6.1. Применение контроллера приточной вентиляции типа «ОВЕН ТРМ-173» (рис.6.2) позволяет поддерживать требуемую температуру в помещении для хранения продукции с заданной степенью точности.

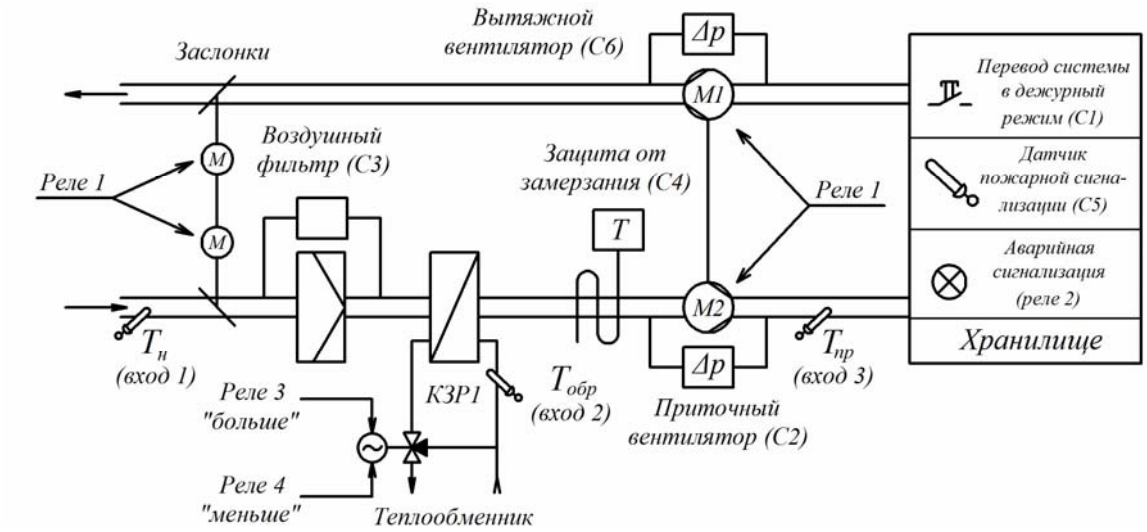

Рис.6.1 Функциональная схема САР температуры в помещении для хранения сельскохозяйственной продукции

Микропроцессорное устройство имеет 7 универсальных входов, к которым можно подключать датчики различных типов, например, термосопротивления ТСП 50П, ТСМ 60М; термопары ТХК (L), ТХА (К); датчики с унифицированным выходным сигналом тока 0…5 мА или напряжения 0…1 В; датчики положения задвижки – резистивные или токовые.

Дискретные входы *С*1…*С*6 диагностируют исправность системы вентиляции и переключают режимы, и предназначены для подключения датчиков: *С*1 – коммутирующее устройство для дистанционного перевода системы в дежурный режим; *С*2 – датчик контроля исправности приточного вентилятора по потоку воздуха; *С*3 – датчик контроля засорения фильтра приточного (вытяжного) вентилятора; *С*4 – датчик перевода системы в режим защиты калорифера от замерзания; *С*5 – датчик пожарной сигнализации; *С*6 – датчик контроля исправности вытяжного вентилятора (рис.6.2).

