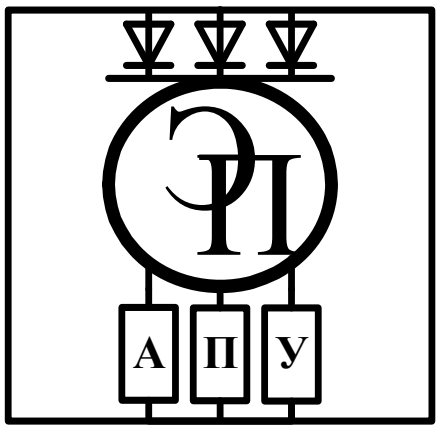
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Электропривод и автоматизация промышленных установок»

СИСТЕМЫ ПРОГРАММНОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИХ КОМПЛЕКСОВ

Методические рекомендации к лабораторным работам для студентов специальности 1-53 01 05 «Автоматизированные электроприводы» дневной и заочной форм обучения



Могилев 2022

УДК 004.4
ББК 32.295
С40

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Электропривод и автоматизация промышленных установок» «30» августа 2022 г., протокол № 1

Составитель ст. преподаватель А. В. Янкович

Рецензент канд. техн. наук, доц. С. В. Болотов

В методических рекомендациях к лабораторным работам для студентов специальности 1-53 01 05 «Автоматизированные электроприводы» изложена методика разработки программ для систем управления на базе программируемых логических контроллеров.

Учебно-методическое издание

СИСТЕМЫ ПРОГРАММНОГО УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИХ КОМПЛЕКСОВ

Ответственный за выпуск	С. М. Фурманов
Корректор	Т. А. Рыжикова
Компьютерная верстка	М. М. Дударева

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл.печ.л. . Уч.-изд.л. .Тираж 99 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

1 Лабораторная работа № 1. Программирование логических контроллеров на языке SFC в системе CoDeSys

Цель работы: изучение принципов составления прикладных программ для промышленных логических контроллеров (ПЛК) на языке SFC пакета CoDeSys.

Общие сведения

Язык последовательных функциональных схем (SFC – Sequential Function Chart) является одним из пяти языков программирования контроллеров по стандарту МЭК 61131-3. Он ориентирован на выполнение операций в определенной последовательности, которая задается моментами времени или событиями.

Теоретической основой языка SFC является теория конечных автоматов, используемая для формализации состояний сложных процессов управления. Теория конечных автоматов опирается на различные графические модели описания состояний. Одной из наиболее известных является модель, предложенная К. Петри, получившая название «сети Петри», или диаграммы состояний.

Язык SFC имеет те же возможности, что и диаграммы состояний, и является наиболее подходящим средством для описания динамических моделей.

Строго говоря, SFC не является языком программирования, по сути, это вспомогательное средство для структурирования программ, состоящих из большого числа программных единиц (программ, функциональных блоков, функций). Язык SFC обеспечивает параллельность выполнения программ, установление и контроль состояния выполняемых процессов, синхронизацию по приему и обработке данных.

Язык SFC предназначен для описания системы управления на самом верхнем уровне абстракции, например, в терминах «Старт», «Выполнение этапа № 1», «Выполнение этапа № 2». Язык SFC может быть использован также для программирования отдельных функциональных блоков, если алгоритм их работы естественным образом описывается с помощью понятий состояний и переходов.

Базовым блоком схемы SFC является шаг. Шаг эквивалентен понятию состояния. Полный процесс управления последовательным процессом является последовательным или параллельным объединением шагов. Программа состоит из шагов и условий переходов. Шаги показываются на схеме прямоугольниками, условия переходов – перечеркивающей линией.

Программа выполняется сверху вниз. Начальный шаг на схеме показывается в виде двойного прямоугольника. Условия переходов записываются рядом с их обозначениями. Каждый шаг программы может представлять собой реализацию сложного алгоритма, написанного на одном из языков МЭК 61131-3.

На рисунке 1.1 представлен пример схемы SFC.

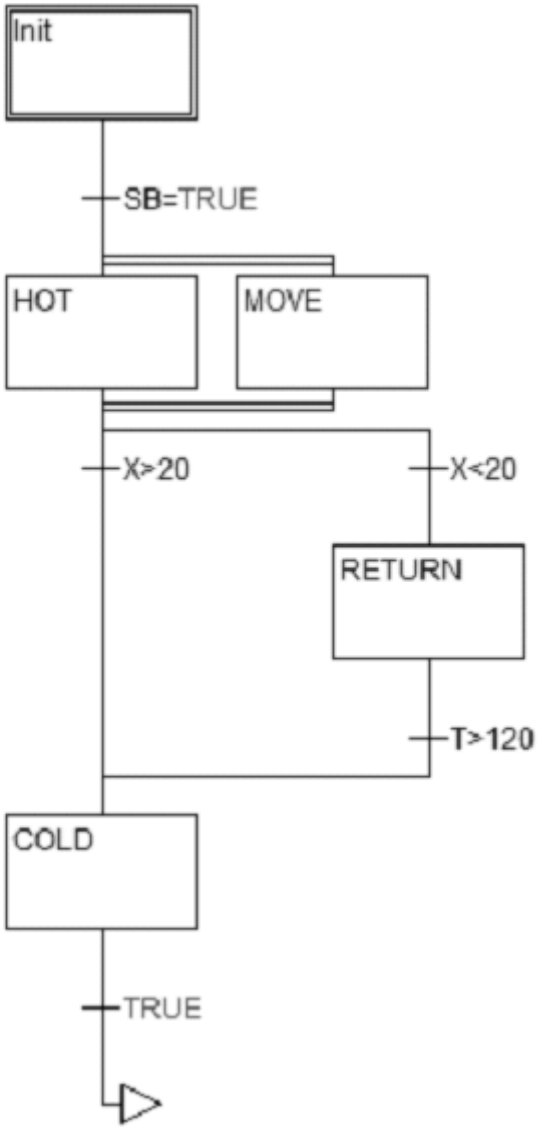


Рисунок 1.1 – Пример SFC-схемы

Благодаря графическому представлению язык SFC максимально прост в использовании и изучении. Вместе с тем, он является наглядным средством представления логики управления.

Использование языка SFC позволяет снизить количество ошибок на ранних этапах проектирования систем управления.

Основные элементы SFC-программ

Шаги.

Любая SFC-программа состоит из элементов, представляющих шаги и условия переходов (рисунок 1.2). Шаги показываются в программе прямоугольниками. Реальная работа шага (действия) описывается в отдельном окне системы программирования и не отражается на SFC-диаграмме. О назначении шага SFC говорит только его название или, если этого недостаточно, краткое текстовое описание (комментарий).

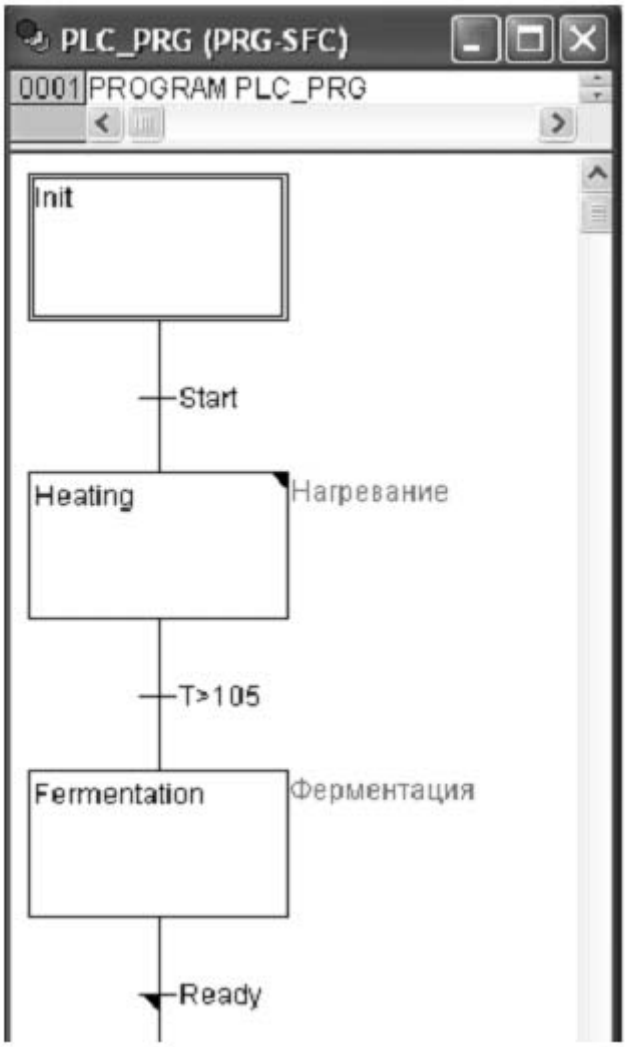


Рисунок 1.2 – Шаги и переходы

Существует два вида шагов:

– шаг простого типа (упрощенный SFC) может включать единственное действие;

– МЭК-шаг (стандартный SFC) связан с произвольным числом действий или логических переменных.

Связанные действия располагаются с правой стороны от шага.

Шаги на схеме могут быть пустыми, что не вызывает ошибки при компиляции проекта. Графический флажок (небольшой треугольник в верхнем углу шага) показывает, пустой шаг или нет.

Пустые шаги являются нормой при применении программирования сверху вниз, характерного для SFC. Определить действия, соответствующие шагу, можно в любое время. Задачей пустого шага в законченном проекте является ожидание перехода.

Каждая SFC-схема начинается с шага, выделенного двойной рамкой. Наименование начального шага может быть произвольным (по умолчанию Init). Начальный шаг присутствует обязательно, хотя и может быть пустым.

Переходы.

Ниже шага на соединительной линии присутствует горизонтальная черта обозначающая переход (см. рисунок 1.2). Условием перехода может служить логическая переменная, логическое выражение, константа.

Переход выполняется при соблюдении двух условий:

1) переход разрешен (соответствующий ему шаг активен);

2) условие перехода имеет значение TRUE.

Простые условия отображаются непосредственно на диаграмме справа от черты, обозначающий переход.

Для громоздких условий применяется другой подход. Вместо условия на диаграмме записывается только идентификатор перехода. Само же условие описывается в отдельном окне с применением языка IL, ST, LD или FBD.

На рисунке 1.3 показано возможное представление перехода Ready на языке LD.

Признаком того, что идентификатор перехода на диаграмме является отдельно реализованным условием, а не простой логической переменной, служит закрашенный угол перехода (см. рисунок 1.2).

В качестве условия перехода может быть задана логическая константа. Если задано TRUE, то шаг будет выполнен однократно, за один рабочий цикл, далее управление перейдет к следующему шагу. Если задано условие FALSE, то шаг будет выполняться бесконечно.

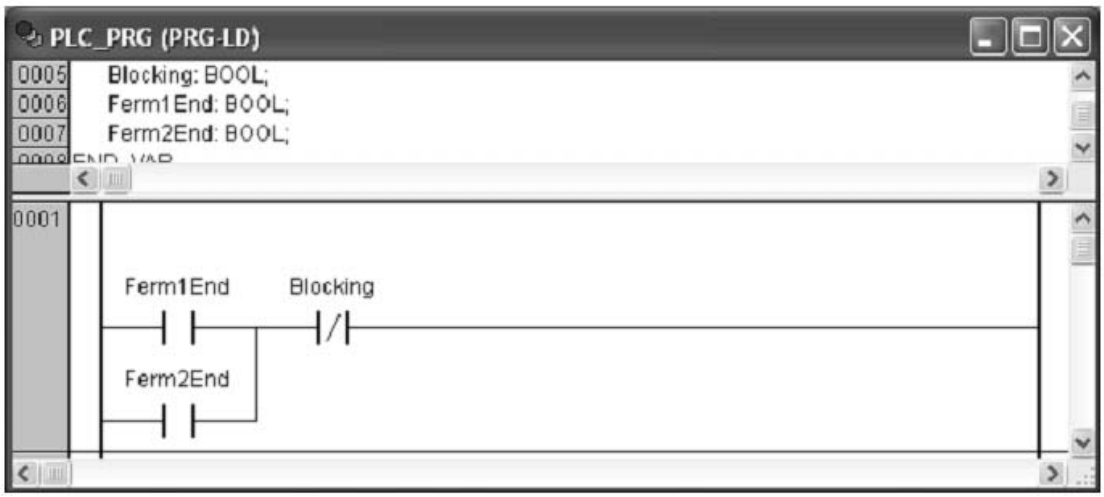


Рисунок 1.3 – Переход Ready на языке LD

Действие.

Действие может содержать команды (инструкции) на языках IL, ST, FBD, LD или SFC.

При использовании простых шагов действие всегда связывается с этим шагом. Помимо основного действия, простой шаг может включать одно входное и одно выходное действие.

Действия МЭК-шагов показаны в Организаторе Объектов, непосредственно под вызывающей их программой. Редактирование действия запускается двойным щелчком мыши или клавишей Enter. Новые действия добавляются командой главного меню [Project] (Проект) – [Add Action] (Добавить действие) (рисунок 1.4). Одному МЭК-шагу можно сопоставить до девяти действий.

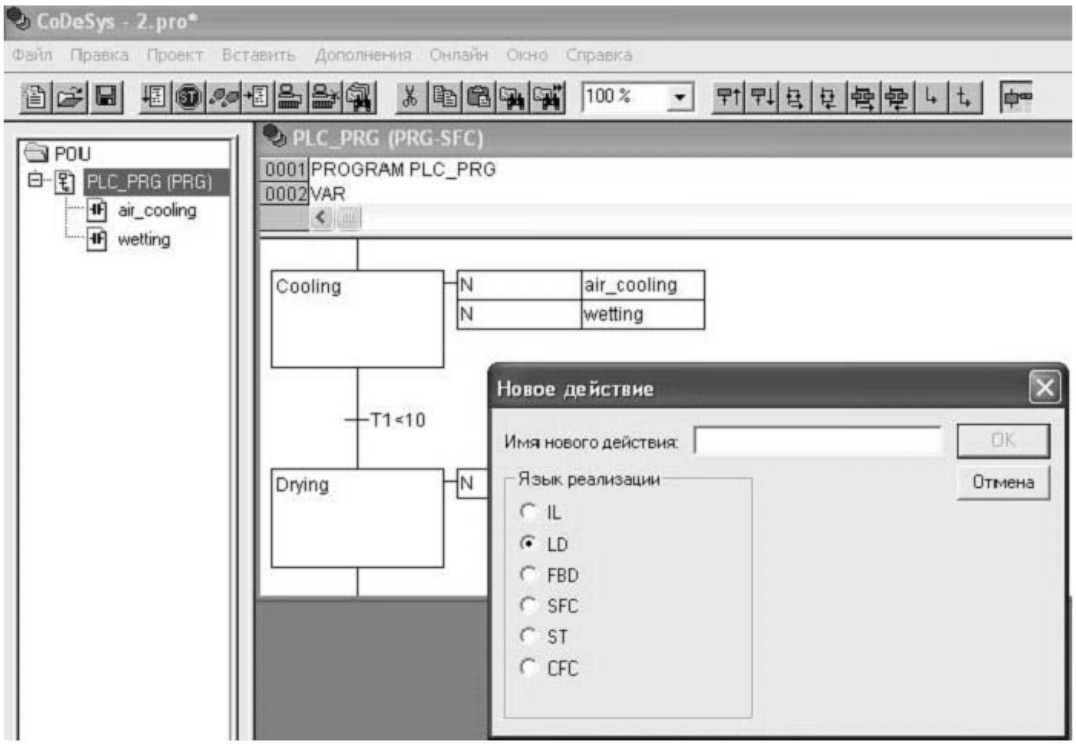


Рисунок 1.4 – Добавление нового действия

Входные и выходные действия.

