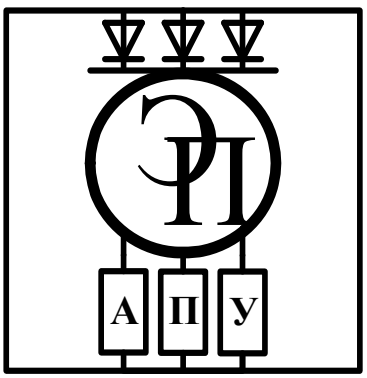
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Электропривод и АПУ»

СИСТЕМЫ ПРОГРАММНОГО УПРАВЛЕНИЯ

Методические рекомендации к лабораторным работам для студентов специальностей 1-53 01 05 «Автоматизированные электроприводы» и 6-05-0713-04 «Автоматизация технологических процессов и производств» очной и заочной форм обучения

Часть 1



Могилев 2025

УДК 000.4
ББК 32.295
С40

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Электропривод и АПУ» «30» августа 2024 г., протокол № 1

Составитель канд. техн. наук, доц. Л. Г. Черная

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

В методических рекомендациях к лабораторным работам для студентов специальностей 1-53 01 05 «Автоматизированные электроприводы» и 6-05-0713-04 «Автоматизация технологических процессов и производств» дневной и заочной форм обучения изложены методы разработки программ для систем управления на базе программируемых логических контроллеров.

Учебное издание

СИСТЕМЫ ПРОГРАММНОГО УПРАВЛЕНИЯ

Часть 1

Ответственный за выпуск	А. С. Коваль
Корректор	А. А. Подошевко
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 26 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

Введение

Лабораторные занятия по дисциплине «Системы программного управления» прививают студентам навык программирования логических контроллеров, облегчают восприятие и понимание основных теоретических положений, способствуя их более глубокому усвоению.

Методические рекомендации соответствуют программе курса «Системы программного управления». Они служат основой для самостоятельной подготовки и проведения лабораторных работ с последующим оформлением и анализом результатов и предусматривают изучение теоретического материала по учебной и справочной литературе.

К выполнению лабораторных работ студенты допускаются после ознакомления с правилами и инструкцией по технике безопасности и проведения инструктажа по безопасным методам работы с программируемыми логическими контроллерами, компьютерами с оформлением соответствующей записи в журнале.

Для получения допуска к очередным занятиям студенты предварительно изучают содержание лабораторной работы, рекомендации к выполнению задания и представляют законченный отчет по предыдущей работе.

1 Лабораторная работа № 1. Изучение устройства и принципов работы контроллера программируемого логического

Цель работы: изучение технических характеристик, принципа работы и структурных схем программируемых логических контроллеров (ПЛК).

Задание

1 Изучить назначение, технические характеристики, состав и конструкцию контроллера.

2 Изучить устройство контроллера, назначение органов управления.

3 Изучить варианты подключения контроллера к компьютеру.

4 Изучить меры безопасной работы, порядок установки и подготовки контроллера к работе.

5 Изучить схему подключения исполнительных устройств и датчиков управляемого объекта к контроллеру.

1.1 Объект изучения

Вариант 1. Контроллеры с HMI для локальных систем автоматизации ОВЕН ПЛК63/ПЛК73.

Вариант 2. Контроллеры для малых систем автоматизации ОВЕН ПЛК100 / ПЛК150 / ПЛК154.

Вариант 3. Моноблочные контроллеры с дискретными и аналоговыми входами/выходами для средних систем автоматизации ПЛК110[М02] / ПЛК110 / ПЛК160.

1.2 Рекомендации к выполнению задания

Для выполнения вариантов задания необходимо обратиться к Руководствам по эксплуатации фирмы ОВЕН (на компьютере: раздел 06 Документация, 01 PLC 100, 150,154; 02 PLC 110, 160; 03 PLC 63_73).

1 Контроллер программируемый логический ПЛК63. Руководство по эксплуатации.

2 Контроллер программируемый логический ПЛК110. Руководство по эксплуатации.

3 Контроллер программируемый логический ПЛК160. Руководство по эксплуатации.