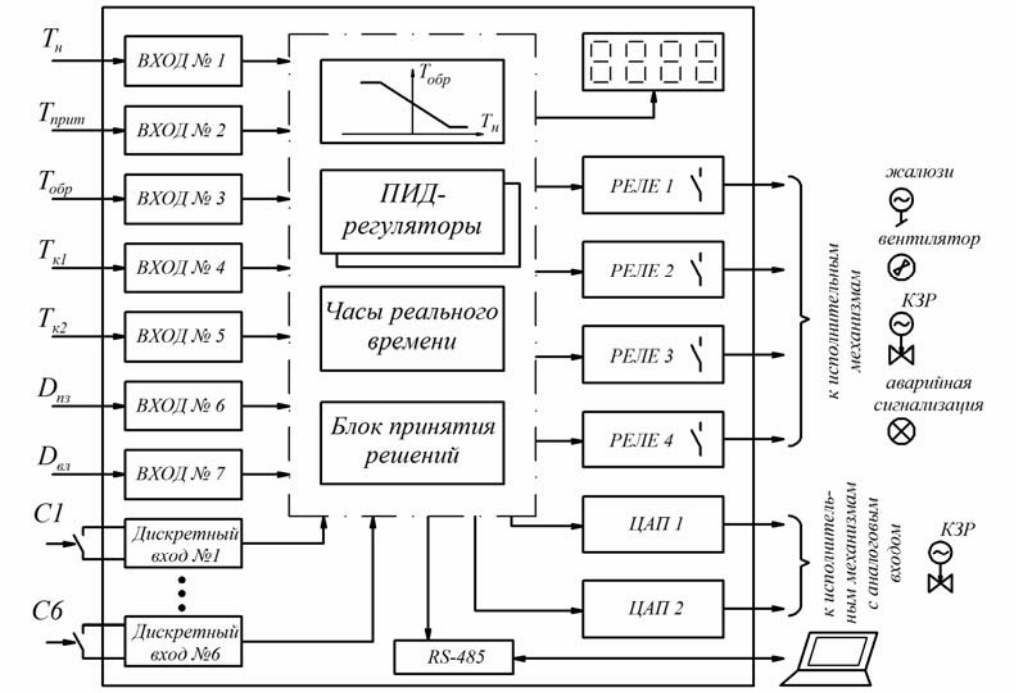

Рис.6.2 Функциональная схема контроллера приточной вентиляции

Для управления вентилятором, жалюзи, калорифером и аварийной сигнализацией контроллер оснащен встроенными выходными элементами: электромагнитными реле и цифро-аналоговыми преобразователями (ЦАП) для управления исполнительным механизмом с аналоговым управлением. При этом регулятор может управлять задвижками как с датчиком положения, так и без него по математической модели, заложенной программой.

Благодаря использованию в контроллере ПИД-закона регулирования и современному алгоритму автонастройки достигается высокая точность поддержания температуры приточного воздуха и обратной воды. Кроме того, используются несколько контуров ПИД-регулирования, что позволяет гибко настраивать прибор для работы в разных режимах.

При организации хранения картофеля и овощей температура в помещении не должна превышать 2…5 ^{o}C. В этом случае для обеспечения заданного значения параметра в осенне-зимний период, применяют регулятор скорости вращения вентилятора в зависимости от температуры в помещении (рис.6.2).

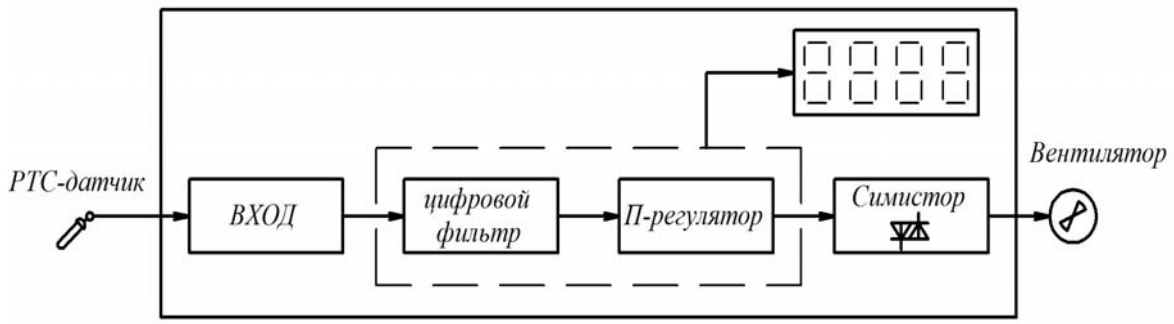

Рис.6.3 Функциональная схема регулятора скорости вращения вентилятора в зависимости от температуры

Температура в хранилище T измеряется с помощью датчика температуры с положительным коэффициентом передачи (PTC-датчик), например, полупроводникового термистора, и подается на П-регулятор, который, в свою очередь, поддерживает заданное значение параметра $T_{зад}$ путем изменения скорости вращения $\omega_в$ вентилятора. При этом чем выше температура в хранилище, тем быстрее вращается вентилятор (рис.6.4). Крутизна характеристики определяется полосой пропорциональности Δ регулятора (дифференциалом) и задается оператором при программировании. Таким образом, температура воздуха в помещении поддерживается автоматически за счет изменения количества теплого внутреннего воздуха, подмешиваемого к холодному наружному.