Весьма вероятен случай, когда определенные действия нужно выполнить в шаге только один раз. Например, включить нагрев в начале активности шага и выключить при переходе на другой шаг. С этой целью в простой шаг можно добавить входное и выходное действие.

Входное действие обозначается сегментом E (Entry) в нижнем левом углу прямоугольника шага и выполняется однократно при активизации шага.

Выходное обозначается сегментом X (eXit) в нижнем левом углу прямоугольника шага (рисунок 1.5). Выходное действие выполняется однократно при завершении работы шага.

Команда [Insert] (Вставка) – [Add Entry-Action] (Добавить входное действие) добавляет входное действие в шаг. Такое действие выполняется только один раз при активации шага. Команда [Insert] (Вставка) – [Add Exit-Action] (Добавить выходное действие) добавляет выходное действие в шаг. Такое действие выполняется только раз при деактивации шага.

Входные и выходные действия могут описываться на любом языке. Для того чтобы отредактировать входное или выходное действие, надо дважды щелкнуть мышкой в соответствующем углу шага.

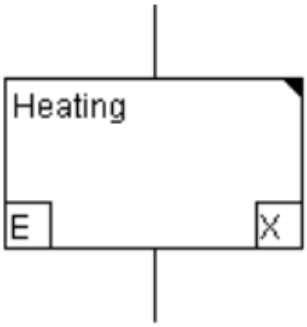


Рисунок 1.5 – Шаг с входным и выходным действиями

Активный шаг.

После вызова SFC-программы начальный шаг (шаг, выделенный двойной рамкой) выполняется первым.

Шаг, выполняемый в данный момент, называется активным. Действия, связанные с активным шагом, выполняются один раз в каждом рабочем цикле контроллера. В режиме онлайн активные шаги выделяются синим цветом.

Следующий за активным шагом шаг станет активным, только когда условие перехода к этому шагу примет значение TRUE.

В каждом цикле будут выполнены действия, содержащиеся в активных шагах. Далее проверяются условия перехода, и, возможно, уже другие шаги становятся активными, но выполняться они будут уже в следующем цикле.

Параллельные ветви.

Несколько ветвей SFC могут быть параллельными (рисунок 1.6). Признаком параллельных ветвей на схеме является двойная горизонтальная линия. Каждая параллельная ветвь начинается и заканчивается шагом. То есть условие входа в параллельность всегда одно, условие выхода тоже одно на всех.

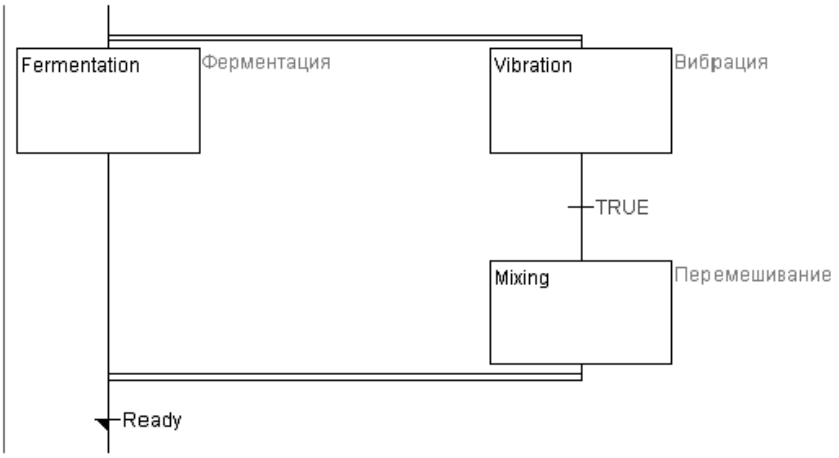


Рисунок 1.6 – Параллельные ветви

Параллельные ветви выполняются теоретически одновременно. В реальности это означает, что в одном рабочем цикле, слева направо. Условие перехода, завершающее параллельность, проверяется только в случае, если в каждой параллельной ветви активны последние шаги.

В примере на рисунке 1.6 шаг Fermentation будет выполнен однократно, далее шаги Vibration и Mixing будут работать параллельно до выполнения условия Ready.

Альтернативные ветви.

Несколько ветвей SFC могут быть альтернативными ветвями. Признаком альтернативных ветвей на схеме является одинарная горизонтальная линия (рисунок 1.7). Каждая альтернативная ветвь начинается и заканчивается собственным условием перехода. Проверка альтернативных условий выполняется слева направо. Если верное условие найдено, то прочие альтернативы не рассматриваются. В альтернативных ветвях всегда работает только одна из ветвей, поэтому ее окончание и будет означать переход к следующему за альтернативной группой шагу.

В примере на рисунке 1.7 альтернатива Stop оценивается первой. Шаги Move_Dwn и Mooe_Up имеют шанс стать активными, только если Stop равен FALSE.

При создании альтернативных ветвей желательно задавать взаимоисключающие условия. В этом случае вероятность допустить ошибку при анализе или в процессе доработки диаграммы значительно ниже.

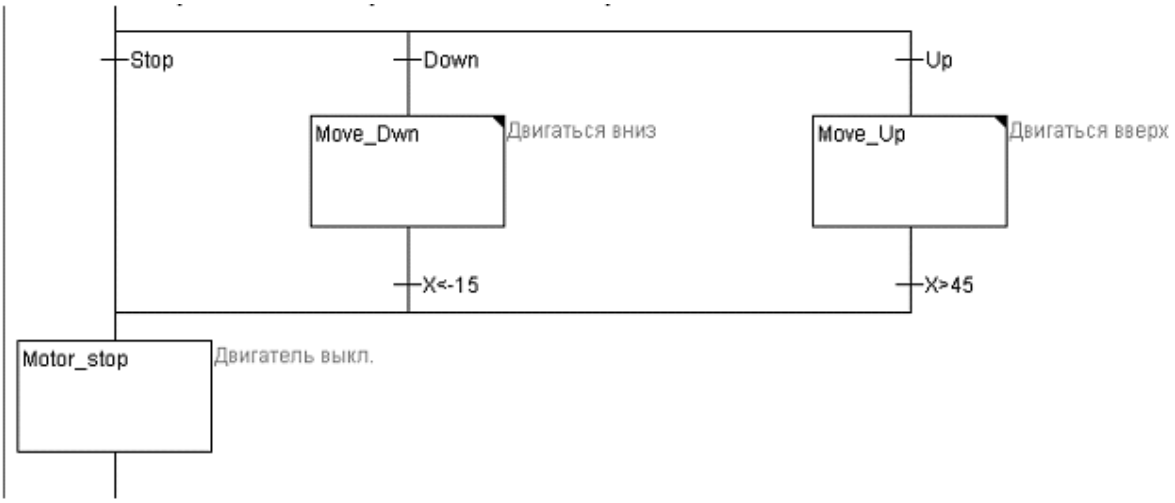


Рисунок 1.7 – Альтернативные ветви

Переход на произвольный шаг.

В общем случае SFC-программа выполняется сверху вниз. Стандартом МЭК допускается создание переходов на произвольный шаг. Для этого применяются соединительные линии с промежуточными стрелками или поименованные переходы. То есть переход выполняется на шаг, имя которого указано под стрелкой.

В примере, показанном на рисунке 1.8, шаги Move_Dwn и Move_Up последовательно активируют друг друга.

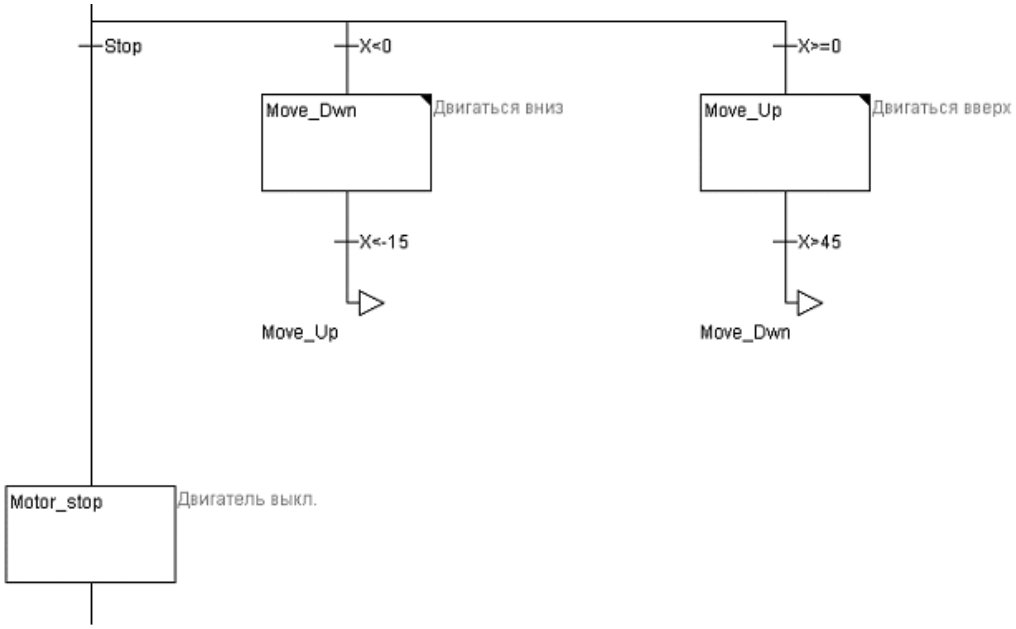


Рисунок 1.8 – Перекрестные переходы

Необходимо отметить, что условие Stop при этом проверяться не будет, шаги Move_Dwn и Move_Up соединены в логическое кольцо, имеющее два варианта входа, но ни одной возможности выхода. Маркер активности будет перемещаться исключительно в этом кольце.

Упрощенный SFC.

Упрощенная реализация SFC (easy mode SFC) позволяет получить диаграммы компактнее и часто проще для понимания. Однако возможности упрощенной реализации несколько у́же – нельзя включать и выключать действия в разных шагах и управлять активностью действий по времени.

Действия могут быть трех классов: текущее (основное), входное и выходное. Графически действия на диаграмме никак не отображаются, их редактирование выполняется в отдельных окнах. В упрощенной реализации действия принадлежат шагу. То есть действие нельзя вызвать из другого шага или откуда-либо еще. Можно считать, что каждый прямоугольник шага при его увеличенном рассмотрении содержит три раздела, соответствующие трем возможным действиям. Если шаг удалить, то и все его действия будут утрачены. Такие действия не требуют отдельных идентификаторов и называются по именам шагов.

Для создания нового или редактирования существующего действия в CoDeSys достаточно щелкнуть мышкой по прямоугольнику шага. Это приведет к открытию соответствующего редактора или вызову диалога создания нового действия, если шаг еще не описан.

На рисунке 1.9 показан момент определения действия шага Fermentation (Ферментация).

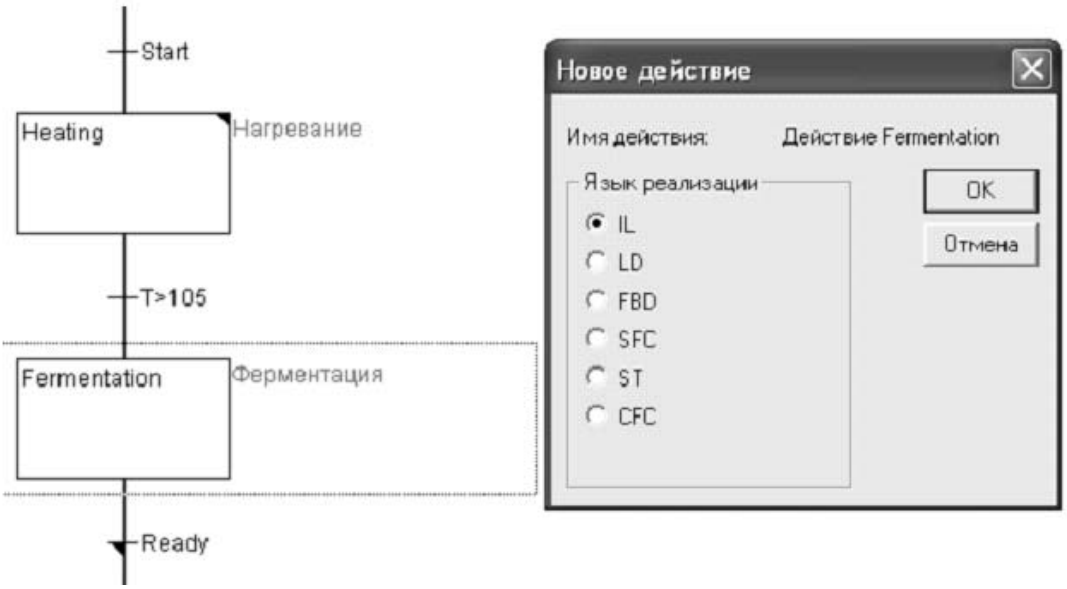


Рисунок 1.9 – Описание действия шага

Шаги, содержащие действие, на схеме отличаются тем, что верхний правый угол прямоугольника закрашен. Пока шаг активен, текущее действие будет выполняться один раз в каждом рабочем цикле.

Стандартный SFC.

Вышеописанная техника проектирование прикладной программы настраивает на то, что изначально определяются шаги, которые наполняются определенным содержимым в процессе работы над проектом.

При применении МЭК-шагов подход к проектированию несколько иной.

Сначала определяются действия (виды работ), которые должна выполнять система, а затем уже составляется диаграмма, в которой определяется их порядок и взаимосвязь. Каждое действие сопоставляется одному или нескольким шагам. Причем вполне возможно, что некоторое действие должно запускаться в одном шаге и останавливаться в другом.

Также возможно, что начатое действие должно закончить свою работу вообще независимо от текущего шага. Например, начав движение, кабина лифта должна как минимум доехать до ближайшего этажа и выпустить пассажиров, даже если дана команда на окончание работы.

В отличие от упрощенной реализации языка SFC МЭК шаги могут включать несколько действий (до девяти). Действия МЭК-шагов описываются отдельно от них и могут неоднократно использоваться в пределах данной программы, для чего их надо связать с шагом с помощью команды главного меню [Extras] (Дополнения)– [Associate action] (Связать действие).