4 Контроллер программируемый логический ПЛК154. Руководство по эксплуатации.

Структура и устройство ПЛК.

Первые программируемые логические контроллеры появились в 70-х гг. прошлого века и предназначались для использования в сфере промышленной

автоматики для решения технологических задач, которые описывались преимущественно логическими уравнениями. Это позволило ПЛК заменять в системах управления блоки релейной автоматики и устройства жесткой логики на интегральных микросхемах малой и средней степени интеграции. Название «Программируемый логический контроллер» происходит от английского названия Programmable Logic Controller (PLC).

Сегодня ПЛК − это специализированное микропроцессорное устройство с широкими функциональными возможностями. Специальное проблемно-ориентированное программное обеспечение ПЛК позволяет реализовать не только алгоритмы программно-логического управления, но и осуществлять регулирование в замкнутых системах автоматического управления, производить сбор, обработку, хранение и передачу информации. Конструкция ПЛК позволяет эксплуатировать устройство в производственных условиях.

Кроме этого, ПЛК отличается универсальностью структуры, программируемостью и возможностью решения определенного класса задач при управлении технологическим объектом в режиме реального времени.

ПЛК имеет конечное количество входов и выходов, к которым подключены датчики и исполнительные устройства, обеспечивающие связь с технологическим объектом управления (рисунок 1.1).

Рисунок 1.1 − Связь ПЛК с датчиками и исполнительными механизмами

Принцип работы ПЛК несколько отличается от «обычных» микропроцессорных устройств. Программное обеспечение универсальных контроллеров состоит из двух частей. Первая часть − это системное программное обеспечение. Проводя аналогию с компьютером можно сказать, что это операционная система, т. е. управляет работой узлов контроллера, взаимосвязи составляющих частей, внутренней диагностикой. Системное программное обеспечение ПЛК расположено в постоянной памяти центрального процессора и всегда готово к работе. По включению питания ПЛК готов взять на себя управление системой уже через несколько миллисекунд.

Вторая часть программного обеспечения − это программа управления, которая разрабатывается при проектировании системы управления технологическим объектом и хранится в перепрограммируемой памяти. Управляющая

программа может изменяться в процессе эксплуатации оборудования, например при модернизации технологической установки.

ПЛК работают циклически по методу периодического опроса входных данных.

Рабочий цикл ПЛК включает четыре фазы:

1) опрос входов;
2) выполнение пользовательской программы;
3) установку значений выходов;
4) некоторые вспомогательные операции (диагностика, подготовка данных для отладчика, визуализации и т. д.).

Выполнение первой фазы обеспечивается системным программным обеспечением. После чего управление передается прикладной программе, той программе, которую вы сами записали в память. По этой программе контроллер делает то, что вы пожелаете, а по ее завершению управление опять передается системному уровню. За счет этого обеспечивается максимальная простота построения прикладной программы – ее создатель не должен знать, как производится управление аппаратными ресурсами. Необходимо знать с какого входа приходит сигнал и как на него реагировать на выходах.

Очевидно, что время реакции на событие будет зависеть от времени выполнения одного цикла прикладной программы. Определение времени реакции – времени от момента события до момента выдачи соответствующего управляющего сигнала – поясняется на рисунке 1.2.

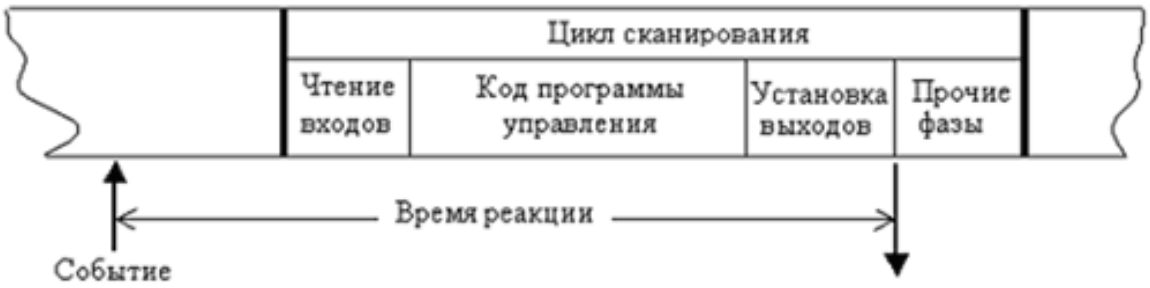


Рисунок 1.2 – Время реакции на событие

Обладая памятью, ПЛК, в зависимости от предыстории событий, способен реагировать по-разному на текущие события. Возможности перепрограммирования, управления по времени, развитые вычислительные способности, включая цифровую обработку сигналов, поднимают ПЛК на более высокий уровень в отличие от простых комбинационных автоматов.