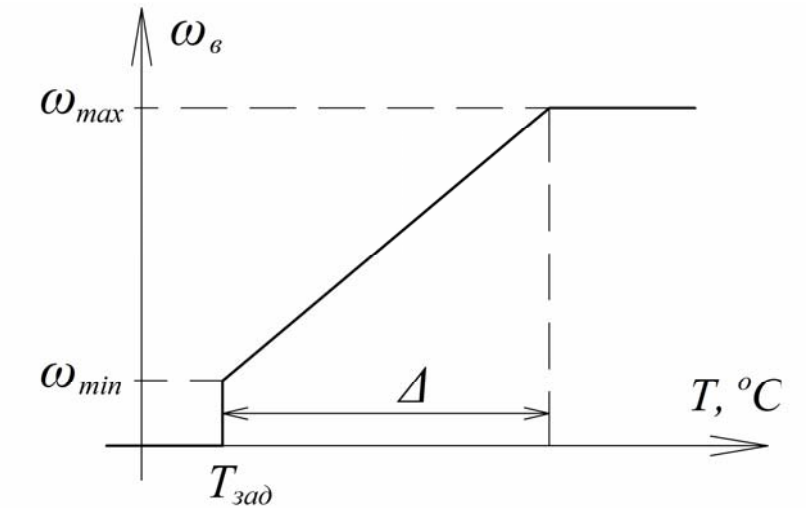

Рис.6.4 График зависимости скорости вращения вентилятора от температуры

Снижение температуры до значения уставки приводит к остановке вентилятора. В случае дальнейшего уменьшения контролируемого параметра автоматически включаются рециркуляционно-отопительные агрегаты.

6.2 САР температуры в холодильном агрегате

При хранении мясной и молочной продукции важное значение имеет контроль температуры в холодильных и морозильных агрегатах. В настоящее время на перерабатывающих предприятиях применяют блок управления средне и низкотемпературными машинами с автоматическим размораживанием типа «ОВЕН ТРМ-974». Функциональная схема САР температуры в холодильном агрегате представлена на рис.6.5.

Контроллер ТРМ-974 имеет два входа для подключения термодатчиков с положительным коэффициентом передачи для измерения температур в холодильной камере и воздухоохладителя. Выходные реле управляют процессами в холодильной камере: реле 1 – компрессором, реле 2 – вентилятором, реле 3 – нагревателем.

Регулирование температуры в холодильных машинах может производиться в двух режимах: режим «термостат» и режим «набор холода».

Температуру в камере в режиме «термостат» (рис.6.6) определяют параметры $T_{зад}$ – контрольная точка и $T_{диф}$ – дифференциал. Для поддержания температуры в холодильной установке контроллер управляет работой компрессора и вентилятора.

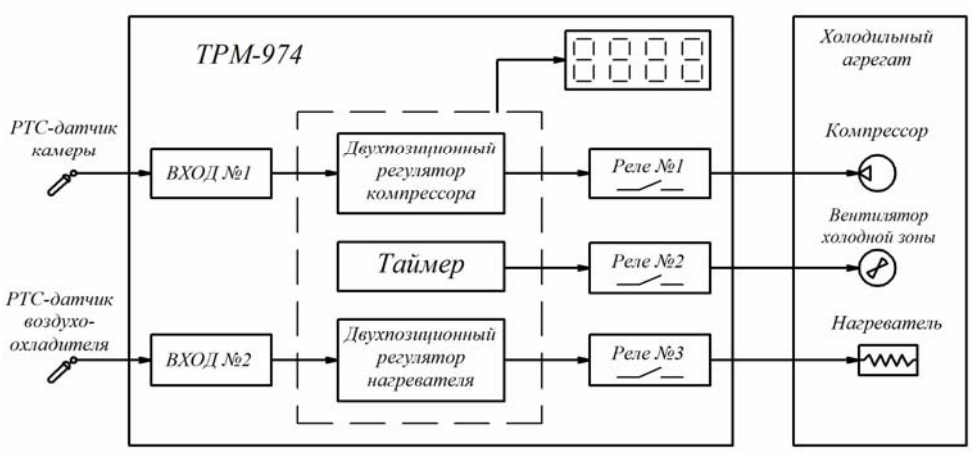

Рис.6.5 Функциональная схема САР температуры в холодильном агрегате

При достижении температуры в холодильной камере до значения $T_{зад} + T_{диф}$ включается компрессор (реле 1) и работает до тех пор, пока температура снизится до значения $T_{зад}$. При этом вентилятор (реле 2) по выбору оператора может работать как непрерывно, так и периодически вместе с компрессором.