Действия МЭК показываются на SFC-диаграмме в виде прямоугольников, расположенных справа от шага и привязанных к нему графически. Пример шагов, содержащих действия, показан на рисунке 1.10.

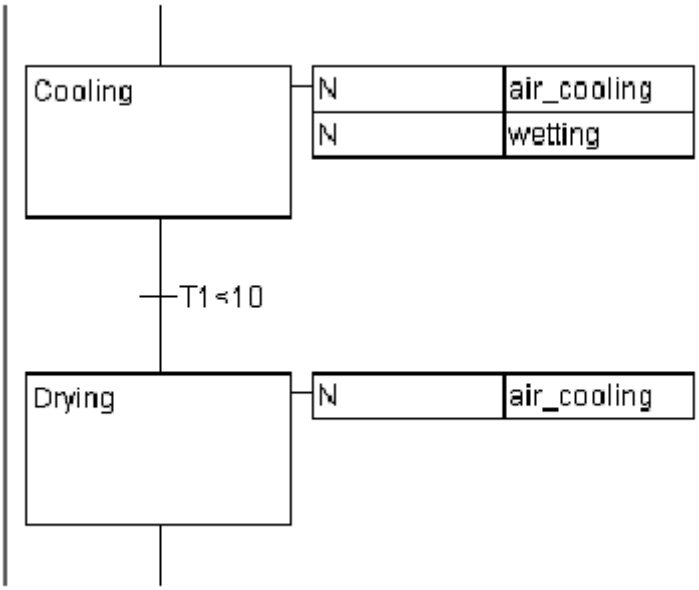


Рисунок 1.10 – Действия в стандартном SFC

Весьма важным здесь является то, что одно и то же действие можно многократно использовать в разных шагах. Так, в данном примере шаги Cooling (охлаждение) и Drying (сушка) используют действие air cooling (воздушное охлаждение).

В отличие от описанных выше упрощенных действий, действия МЭК не принадлежат конкретному шагу, а являются самостоятельными программными элементами SFC-компонента.

С помощью так называемых классификаторов действия могут активироваться и деактивироваться, возможно, с задержкой времени. Например: действие может продолжать работу, даже если запустивший его шаг утратил активность.

Действие, связанное с МЭК-шагом, описывается справа от него в блоке, состоящем из двух частей. Левая часть этого блока содержит классификатор, возможно, с константой времени, а правая часть содержит имя действия.

В режиме онлайн все активные действия выделяются синим цветом подобно активным шагам, благодаря чему легко проследить ход выполнения процесса после каждого управляющего цикла.

Для того чтобы использовать шаги с МЭК-действиями, необходимо установить в главном меню опцию [Extras] (Дополнения) – [Use IEC-Steps] (Использовать МЭК-шаги).

Идентификаторы действий должны быть уникальны в пределах SFC-программы и не должны совпадать с идентификаторами шагов и переходов.

Классификаторы действий.

Прямоугольник, отображающий действие, содержит в левой части специальное поле – классификатор (см. рисунок 1.10). Классификатор определяет способ влияния активного шага на данное действие.

Основные виды классификаторов приведены в таблице 1.1.

Таблица 1.1 – Виды классификаторов действий

Обозначение классификатора	Тип действия	Влияние активного шага на данное действие
N	Несохраняемое	Действие активно в течение активности шага
R	Внеочередной сброс	Деактивация действия
S	Установка	Действие активно вплоть до сброса
L	Ограниченное по времени	Действие активно в течение указанного времени, но не дольше времени активности шага
D	Отложенное	Действие активируется по прошествии указанного времени, если шаг еще активен и продолжает быть активным.
P	Импульс	Действие выполняется один раз, если шаг активен
SD	Сохраняемое и отложенное	Действие активно после указанного времени до сброса
DS	Отложенное и сохраняемое	Действие активно после указанного времени, если шаг еще активен, вплоть до сброса
SL	Сохраняемое и ограниченное по времени	Активно после указанного времени

Порядок проведения лабораторной работы

При выполнении работы необходимо дополнительно использовать руководство пользователя по программированию ПЛК в CoDeSys 2.3, справочную систему комплекса программирования CoDeSys.

1 Изучить основные правила составления управляющих программ с использованием SFC-схем в системе CoDeSys.

2 Изучить порядок ввода редактирования и отладки SFC-схем в системе CoDeSys.

3 Подготовить ответы на контрольные вопросы.

4 Получить у преподавателя задание к лабораторной работе.

5 Составить управляющую программу в виде SFC-схемы.

6 Проверить работу управляющей программы в режиме эмуляции.

7 Записать программу в память контроллера и проверить ее выполнение.

8 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующие разделы.

1 Титульный лист установленного образца.

2 Цель работы.

3 Основные правила составления SFC-схем.

4 Листинг программы.

5 Ответы на контрольные вопросы.

6 Вывод по лабораторной работе.

Контрольные вопросы

1 Какие виды шагов могут использоваться в SFC-схемах системы CoDeSys?

2 Как связать действия с шагом в программах на языке SFC-системы CoDeSys?

3 Какое выражение используется в качестве условия перехода в SFC-схемах системы CoDeSys?

4 Что определяет логические переменные активности шага и как она изменяется при выполнении программы?

5 В чем состоит отличие шагов в упрощенной реализации SFC от МЭК- шагов?

6 Каким образом организуются переходы в управляющих программах, написанных на языке SFC?

7 Для чего используется неявные переменные и флаги в языке SFC-системы CoDeSys?

8 Как вставить комментарии в управляющую программу на языке SFC?

9 Как выполняются альтернативные и параллельные ветви SFC-схемы?

10 Как установить классификатор действия и что он определяет в управляющих программах на языке SFC?

11 Как установить точки останова при отладке программы на языке SFC?

12 Как определить время выполнения шага при отладке программы на языке SFC?

2 Лабораторная работа № 2. Разработка программ управления технологическими установками на языке SFC

Цель работы: изучение методики разработки программ управления технологическими установками для промышленных логических контроллеров (ПЛК) на языке SFC пакета CoDeSys; приобретение навыков разработки управляющих программ на языке SFC в системе CoDeSys.

Основные этапы разработки программ на языке SFC.

Многие автоматизируемые технологические установки в процессе работы выполняют действия в определенной последовательности или с определенной цикличностью. Технологический процесс у них является последовательным и представляет собой совокупность отдельных тактов, каждый из которых характеризуется выполняемыми операциями или состоянием оборудования.

Выполнение каждого такта технологического цикла должно начинаться только по окончании выполнения предыдущего такта. Переход между тактами осуществляется по информационному сигналу о состоянии объекта управления либо по сигналам с пульта оператора.

Устройство управления последовательным технологическим процессом может быть реализовано на базе программируемого логического контроллера (ПЛК).

Задача ПЛК в этом случае обеспечить не только логику управления установкой, но и заданную последовательность выполнения тактов.

Управляющую программу для таких систем рекомендуется составлять на стандартном языке программирования ПЛК – языке SFC.

Типовая последовательность разработки управляющей программы на языке SFC содержит следующие этапы:

– изучение технологической установки;

– выбор элементов управления пульта оператора;

– составление алгоритма управления установкой;

– составление таблиц входных и выходных сигналов контроллера;

– написание программы.

Приступая к разработке программы, следует вначале изучить работу установки, определить состав оборудования и особенности выполнения отдельных технологических операций. Особое внимание необходимо обратить на назначение датчиков и исполнительных устройств технологической установки, определить порядок использования информационных сигналов датчиков и формирования управляющих сигналов на исполнительные устройства.

Затем следует определить состав управляющих элементов пульта оператора (кнопки, переключатели, индикаторы и др.), которые необходимы оператору для контроля состояния технологического оборудования и управления им. Основные требования к пульту оператора – обеспечение работоспособности технологической установки в различных режимах и удобство работы обслуживающему персоналу.

Разработку алгоритма необходимо начинать с анализа технологического процесса. При анализе работы технологического объекта следует выделить состояния, в которых может находиться оборудование в процессе работы, определить действия, выполняемые в каждом состоянии, и условия перехода между состояниями. Если технологический объект представляет собой комплексную установку, состоящую из нескольких функциональных устройств, управление которыми может осуществляться отдельно, то для каждого устройства определяется своя последовательность состояний. Результатом анализа должно быть

описание работы оборудования в словесной или формализованной форме. Основные способы описания работы оборудования в формализованном виде:

– циклограмма работы оборудования;

– сетевой граф изменения состояний;

– блок-схема алгоритма последовательности действий.

На циклограмме показывают схему согласованности во времени работы исполнительных устройств технологического объекта, функционирующего по заданному циклу. Цикл разбивается на отдельные такты. В рамках одного такта технологический объект функционирует с неизменной комбинацией состояния («включено – выключено») дискретных исполнительных устройств и датчиков.

Сетевой граф изображается в виде вершин (окружностей) и ребер (линий, соединяющих окружности). Вершинам соответствуют состояния технологического объекта, ребрам – условия перехода между состояниями.

Блок-схема схема алгоритма составляется в виде последовательности условных графических изображений действий и условий перехода. Действия изображаются в виде прямоугольников, условия переходов – в виде ромбов.

Алгоритм управления технологической установкой описывает процесс взаимодействия ПЛК с объектом и оператором. В отличие от алгоритма работы оборудования в нем указываются действия, которые необходимо выполнять контроллеру, чтобы обеспечить требуемое поведение технологической установки. Действия контроллера заключаются в обработке информационных сигналов датчиков, выдаче управляющих сигналов на исполнительные устройства, выполнения временных функций и функций счета, обслуживании пульта оператора.

Для разработки программы желательно составление таблиц входных и выходных сигналов программируемого контроллера. Таблицы должны содержать наименование и условное обозначение сигналов, их источник или приемник, адресацию сигналов и их привязку к контактам разъемов контроллера.

Завершающим этапом проектирования программно-логической подсистемы управления дискретным процессом являются написание программы управления для программируемого контроллера, которая составляется на основании информации, подготовленной на предыдущих этапах, и ее отладка.

Пример разработки управляющей программы

Объект управления представляет собой роботизированный технологический комплекс (РТК). В состав РТК входят (рисунок 2.1) промышленный робот (ПР), конвейер подающий, контейнер для складирования деталей.

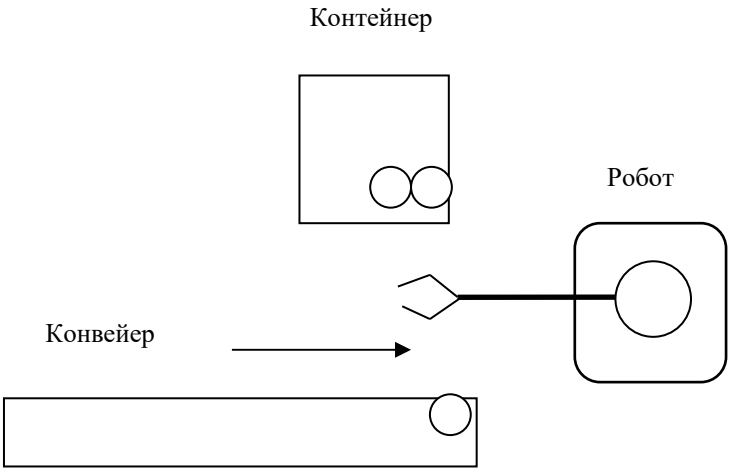


Рисунок 2.1 – Структурная схема РТК

В исходном состоянии РТК отключен и не выполняет никаких действий. Переключение в рабочее состояние происходит по сигналу оператора. В рабочем состоянии РТК выполняет следующие действия. Если деталь поступает на начало конвейера, конвейер включается и деталь перемещается к конечной точке, конвейер отключается. Робот поворачивается, берет деталь и укладывает ее в контейнер. После поступления очередной детали цикл повторяется. Работа установки прекращается по сигналу оператора или при заполнении контейнера (емкость контейнера – 50 деталей). Возобновляется работа после смены контейнера по сигналу оператора.

Привод конвейера осуществляется от асинхронного двигателя, в начале и конце конвейера установлены датчики контроля наличия детали. Робот приводится в движение с помощью пневмоцилиндров, которые управляются электроклапанами, установленными на распределительной пневмопанели. Крайние положения руки робота контролируются датчиками конечного положения. Еще один датчик конечного положения используется для контроля наличия контейнера в позиции складирования. Для включения-отключения РТК на панели оператора установлены две кнопки. Состояние оборудования можно определить по индикатору, который также находится на панели оператора. Светящийся индикатор указывает, что РТК находится в рабочем режиме.

Управление РТК осуществляется контроллером ПЛК-110-60, который расположен в шкафу электроавтоматики. Там же находится автоматический выключатель и пускатель двигателя конвейера.

Структурная схема системы управления РТК представлена на рисунке 2.2.

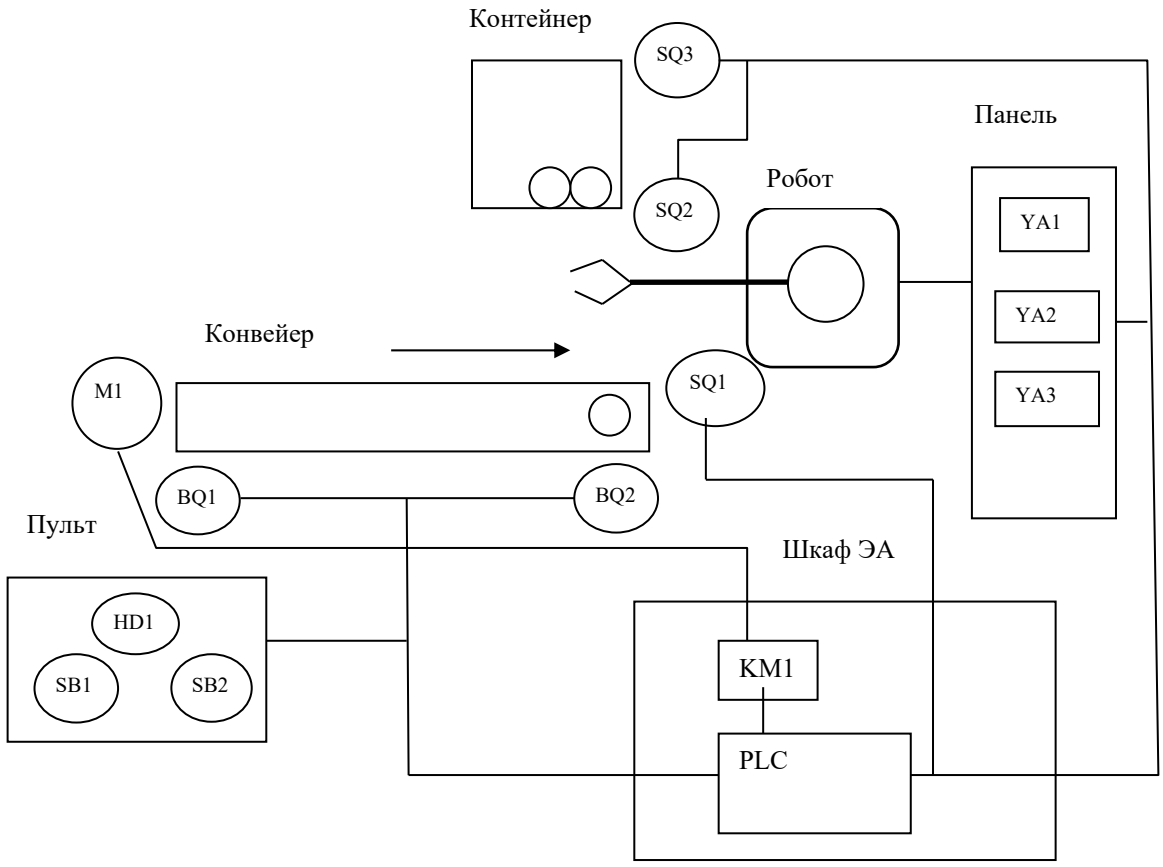


Рисунок 2.2 – Структурная схема системы управления РТК

Назначение датчиков РТК:
BQ1 – наличие детали в начале конвейера;
BQ2 – наличие детали в конце конвейера;
SQ1 – робот в позиции захвата детали;
SQ2 – робот в позиции складирования детали;
SQ3 – наличие контейнера в позиции складирования.
Назначение исполнительных устройств:
M1 – асинхронный двигатель привода конвейера;
YA1 – клапан пневмоцилиндра схвата;
YA2 – клапан пневмоцилиндра поворота робота в позицию захвата;
YA3 – клапан пневмоцилиндра поворота робота в позицию складирования.
Назначение элементов пульта оператора:
SB1 – кнопка ПУСК – включение РТК;
SB2 – кнопка СТОП – отключение РТК;
HD1 – индикатор РАБОТА.