Существует три вида входов ПЛК: дискретные, аналоговые и специальные.

Один дискретный вход ПЛК способен принимать один бинарный электрический сигнал, описываемый двумя состояниями – включен или выключен. Все дискретные входы (общего исполнения) контроллеров обычно рассчитаны на прием стандартных сигналов с уровнем 24 В постоянного тока. Типовое значение тока одного дискретного входа (при входном напряжении 24 В) составляет около 10 мА.

Аналоговый электрический сигнал отражает уровень напряжения или тока, соответствующий некоторой физической величине в каждый момент времени. Это может быть температура, давление, вес, положение, скорость, частота и т. д. Поскольку ПЛК является цифровой вычислительной машиной, аналоговые входные сигналы обязательно подвергаются аналого-цифровому преобразованию (АЦП). В результате, образуется дискретная переменная определенной разрядности. Как правило, в ПЛК применяются 8…12-разрядные преобразователи, что в большинстве случаев, исходя из современных требований по точности управления технологическими процессами, является достаточным. Кроме этого, АЦП более высокой разрядности не оправдывают себя, в первую очередь из-за высокого уровня индустриальных помех, характерных для условий работы контроллеров.

Стандартные дискретные и аналоговые входы ПЛК способны удовлетворить большинство потребностей систем промышленной автоматики. Необходимость применения специализированных входов возникает в случаях, когда непосредственная обработка некоторого сигнала программно затруднена, например, требует много времени.

Наиболее часто ПЛК оснащаются специализированными счетными входами для измерения длительности, фиксации фронтов и подсчета импульсов. Например, при измерении положения и скорости вращения вала очень распространены устройства, формирующие определенное количество импульсов за один оборот – поворотные шифраторы. Частота следования импульсов может достигать нескольких мегагерц. Даже если процессор ПЛК обладает достаточным быстродействием, непосредственный подсчет импульсов в пользовательской программе будет весьма расточительным по времени. Здесь желательно иметь специализированный аппаратный входной блок, способный провести первичную обработку и сформировать, необходимые для прикладной задачи величины.

Вторым распространенным типом специализированных входов являются входы, способные очень быстро запускать заданные пользовательские задачи с прерыванием выполнения основной программы – входы прерываний.

Дискретный выход также имеет два состояния – включен и выключен. Они нужны для управления: электромагнитных клапанов, катушек, пускателей, световые сигнализаторы и т. д. В общем сфера их применения огромна и охватывает почти всю промышленную автоматику.

Конструктивно ПЛК подразделяются на моноблочные, модульные и распределенные. Моноблочные имеют фиксированный набор входов/выходов.

В модульных контроллерах модули входов/выходов устанавливаются в разном составе и количестве в зависимости от предстоящей задачи. В распределенных системах модули или даже отдельные входы/выходы, образующие единую систему управления, могут быть разнесены на значительные расстояния.

1.3 Порядок проведения лабораторной работы

При выполнении работы следует использовать руководство по эксплуатации изучаемого контроллера.

1 Изучить назначение, технические характеристики, состав и конструкцию контроллера.

2 Изучить устройство контроллера, назначение органов управления.

3 Изучить варианты подключения контроллера к компьютеру.

4 Изучить меры безопасной работы, порядок установки и подготовки контроллера к работе.

5 Изучить схему подключения исполнительных устройств и датчиков управляемого объекта к контроллеру.

6 Подготовить ответы на контрольные вопросы.

7 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Назначение и основные технические параметры контроллера (согласно варианту задания).