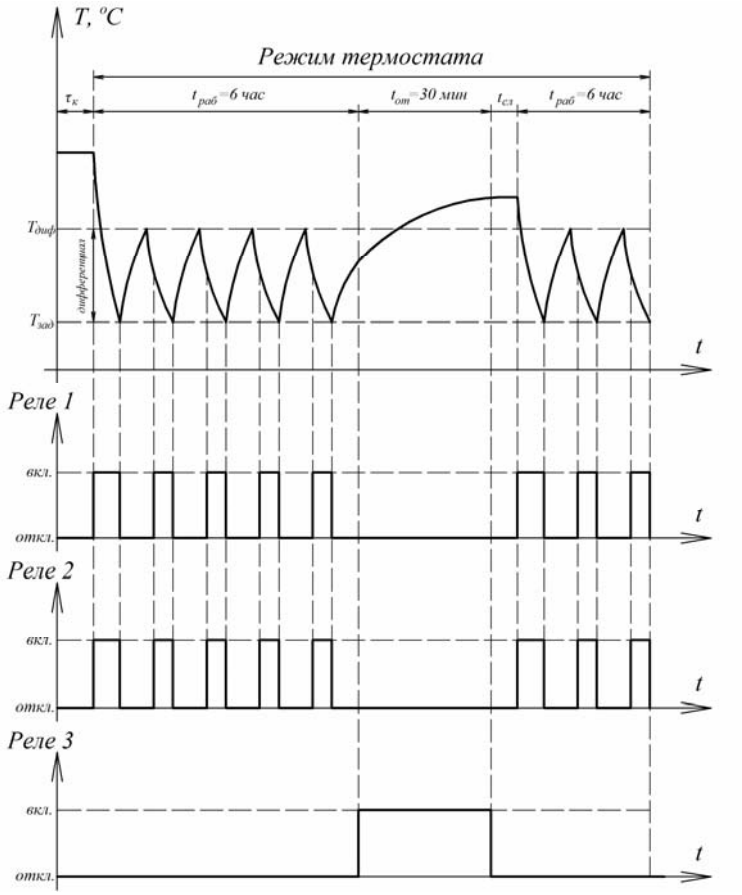

Рис.6.6 Регулирование температуры в холодильной камере в режиме «термостат»

Для автоматического управления процессом оттайки холодильной камеры контроллер программируют либо по времени (1…99 ч.), либо по времени наработки

компрессора. При этом оттайку можно производить специальным нагревателем (реле 3) при выключенном компрессоре. В ТРМ-974 также предусмотрена возможность управления сливом конденсата и установка времени задержки включения вентилятора по окончании процесса оттайки.

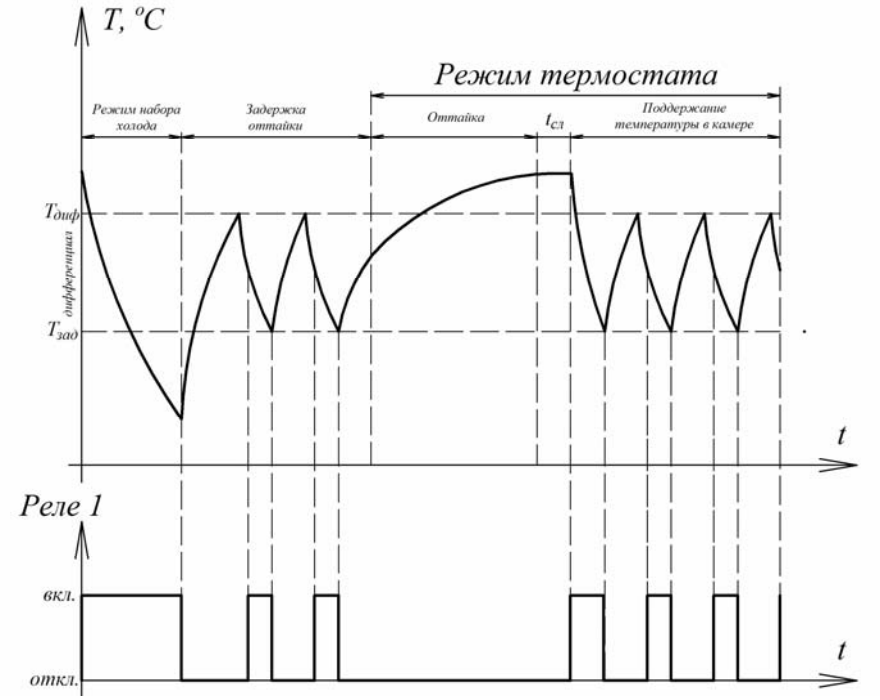

Рис.6.7 Регулирование температуры в холодильной камере в режиме «набора холода»

Режим «набора холода» (рис.6.7) предназначен для быстрого охлаждения холодильной камеры, заполненной новым (теплым) продуктом. Продолжительность набора холода задается в диапазоне от 1 до 24 часов, когда компрессор принудительно включен (реле 1). При этом обеспечивается задержка оттайки после набора холода, по окончании которой прибор автоматически переводится в режим «термостат».