В составе оборудования РТК имеется два функциональных устройства, работающих практически независимо друг от друга: конвейер и робот. Это позволяет рассматривать работу каждого устройства в отдельности.

В цикле работы конвейера можно выделить четыре состояния:

– конвейер отключен;

– конвейер ожидает поступления детали в начальную позицию;

– конвейер перемещает деталь в конечную позицию;

– конвейер ожидает снятия детали.

Изменение состояния конвейера определяется сигналами кнопок пульта оператора и датчиков наличия детали на конвейере.

В исходном состоянии конвейер отключен. Переход в состояние ожидания детали происходит при наличии сигнала кнопки ПУСК (SB1=1) при условии, что контейнер не заполнен (CT < 50) и находится в позиции складирования (SQ3 = 1). При поступлении детали в начальную позицию (BQ1 = 1) включается двигатель (KM1 = 1), и деталь перемещается в конечную позицию (BQ2 = 1), двигатель отключается (KM1 = 0) и конвейер ожидает снятия детали роботом. После снятия детали (BQ2 = 0) конвейер переходит в состояние ожидания поступления очередной детали. Отключение конвейера происходит при поступлении сигнала от кнопки СТОП (SB2 = 0) при условии, что он находится в режиме ожидания детали, или после заполнения контейнера (CT = 50).

В цикле работы робота можно выделить следующие состояния:

– робот отключен;

– робот ожидает поступления детали в позицию захвата;

– робот переносит деталь в контейнер;

Изменение состояния робота определяется сигналами кнопок пульта оператора и датчиком наличия детали в позиции захвата.

В исходном состоянии робот отключен. Переход в состояние ожидания детали происходит при наличии сигнала кнопки ПУСК (SB1 = 1) при условии, что контейнер не заполнен (CT < 50) и находится в позиции складирования (SQ3 = 1). При поступлении детали в позицию захвата (BQ2 = 1) робот переносит деталь в контейнер и переходит в состояние ожидания поступления очередной детали. Отключение робота происходит при поступлении сигнала от кнопки СТОП (SB2 = 0) при условии, что он находится в режиме ожидания детали, или после заполнения контейнера (CT = 50).

Цикл переноса детали включает в себя следующие действия:

– поворот руки робота в позицию захвата (YA1 = 1);

– зажим схвата (YA3 = 1);

– поворот руки робота в позицию складирования (YA1 = 0, YA2 = 1);

– разжим схвата (YA3 = 0);

– поворот руки робота в исходное состояние (YA2 = 0).

Выполнение операций поворота руки робота подтверждается сигналами датчиков SQ1 и SQ2. Для повышения надежности при переносе детали необходимо предусмотреть выдержку времени между операциями разжима-зажима схвата и началом движения руки робота.

Алгоритм управления конвейером приведен на рисунке 2.3.

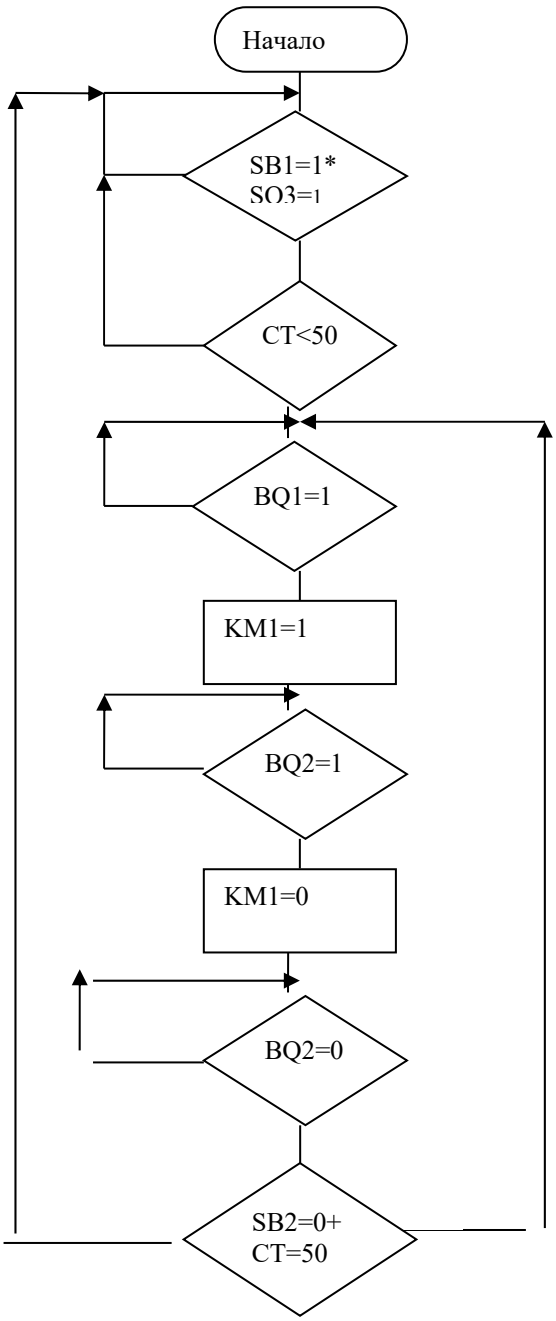


Рисунок 2.3 – Блок-схема алгоритма управления конвейером

Алгоритм управления роботом представлен на рисунке 2.4.

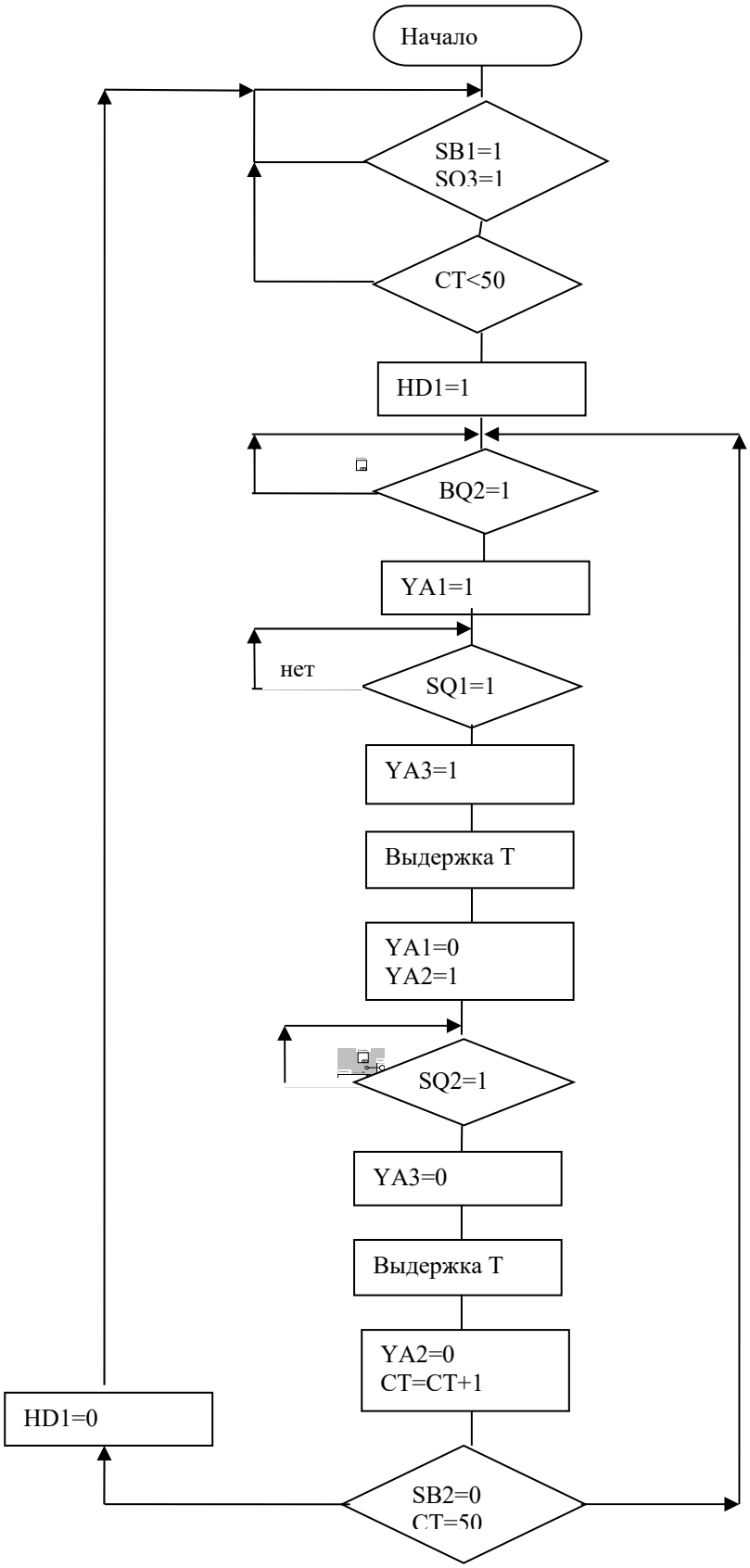


Рисунок 2.4 – Блок-схема алгоритма управления роботом

Схема подключения пульта оператора, датчиков и исполнительных устройств к ПЛК проектируемой системы управления представлена на рисунке 2.5.

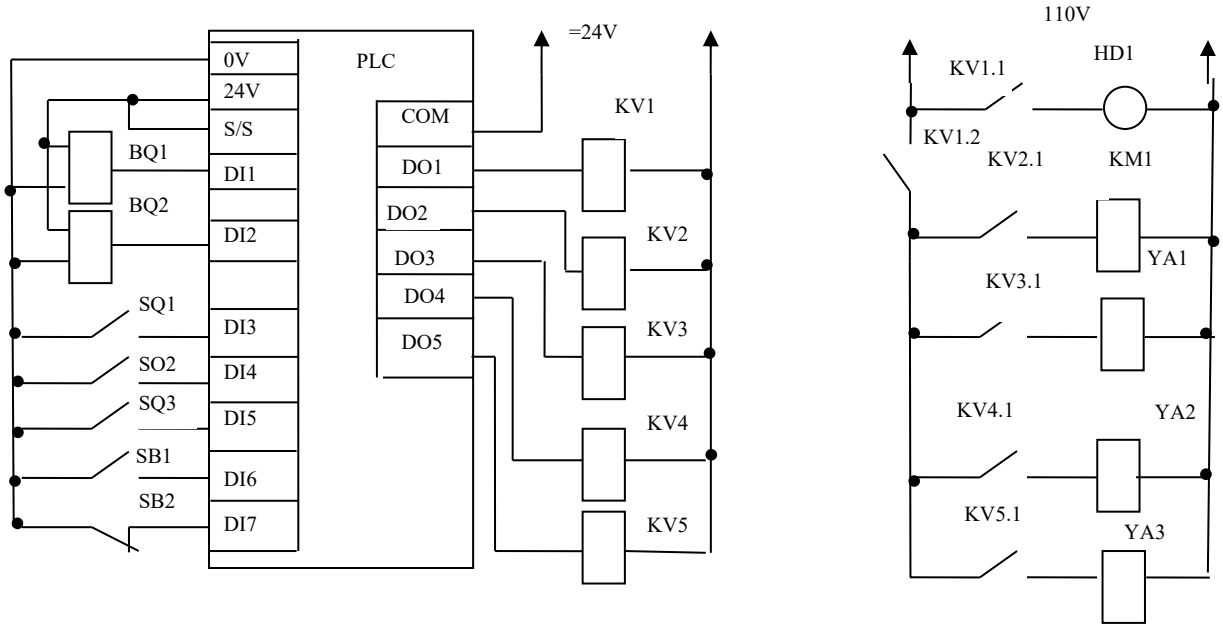


Рисунок 2.5 – Схема подключения ПЛК 110-60К

Входные сигналы ПЛК и соответствующие им символьные переменные, используемые в управляющей программе, приведены в таблице 2.1.

Таблица 2.1 –Таблица входных сигналов ПЛК

Номер входа	Подключаемый элемент	Тип сигнала	Символьное обозначение	Адрес
DI1	BQ1	BOOL	DETN	%IX0.0
DI2	BQ2	BOOL	DETK	%IX0.1
DI3	SQ1	BOOL	ROBZH	%IX0.2
DI4	SQ2	BOOL	ROBSKL	%IX0.3
DI5	SQ3	BOOL	KONT	%IX1.0.0
DI6	SB1	BOOL	PUSK	%IX1.0.1
DI7	SB2	BOOL	STOP	%IX1.0.2

Выходные сигналы ПЛК и соответствующие им символьные переменные, используемые в управляющей программе, представлены в таблице 2.2.

Таблица 2.2 – Таблица выходных сигналов ПЛК

Номер входа	Подключаемый элемент	Тип сигнала	Символьное обозначение	Адрес
DO1	KV1-HD1	BOOL	RAB	%QX2.0
DO2	KV2-KM1	BOOL	KONV	%QX2.1
DO3	KV3-YA1	BOOL	POVZH	%QX2.2
DO4	KV4-YA2	BOOL	POVSKL	%QX2.3
DO5	KV5-YA3	BOOL	SHVAT	%QX3.0.0

В таблице 2.3 приведены символьные переменные, объявляемые в управляющей программе.