Вопросы для самоконтроля

1. САР температуры в помещении для хранения сельхоз продукции.
2. САР температуры в холодильном агрегате.
3. Перспективы применения программируемых логических контроллеров для автоматизации процессов в растениеводстве.

СПИСОК ЛИТЕРАТУРЫ

Основная

1. Бородин, И.Ф. Автоматизация технологических процессов и систем автоматического управления [Текст] / И.Ф. Бородин, С.А. Андреев – М.: КолосС, 2005. – 352с.: ил. – 1000 экз. – ISBN: 5-9532-0140-0.
2. Змеев, А.Я. Проектирование систем электрификации [Текст]: учеб. пособие / А.Я. Змеев, К.М. Усанов, В.А. Каргин. – 2-е изд., перераб. и доп. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010 – 152 с. : ил. – 100 экз. – ISBN 978-5-7011-0694-7.

3. Усанов, К.М. Автоматизация технологических процессов [Текст]: учеб. пособие / К.М. Усанов, А.Я. Змеев, А.В. Волгин, В.А. Каргин, Е.А. Четвериков, Т.В. Улыбина. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010. – 108 с.: ил. – 100 экз. – ISBN 978-5-7011-0691-6.

Дополнительная

1. Усанов, К.М. Автоматика [Текст]: учеб. пособие для вузов/ К.М. Усанов, А.Я. Змеев, А.В. Волгин. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2008 – 108 с. : ил. – 100 экз. – ISBN 978-5-7011-0545-2.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Минаев, И.Г.* Программируемые логические контроллеры : практическое руководство для начинающего инженера [Текст] / И.Г. Минаев, В.В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.
2. *Парр, Э.* Программируемые контроллеры : руководства для инженера. – М.: Бином; Лаборатория знаний, 2007. – 516 с.
3. *Костров Б.В.* Микропроцессорные системы и микроконтроллеры [Текст] / Б.В. Костров, В.Н. Ручкин. – М.: «ТехБук», 2007. – 320 с.
4. *Петров, И.В.* Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования. – М.: СОЛОН-Пресс, 2004. – 246 с.
5. *Басалин, П.Д.* Архитектура вычислительных систем [Текст]: Учебник. – Нижний Новгород: Изд-во Нижегородского госуниверситета, 2003. – 243 с.
6. *Иванов, И.Ю.* Микропроцессорные устройства систем управления [Текст]: Учебное пособие / Ю.И. Иванов, В.Я. Ягай. – Таганрог: Изд-во ТРТУ, 2005. – 133.
7. *Бойко, В.И.* Схемотехника электронных систем. Микропроцессоры и микроконтроллеры [Текст]: Учебник. – БХВ-Петербург, 2004. – 464 с.
8. *Корнеев, В.В.* Современные микропроцессоры [Текст] / В.В. Корнеев, А.В. Киселёв. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2003. – 448 с.
9. Бородин, И.Ф. Автоматизация технологических процессов и систем автоматического управления [Текст] / И.Ф. Бородин, С.А. Андреев – М.: КолосС, 2005. – 352с.
10. Змеев, А.Я. Проектирование систем электрификации [Текст]: учеб. пособие / А.Я. Змеев, К.М. Усанов, В.А. Каргин. – 2-е изд., перераб. и доп. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010 – 152 с.
11. Усанов, К.М. Автоматизация технологических процессов [Текст]: учеб. пособие / К.М. Усанов, А.Я. Змеев, А.В. Волгин, В.А. Каргин, Е.А. Четвериков, Т.В. Улыбина. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2010. – 108 с.
12. Усанов, К.М. Автоматика [Текст]: учеб. пособие для вузов/ К.М. Усанов, А.Я. Змеев, А.В. Волгин. – Саратов: ФГОУ ВПО «Саратовский ГАУ», 2008 – 108 с.

СОДЕРЖАНИЕ