Таблица 2.3 – Таблица символьных переменных

Символьное обозначение	Тип переменной	Адрес	Атрибут	Начальное значение	Комментарий
COUNT	BYTE	%MB10.0	RETAIN		Счетчик деталей

Управляющая программа, разработанная в виде схемы на языке SFC, представлена на рисунке 2.6.

Каждому шагу схемы SFC соответствует определенное состояние оборудования РТК. Управление конвейером и роботом в программе осуществляется в параллельных ветвях, что отражает их определенную функциональную независимость.

Действия, выполняемые в шаге, заключаются во включении-отключении исполнительных устройств РТК, т. е. в присваивании переменной, соответствующей выходному сигналу, значения 0 либо 1. Исполнительные устройства, на которые выдается управляющий сигнал, соответствуют устройствам, указанным в соответствующих блоках алгоритма управления РТК. Шаги Hold1, Hold2 и Hold3 являются пустыми, никаких действий они не выполняют и служат для перевода робота и конвейера в состояние ожидания.

Условия переходов между шагами определяются значением переменных входных сигналов датчиков и кнопок пульта оператора, а также переменной COUNT, значение которой равно количеству деталей в контейнере. Выдержка времени после зажима-разжима схвата реализована с помощью неявных переменных контроля времени активности шага Shvaz.t и Shvar.t.

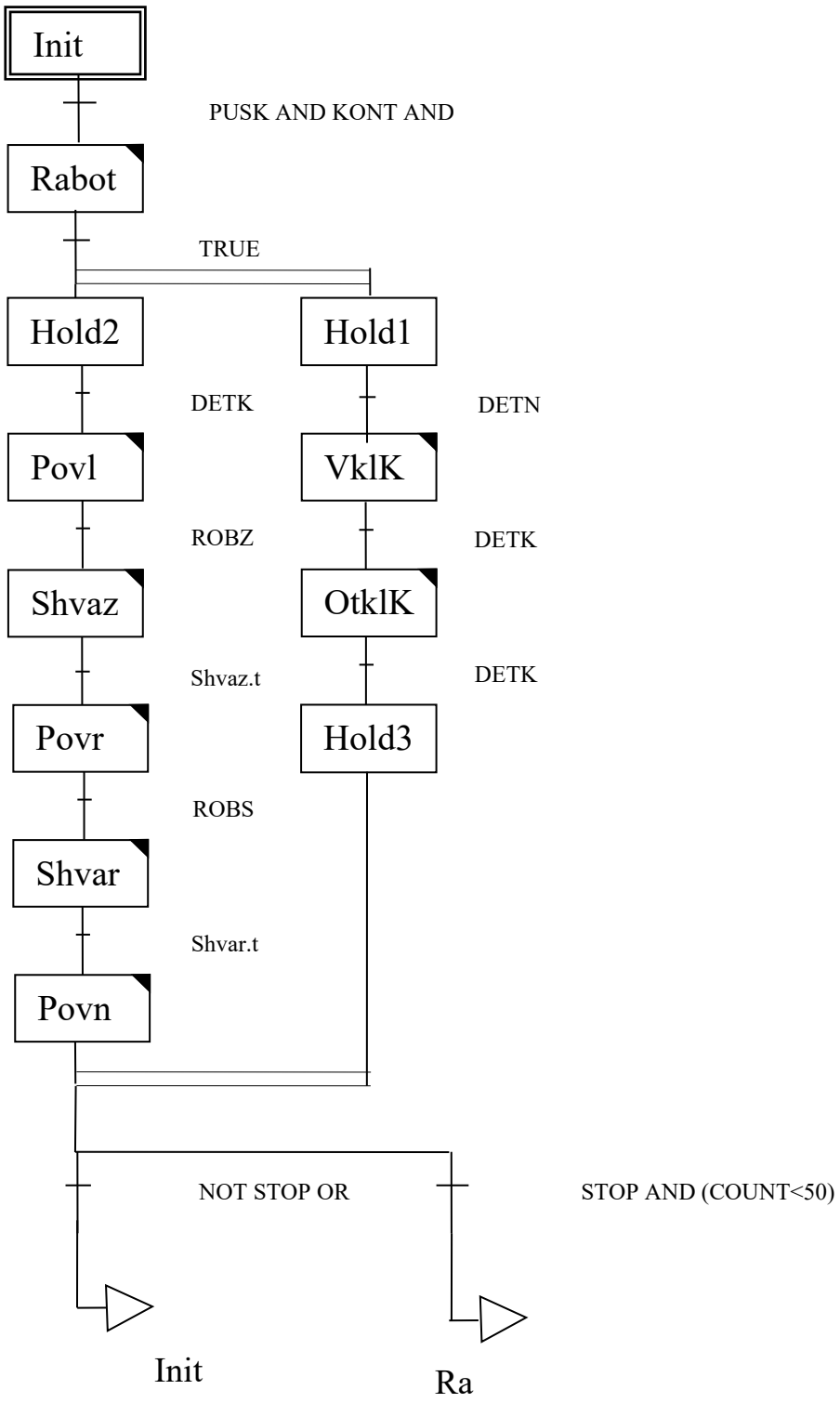


Рисунок 2.6 – Управляющая программа на языке SFC

Порядок проведения лабораторной работы

1 Изучить принцип работы заданной технологической установки.

2 Разработать функциональную схему управления технологической установкой.

3 Выбрать элементы (кнопки, датчики исполнительных устройств, индикаторы), необходимые для реализации схемы управления установкой.

4 Составить блок-схему алгоритма управления установкой.

5 Разработать схему подключения выбранных элементов к программируемому контроллеру.

6 Составить таблицу входных и выходных сигналов контроллера с указанием подключаемого элемента, типа сигнала, символьного обозначения, адреса.

7 Составить управляющую программу, реализующую управление установкой в виде SFC-схемы.

8 Проверить работу управляющей программы в режиме эмуляции.

9 Записать программу в память контроллера и проверить ее выполнение.

10 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующие разделы.

1 Титульный лист установленного образца.

2 Цель работы.

3 Функциональная схема управления технологической установкой.

4 Блок-схема алгоритма работы установки.

5 Схема подключения элементов управления к программируемому логическому контроллеру.

6 Таблица входных и выходных сигналов контроллера.

7 Таблица внутренних переменных с указанием, символьного обозначения, типа сигнала и адреса.

8 Листинг программы.

9 Ответы на контрольные вопросы.

10 Вывод по лабораторной работе.

Контрольные вопросы

1 В каких случаях технологический процесс можно разделить на отдельные такты?

2 Какие элементы системы управления технологическим процессом рекомендуется располагать на пульте оператора?

3 Каким образом можно описать алгоритм работу технологической установки?

4 Чем характеризуется состояние технологического оборудования при описании его работы в виде сетевого графа?

5 Какую информацию рекомендуется помещать в таблицу входных- входных сигналов контроллера?

6 Чем отличается алгоритм работы технологического оборудования от алгоритма управления оборудованием?

7 Каким образом шаги схемы на языке SFC связаны с алгоритмом работы технологического оборудования?

3 Лабораторная работа № 3. Создание визуализаций в среде CoDeSys

Цель работы: изучение принципов создания визуализаций проекта в системе CoDeSys; приобретение навыков создания визуализаций в системе CoDeSys.

Общие сведения

Визуализация представляет собой графическое изображение проектируемой системы, которое может служить пользовательским интерфейсом для контроля и управления работой системы.

Визуализация может исполняться в системе программирования, в отдельном приложении CoDeSys HMI, как веб-приложение на сервере или как целевая программа в ПЛК.

Редактор визуализации CoDeSys предоставляет набор готовых графических элементов, которые могут быть связаны соответствующим образом с переменными управляющей программы. Форма и цвет графических элементов могут изменяться при работе программы в зависимости от значений переменных.

Свойства отдельных элементов визуализации, а также визуализации в целом устанавливаются в соответствующих диалогах конфигурации и диалоге свойств объекта. Здесь определяется начальный вид элементов и выполняется привязка динамических свойств к значениям переменных проекта.

На рисунке 3.1 приведен пример визуализации.

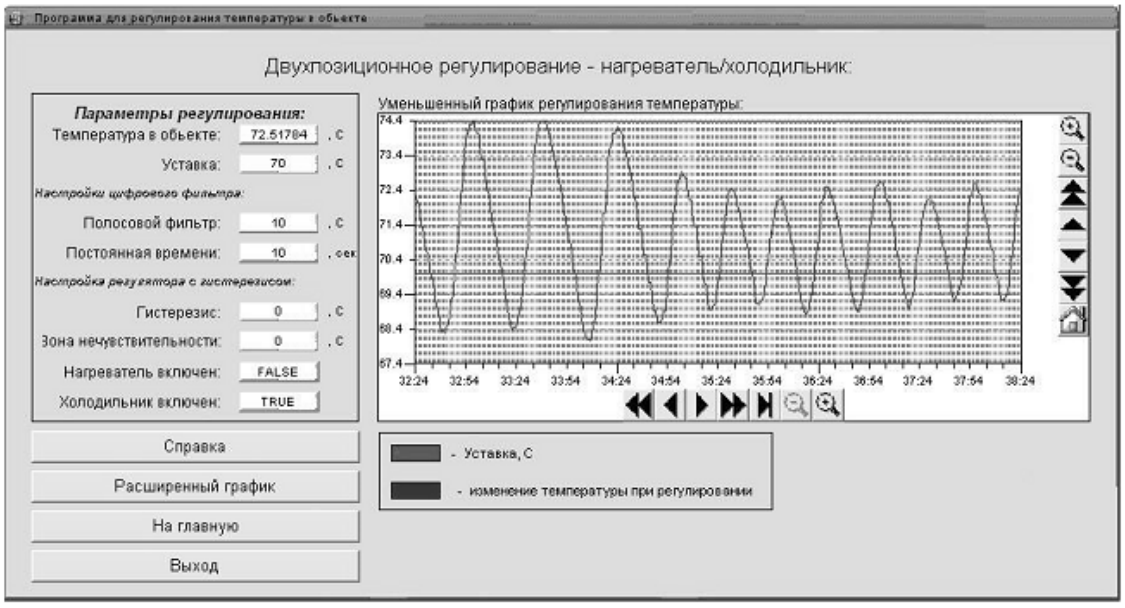


Рисунок 3.1 – Пример визуализации программы контроллера

Создание файла визуализации

Для создания визуализации нужно перейти на вкладку «Визуализации» организатора объектов, после чего правой кнопкой мыши вызвать контекстное меню в пустом поле и выбрать строку «Добавить объект», после чего появится окно визуализации (рисунок 3.2).

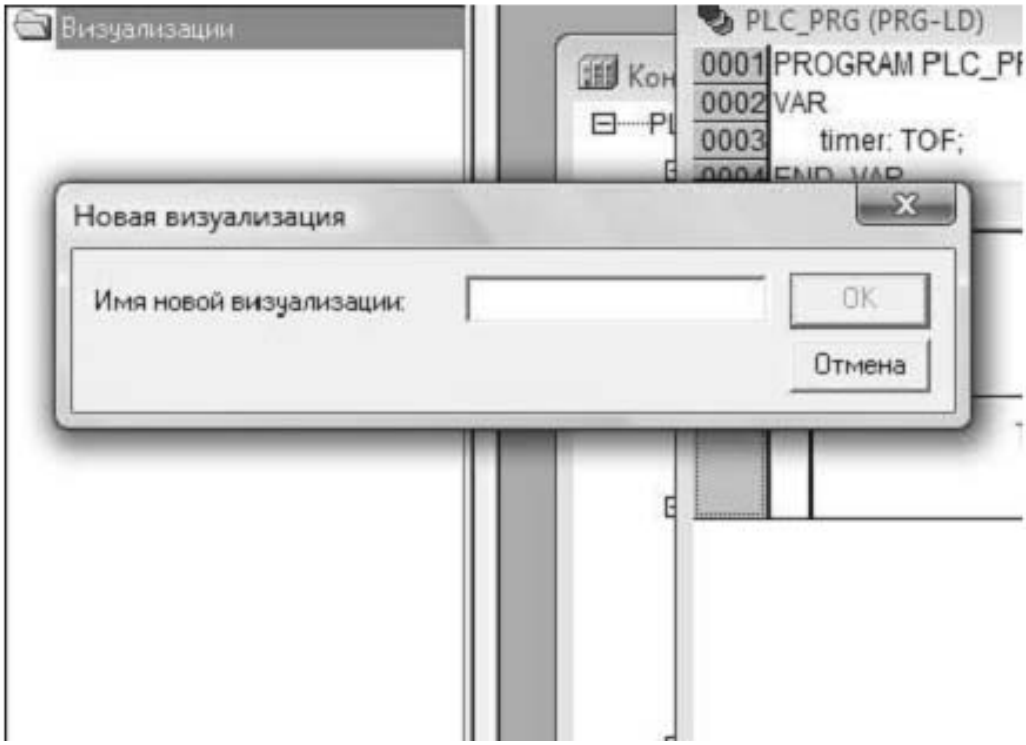


Рисунок 3.2 – Вызов окна визуализации

Элемент визуализации – это графический элемент, который используется при построении объекта визуализации. Возможные элементы представлены в виде иконок на панели инструментов CoDeSys (рисунок 3.3). Каждый элемент имеет собственную конфигурацию (набор свойств). Имеется возможность вставлять в визуализацию различные геометрические формы, а также точечные рисунки, метафайлы, кнопки и существующие визуализации.

Рисунок 3.3 – Панель инструментов редактора визуализации

У каждого элемента визуализации есть свои свойства, вызвать свойства объекта визуализации можно двойным щелчком мыши по объекту либо активизировать правой кнопкой мыши объект в контекстном меню и выбрать строку «Конфигурировать». В открывшемся окне можно задавать необходимые свойства объекта визуализации (рисунок 3.4).

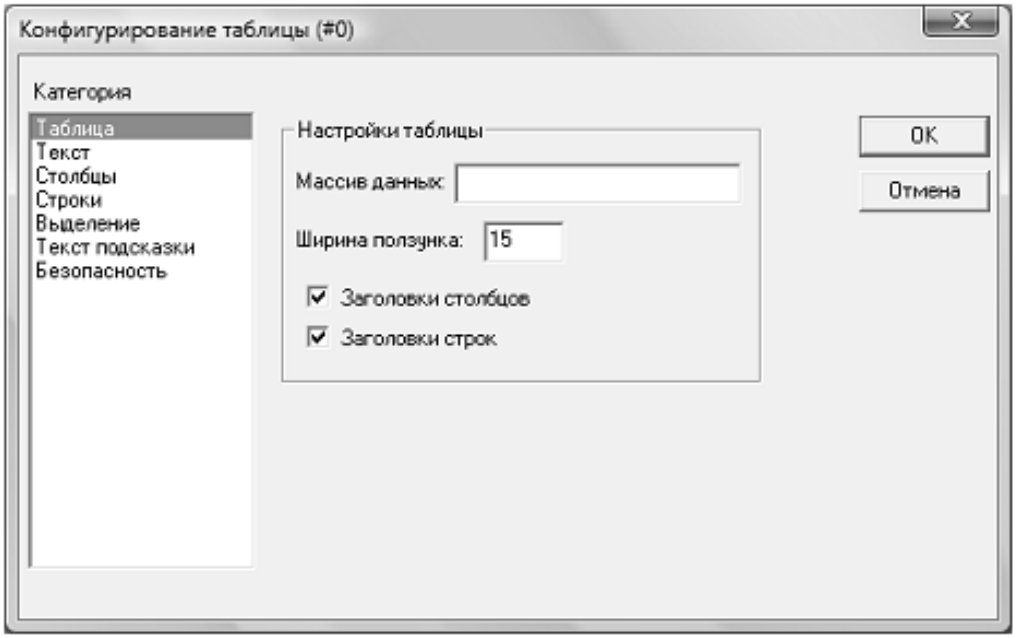


Рисунок 3.4 – Окно конфигурирования

Пример – создание простейшей визуализации.

Создадим программу на языке LD в виде цепочки, состоящей из «Контакта» и «Катушки».

Перейдем на вкладку «Визуализация» и вызовем контекстное меню для добавления новой визуализации (рисунок 3.5).

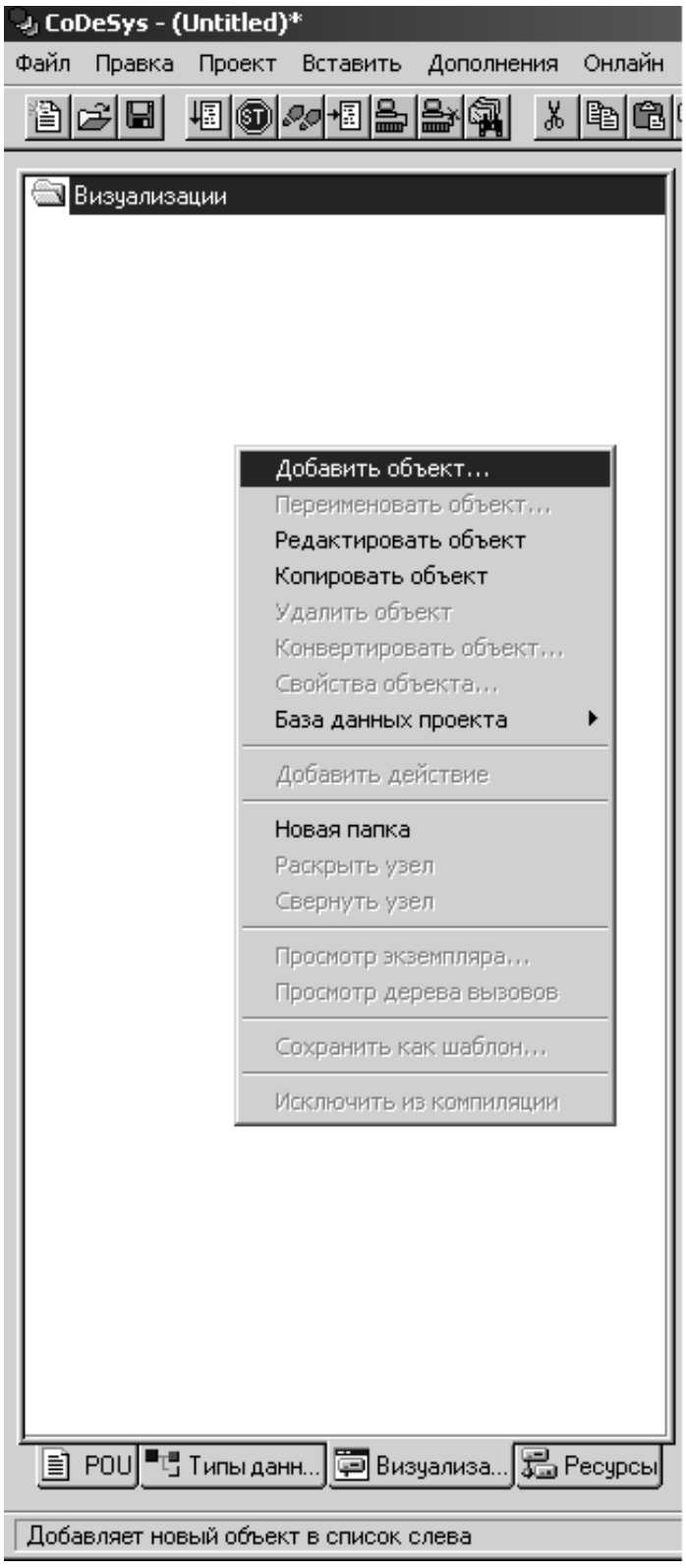


Рисунок 3.5 – Добавление объекта визуализации

В появившемся окне введем имя для визуализации (рисунок 3.6).

Рисунок 3.6 – Ввод имени визуализации

После ввода имени, появляется новое окно для создания визуализации.

Вставляем с помощью панели инструментов элементы «Эллипс» и «Кнопка», как на рисунке 3.7.

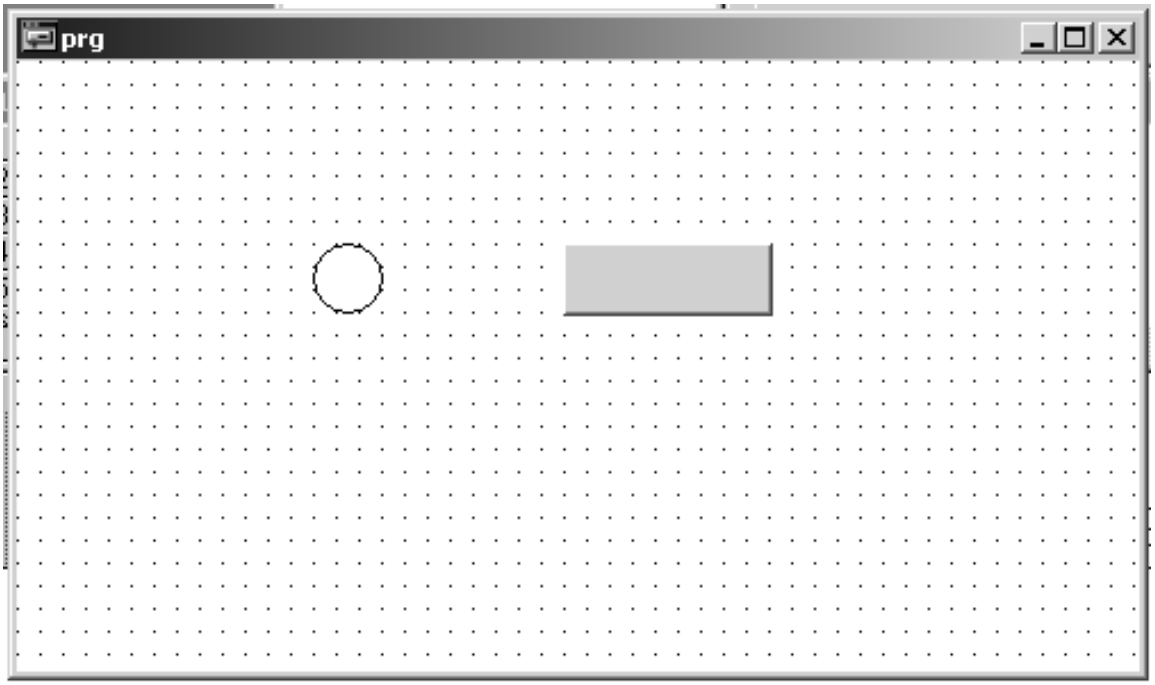


Рисунок 3.7 – Ввод элементов визуализации

Теперь необходимо сконфигурировать эти элементы. Пусть необходимо сделать следующую визуализацию: эллипс должен менять цвет, когда срабатывает катушка реле, а при нажатии на кнопку должна изменять значение переменная типа BOOL.

Сначала добавляем переменную C типа BOOL в проект (рисунок 3.8).

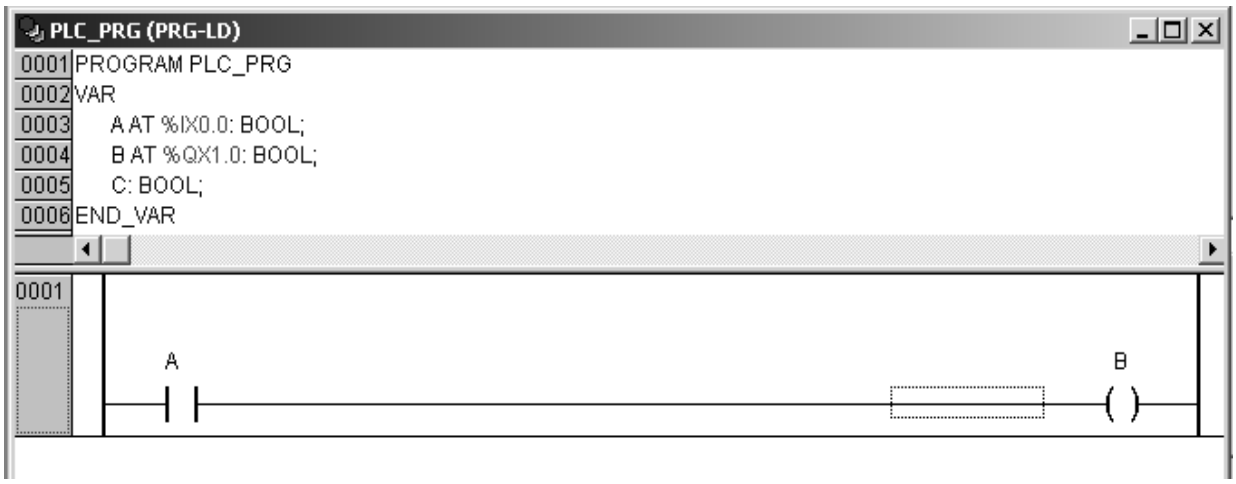


Рисунок 3.8 – Переменные проекта

Затем перейдем в окно редактора визуализации и щелкнем по эллипсу для вызова окна с настройками (рисунок 3.9).

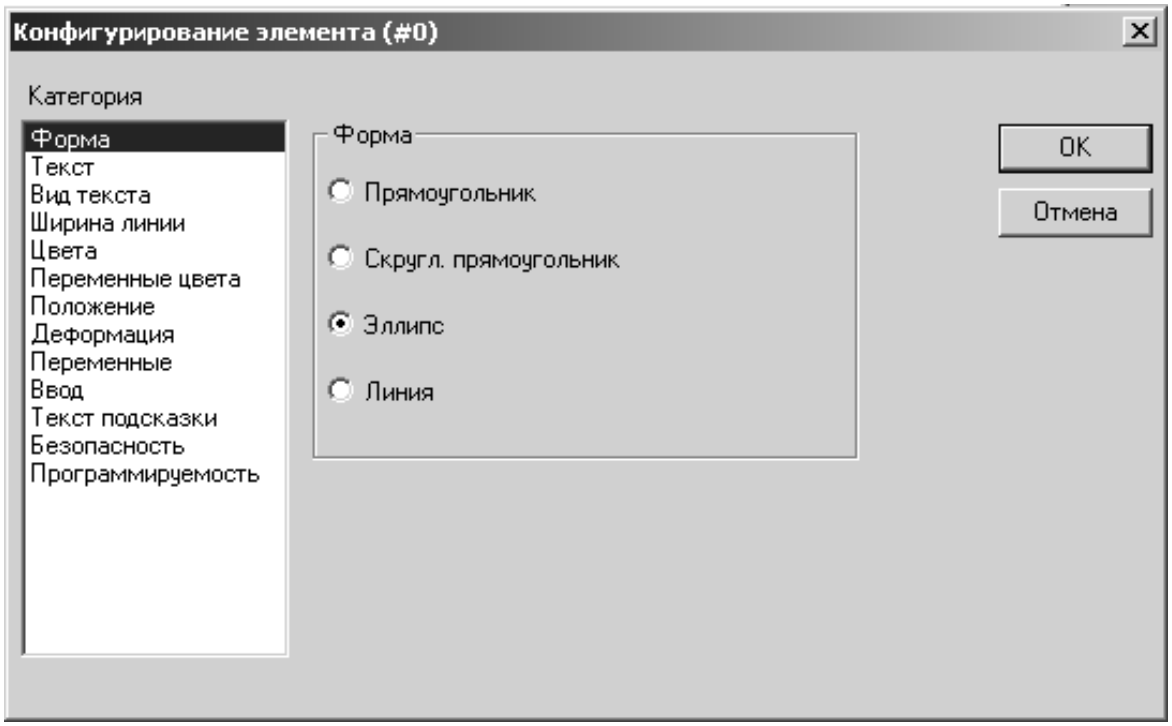


Рисунок 3.9 – Окно конфигурирования эллипса

Настроим категории «Цвета» и «Переменные». В категории «Цвета» выберем цвет эллипса в двух состояниях (рисунок 3.10).

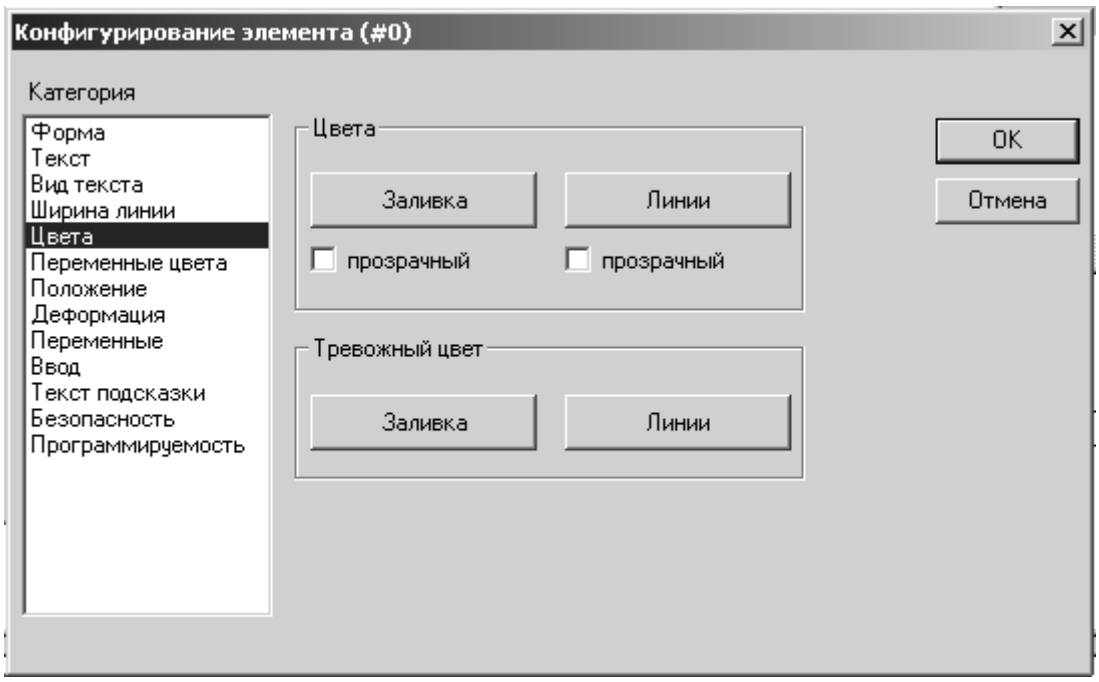


Рисунок 3.10 – Окно конфигурации цвета эллипса

Затем необходимо указать переменную, которая будет изменять цвет, для этого перейдем в категорию «Переменные» и установим курсор в поле «Изм. цвета:» (рисунок 3.11).

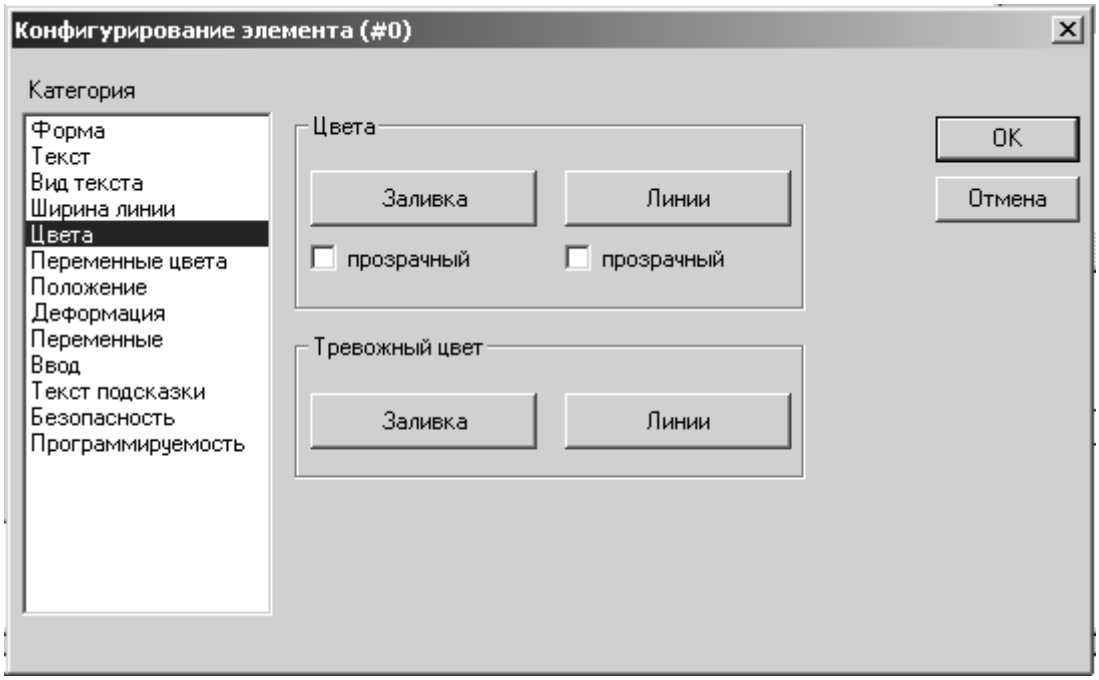


Рисунок 3.11 – Окно конфигурации переменных эллипса

Для связывания с переменными используем ассистент ввода (клавиша F2), в появившемся окне выберем нужную переменную – переменная B (рисунок 3.12).

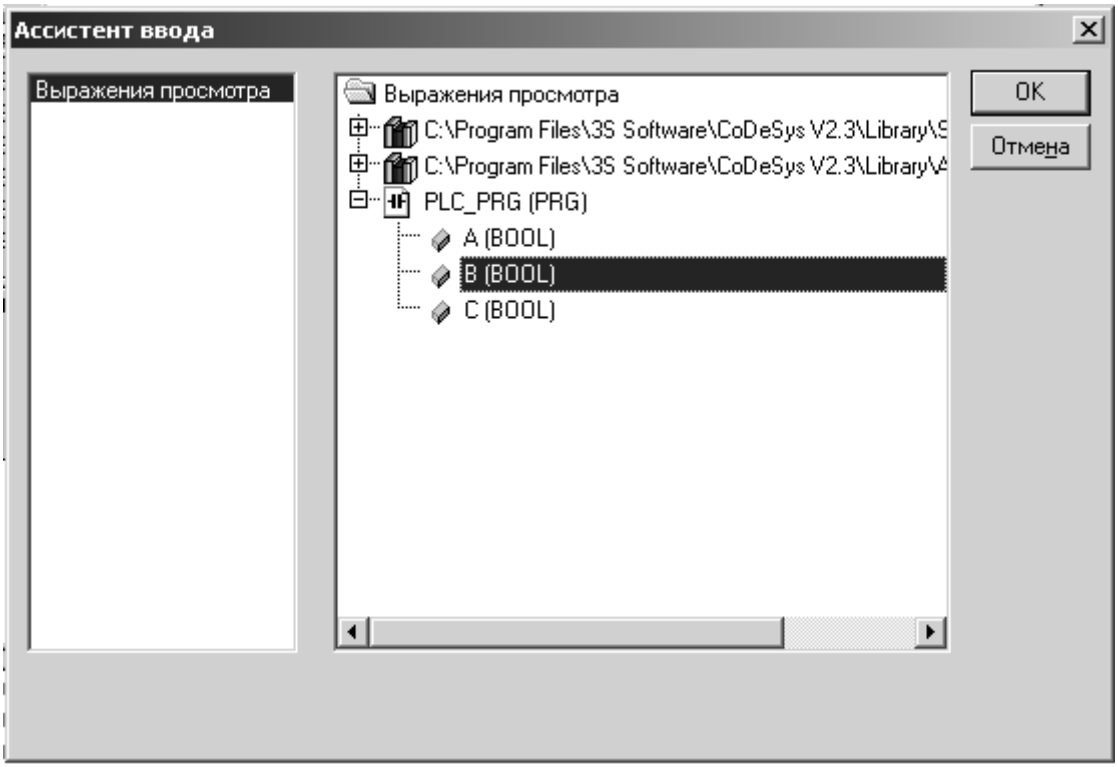


Рисунок 3.12 – Окно ассистента ввода

После нажатия кнопки ОК в окне конфигурации переменных появится переменная для изменения цвета (рисунок 3.13).

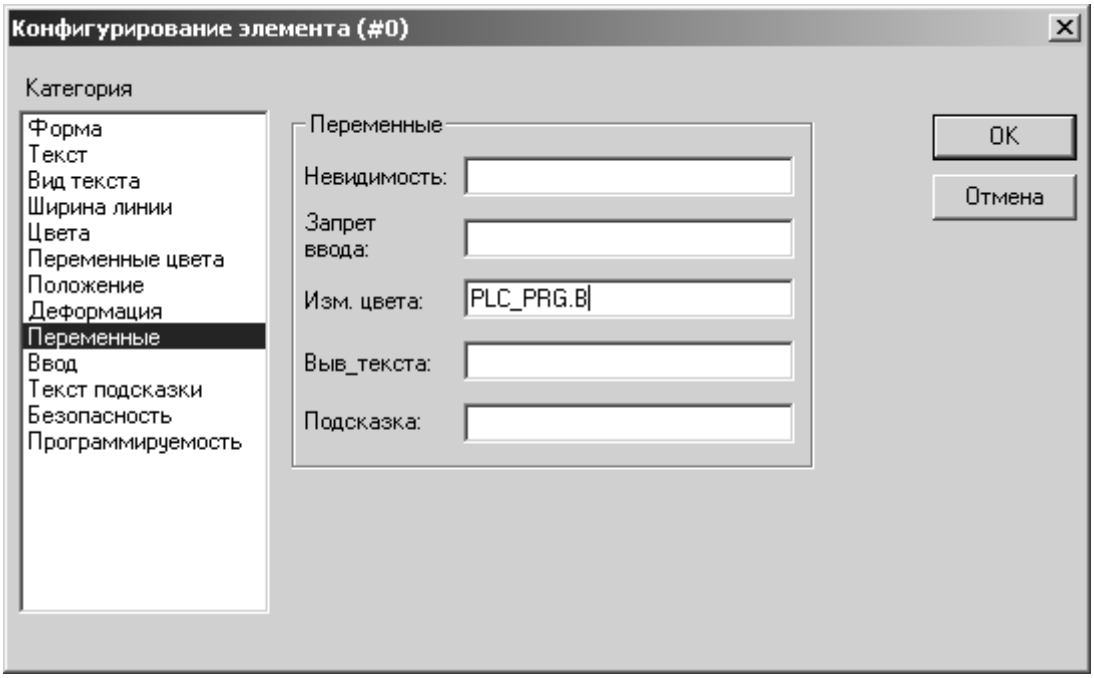


Рисунок 3.13 – Сконфигурированное окно переменных эллипса

Далее вызываем окно конфигурирования кнопки (рисунок 3.14) и выбираем категорию «Ввод».

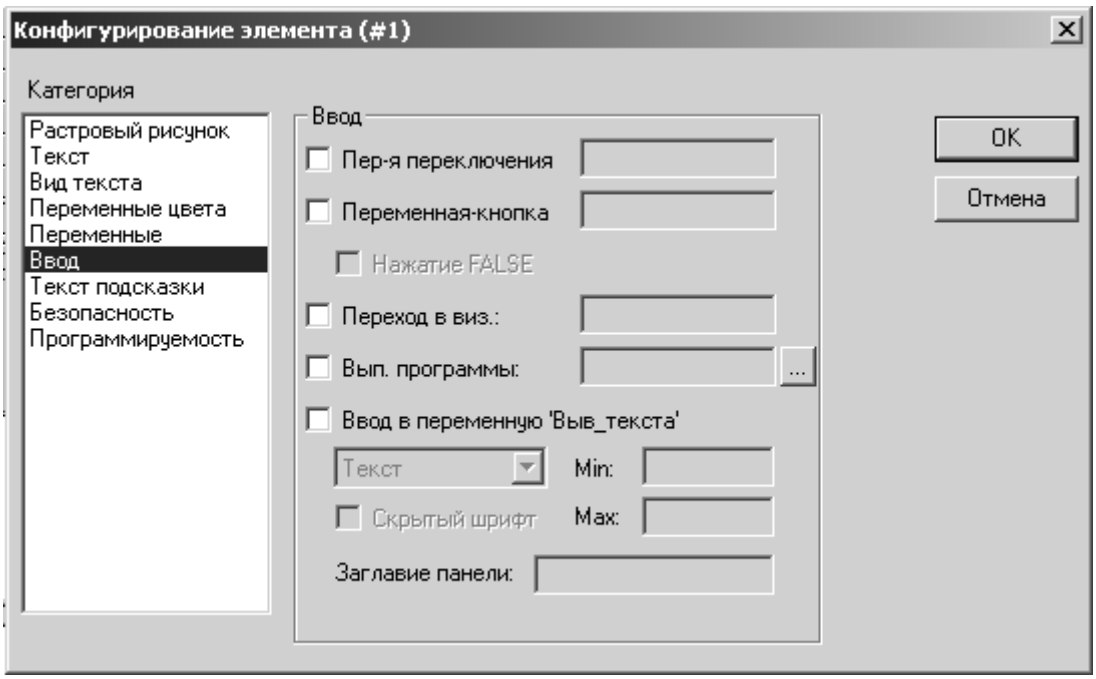


Рисунок 3.14 – Окно конфигурирования переменных кнопки

Поставим галочку «Переменная кнопка» (чтобы получить кнопку с фиксацией, необходимо выбирать «Переменная переключения») и, установив курсор в появившееся поле ввода, вызовем ассистент ввода, чтобы привязать к кнопке переменную C (рисунок 3.15).

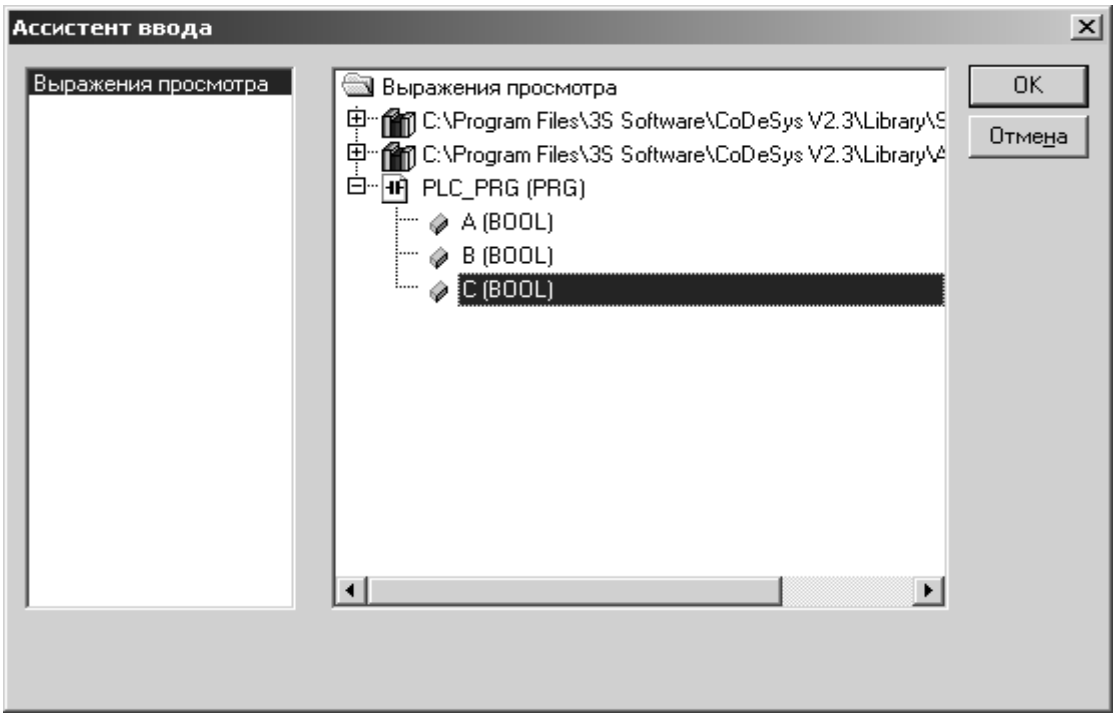


Рисунок 3.15 – Присваивание переменной

Сконфигурированное окно переменных показано на рисунке 3.16.

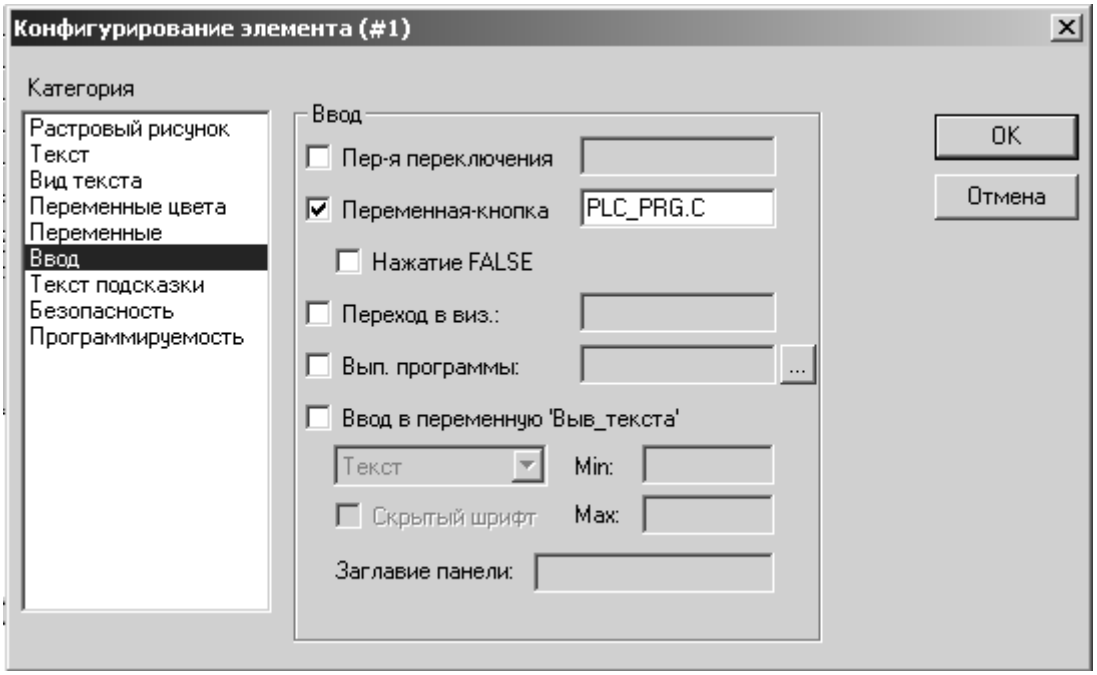


Рисунок 3.16 – Сконфигурированное окно переменных кнопки

Создание визуализации завершено. Можно загрузить программу в ПЛК и проверь работу визуализации.

Задание к лабораторной работе

Создать визуализацию для проекта по заданию преподавателя. При создании визуализации предусмотреть индикацию состояния датчиков технологической установки, а также возможность управления установкой из визуализации.

Порядок проведения лабораторной работы

При выполнении лабораторной работы необходимо дополнительно использовать справочную систему среды программирования CoDeSys, дополнение к руководству пользователя по программированию ПЛК в CoDeSys 2.3. Визуализация CoDeSys.

1 Изучить основные правила создания визуализаций в системе CoDeSys.

2 Изучить состав элементов визуализации в системе программирования CoDeSys.

3 Изучить порядок вставки, позиционирования и конфигурирования элементов визуализации в системе CoDeSys.

4 Изучить порядок конфигурирования объектов визуализации в системе CoDeS.

5 Подготовить ответы на контрольные вопросы.

6 Изучить пример и задание к лабораторной работе.

7 Создать визуализацию для программы управления технологической установкой.

8 Проверить работу визуализации при выполнении программы.

9 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующие разделы.

1 Титульный лист установленного образца.

2 Цель работы.

3 Описание последовательности создания визуализации.

4 Список элементов визуализации в системы CoDeSys.

5 Ответы на контрольные вопросы.

6 Вывод по лабораторной работе.

Контрольные вопросы

1 Какие формы визуализации реализованы в системе CoDeSys?

2 Что такое объект визуализации и элемент визуализации в системе CoDeSys?

3 Какие элементы визуализации можно использовать при создании визуализаций в системе CoDeSys?

4 Каким образом осуществляются выбор и вставка элемента визуализации в системе CoDeSys?

5 Какие действия можно производить над элементом визуализации в системе CoDeSys?

6 Как одновременно переместить несколько элементом визуализации в системе CoDeSys?

7 Какие свойства элемента визуализации можно изменять при конфигурировании его в системе CoDeSys?

8 Как добавить текст к элементу визуализации в системе CoDeSys?

9 Какие свойства объекта визуализации можно изменять при конфигурировании его в системе CoDeSys?

10 Как установить фоновый рисунок при создании визуализации в системе CoDeSys ?

4 Лабораторной работе № 4. Разработка программ управления электроприводами типовых механизмов

Цель работы: изучение методики разработки программ управления технологическим оборудованием в среде CoDeSys, приобретение навыков разработки управляющих программ для управления автоматизированным электроприводом.

Основные этапы проектирования

В общем случае процесс проектирования системы управления технологической установкой с использованием ПЛК, предусматривает выполнение следующих этапов:

– разработку технического задания на проектирование системы управления технологическим объектом;

– выбор управляемых и контролируемых параметров, степень и формы автоматизации;

– технико-экономическое обоснование применения ПЛК для управления заданным технологическим объектом (процессом);

– разработку общей структурной схемы системы управления;

– выбор технических средств системы управления: ПЛК, информационных и исполнительных устройств, блоков питания, устройств защиты, средств операторского интерфейса и др.;

– разработку схемы электрической принципиальной системы управления;

– построение алгоритм управления технологической установкой;

– разработку на основе алгоритма управления управляющей программы для ПЛК;

– оформление технической документации.

Разработка технического задания

Техническое задание содержит основные требования к проектируемой системе управления функциональные, технические, эксплуатационные и др.

Приступая к формированию требований к системе управления, вначале необходимо определить цели и задачи управления, выделить функции устройства управления, возможные пути повышения надежности и безопасности системы.

В общем случае система управления должна обеспечивать:

– работу технологического механизма в различных режимах (автоматический, ручной);

– возможность оперативного управления объектом в случае возникновения внештатных ситуаций;

– возможность установки оборудования в исходное состояние после включения или внештатной ситуации;

– возможность контроля состояния оборудования и параметров технологического процесса;

– возможность оперативного задания значений параметров;

– защиту обслуживающего персонала и оборудования при возникновении неисправностей или аварийных ситуаций;

– диагностику состояния оборудования системы при включении и в процессе работы.

На основании требований к системе осуществляют выбор управляемых и контролируемых параметров технологической установки, степень и формы ав-

томатизации отдельных звеньев и всего механизма в целом, определяют порядок взаимодействия объекта и устройства управления. При этом определяют задачи, которые необходимо решать в процессе управления, и функции устройства управления.

Разработка структурной схемы

На структурной схеме системы управления электроприводом типового промышленного механизма показывают: условное изображение технологического оборудования, приводные двигатели механизма, программируемый логический контроллер (ПЛК), шкаф электроавтоматики и пульт оператора, а также дополнительные устройства и элементы, обеспечивающие работу механизма в соответствии с его назначением.

К дополнительным устройствам относятся:

– тормозные устройства;

– управляемые муфты;

– регулирующие клапана;

– датчики технологических параметров и т. д.

В шкафу электроавтоматики показывают силовые коммутационные аппараты, пусковые устройства электродвигателей, преобразователи энергии, если привод механизма регулируемый. Промежуточные элементы, которые используются для преобразования или усиления сигналов, можно не указывать.

На пульте оператора изображают элементы, которые необходимы для управления механизмом: переключатели режимов работы, задатчики технологических параметров, кнопки управления и индикаторы.

Кроме этого, на схеме изображают датчики, которые позволяют контролировать состояния механизма и обеспечивают безопасность его работы.

К таким устройствам относятся:

– датчики ограничения перемещений;

– датчики контроля скорости;

– датчики превышения веса;

– датчики превышения температуры;

– датчики превышения давления и т.д.

Все устройства и элементы структурной схемы соединяются линиями, которые отражают взаимодействие элементов системы, потоки передачи информационных и управляющих сигналов.

Выбор технических средств

Важным этапом проектирования системы управления технологическими установкам является выбор управляющего устройства, информационных и исполнительных элементов.

Основой проектируемой системы управления должен быть программируемый логический контроллер (ПЛК). Могут быть использованы ПЛК производства российских и зарубежных фирм, например, Овен, Siemens, Omron и др.

Выбор модели ПЛК, состава и количества модулей ввода-вывода осуществляется в зависимости от сложности решаемой задачи, количества требуемых входов и выходов, электрических характеристик сигналов датчиков и исполнительных устройств, условий эксплуатации, стоимости и других параметров.

Для выполнения задач контроля состояния и параметров технологических установок выбираются информационные устройства, к которым в первую очередь относятся различные датчики (положения, скорости и т. п.), установленные на объекте, а также элементы пульта оператора (кнопки управления, переключатели и т. п.).

Исполнительные устройства выбирают по предварительно рассчитанным основным параметрам с учетом исполнения, способа монтажа, условий эксплуатации, степени защиты от воздействий окружающей среды, надежности и других характеристик. Для управления исполнительными устройствами необходимо выбрать элементы коммутации и защиты (реле, пускатели, контакторы и т. п.), а также, если необходимо, устройства для их питания (усилители, преобразователи и т. п.).

В процессе проектирования анализируют технические характеристики двух – трех альтернативных вариантов выбираемых элементов системы управления и по результатам анализа характеристик и параметров средств автоматизации обосновывают выбор конкретных моделей элементов системы.

Разработка схемы электрической принципиальной

На базе выбранных средств разрабатывают электрическую принципиальную схему. На данной схеме показываются все электрические элементы и устройства, все связи между ними, а также элементы подключения, которыми заканчиваются входные и выходные цепи (разъемы, контакты, клеммники и т. п.).

Выполнение электрической принципиальной схемы осуществляется в соответствии с действующими стандартами на оформление схем, на условное графическое и буквенное обозначение электрических элементов и линий соединения между ними.

При разработке схемы подключения ПЛК важно учитывать требования фирм-разработчиков ПЛК в части подключения датчиков и исполнительных устройств к входам и выходам контроллера, а также цепей их электропитания.

Разработка алгоритма управления

Типовая последовательность разработки алгоритма управления содержит следующие этапы:

– анализ работы технологической установки;
– выделение задач, решаемых ПЛК в процессе управления;
– определение последовательности выполнения задач;
– описание последовательности выполнения задач в виде алгоритма;
– описание последовательности решения каждой задачи в виде алгоритма;

– составление общего алгоритма управления.

При анализе работы технологической установки необходимо:

– выделить режимы работы объекта управления и определить особенности управления в каждом из них;

– определить порядок взаимодействия оператора с устройством управления;

– определить последовательность обработки сигналов датчиков и формирования управляющих сигналов;

– выделить возможные аварийные ситуации и определить действия при их наступлении;

– определить возможности диагностики работы оборудования и определить действия при появлении неисправностей.

Задачи, которые решаются в процессе управления, могут выполняться в определенной последовательности или параллельно, могут иметь связи через переменные или выполняться независимо. Все это должно быть отражено на схеме алгоритма управления.

При составлении алгоритма решения каждой задачи необходимо определить условия выполнения задачи, обрабатываемые входные сигналы и используемые данные других задач, формируемые управляющие выходные сигналы или переменные для других задач.

Если алгоритм управления получается сложным, его можно представить его в виде алгоритмов управления в каждом режиме или алгоритмов решения отдельных задач.

Разработка программы управления ПЛК

Завершающим этапом проектирования системы управления является написание программы управления для программируемого контроллера.

Накопленный многими фирмами опыт программирования ПЛК обобщен в виде стандарта IEC 61131, где определены пять языков программирования контроллеров: SFC – язык последовательных функциональных схем, LD – язык релейных диаграмм, FBD – язык функциональных блоковых диаграмм, ST – язык структурированного текста, IL – язык инструкций.

Программу управления для ПЛК можно разрабатывать на любом стандартном языке программирования в среде одного из специализированных систем программирования (пакета прикладных программ).

Примерами таких комплексов программирования являются CoDeSys фирмы 3S Smart Software Solutions, Step 7 и LOGO!Soft Comfort фирмы Siemens, Multiprog фирмы Klopper und Wiege Software GmbH, CX-Programmer фирмы Omron, а также UltraLogik и ISaGRAF и др.

Задание к лабораторной работе

Вариант 1. Разработать систему управления электроприводом ленточного конвейера, обеспечивающего плавный пуск.

Вариант 2. Разработать систему управления электроприводом ленточного конвейера, подачу заготовок в заданную позицию.

Вариант 3. Разработать систему управления электроприводом вентилятора для поддержания заданной температуры в помещении.

Вариант 4. Разработать систему управления электроприводом насоса с поддержанием давления в магистрали.

Вариант 5. Разработать систему управления электроприводом механизма подъема мостового крана.

Вариант 6. Разработать систему управления электроприводом механизма передвижения тележки мостового крана.

Вариант 7. Разработать систему управления электроприводом лифта.

Вариант 8. Разработать систему управления электроприводом механизма передвижения мостового крана.

Порядок проведения лабораторной работы

1 Определить требования к электроприводу и системе управления заданного типового механизма.

2 Выбрать элементы системы управления электроприводом механизма.

3 Определить состав дополнительных датчиков и исполнительных устройств, необходимых для надежной и безопасной работы системы.

4 Выбрать элементы пульта оператора, которые используются для управления системой.

5 Разработать структурную схему системы управления.

6 Составить блок-схему алгоритма управления заданным механизмом.

7 Разработать схему подключения информационных и исполнительных элементов к программируемому контроллеру.

8 Составить таблицу входных и выходных сигналов контроллера с указанием подключаемого элемента, типа сигнала, символьного обозначения, адреса.

9 Составить управляющую программу, реализующую управление механизмом.

10 Проверить работу управляющей программы в режиме эмуляции.

11 Записать программу в память контроллера и проверить ее выполнение.

12 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующие разделы.

1 Титульный лист установленного образца.

2 Цель работы.

3 Структурная схема системы управления.

4 Блок-схема алгоритма управления.

5 Схема подключения элементов управления к программируемому логическому контроллеру.

6 Таблица сигналов контроллера и внутренних переменных с указанием символьного обозначения, типа сигнала и адреса.

7 Листинг программы.

8 Вывод по лабораторной работе.

Список литературы

1 **Петров**, **И. В**. Программируемые контроллеры. Стандартные языки и инструменты / И. В. Петров. – Москва: СОЛОН-Пресс, 2003. – 256 с.

2 Руководство пользователя по программированию ПЛК в CoDeSys 2.3. – Смоленск: Пролог, 2008. – 452 с.

3 **Минаев**, **И. Г**. Программируемые логические контроллеры. Практическое руководство для начинающего инженера / И. Г. Минаев, В. В. Самойленко. – Ставрополь: АРГУС, 2009. – 100 с.

4 Общие сведения о CoDeSys [Электронный ресурс]. – Режим доступа: http://www.3s-software.ru /publications. – Дата доступа : 06.05.2017.

5 Каталог продукции фирмы ОВЕН [Электронный ресурс]. – Режим доступа : http://www.owen.ru/catalog. – Дата доступа: 16.05.2017.