4 Схемы подключения исполнительных устройств и датчиков к контроллеру.

5 Вывод по лабораторной работе.

6 Ответы на контрольные вопросы.

Контрольные вопросы

1 Какие имеются модификации изучаемого контроллера?

2 Какая длительность рабочего цикла контроллера и чем она определяется?

3 Какие датчики и исполнительные устройства могут подключаться к каналам ввода/вывода контроллера?

4 Что происходит при нажатии кнопки «Сброс»?

5 Что означает безопасное состояние выходов?

6 Каким образом можно определить состояния контроллера?

7 Где размещается управляющая программа пользователя?

8 Для чего предназначена Retain-память?

9 Какие способы программирования предусмотрены в контроллере?

10 Какие устройства контроллера можно использовать в управляющих программах?

11 Какие интерфейсы обмена информацией имеются в контроллере?

12 Какая линия COM-порта компьютера используется при программировании контроллера?

13 Как осуществить подключение нескольких контроллеров для совместной работы?

14 Каким образом осуществляется подключение питания к контроллеру?

2 Лабораторная работа № 2. Изучение среды CoDeSys для программирования контроллеров

Цель работы: приобретение навыков работы в среде программирования CoDeSys.

Задание

1 Описание структуры проекта и основных правил его создания.

2 Описание компонентов POU, которые могут содержаться в проекте.

3 Описание операндов, используемых в системе программирования.

4 Описание классов переменных, используемых в компонентах проекта.

5 Создание окон проекта в среде CoDeSys согласно указанного преподавателем типа контроллера программируемого логического (см. варианты задания лабораторной работы № 1).

2.1 Рекомендации к выполнению задания

Для выполнения задания необходимо обратиться к руководству фирмы ОВЕН (на компьютере: раздел 6 Документация, 04 CoDeSys) «Руководство пользователя по программированию ПЛК в CoDeSys 2.3». (UserManual_V23_RU.pdf) и оперативной справочной системе CoDeSys.

2.1.1 Общие сведения о системе программирования CoDeSys.

CoDeSys – универсальная среда разработки программ для программируемых логических контроллеров (ПЛК) на основе языков стандарта МЭК 61131-3. Название системы расшифровывается как Controller Development System и выпускается она немецкой фирмой 3S-Smart Software Solutions GmbH.

Первая версия CoDeSys вышла в 1994 г., в настоящее время доступна третья версия пакета, в которой поддерживается объектно-ориентированное программирование. В состав комплекса программирования ПЛК входят две обязательные части: среда разработки программ и система исполнения. Среда разработки программ CoDeSys функционирует на персональном компьютере (ПК) и не имеет конкретной аппаратной привязки, поэтому с её помощью можно создавать программы для любых контроллеров. Система исполнения (CoDeSys SP) функционирует в контроллере и устанавливается его производителем. Она обеспечивает загрузку кода программы, ее исполнение, а также отладочные функции.

Среда разработки программ CoDeSys является бесплатной и ее можно скачать с сайта производителя. Компания 3S лицензирует только системы

исполнения. Существует русифицированная версия CoDeSys. Этот программный пакет может быть установлен на компьютер, работающий под управлением операционной системы Windows. Для загрузки написанной программы в какой-либо конкретный ПЛК необходим target-файл (файл целевой платформы), обычно поставляемый производителем контроллера.

В этом файле находится информация о ресурсах данного контроллера, в том числе о входах и выходах, типах и расположении данных в памяти и т. д. Для инсталляции target-файлов служит программа InstallTarget, которая устанавливается на компьютер вместе с пакетом CoDeSys.

CoDeSys применяется для программирования многими фирмами, в частности, WAGO, ABB, Beckhoff и т. д. В России CoDeSys широко применяется фирмой ОВЕН (Москва) и Пролог (Смоленск).

Одним из преимуществ CoDeSys является наличие в нем режима эмуляции, что позволяет отлаживать программы непосредственно на компьютере, не загружая их в контроллеры и не привлекая на стадии отладки работающее технологическое оборудование. Это свойство позволяет также широко использовать CoDeSys в учебном процессе.

В настоящее время существует некоммерческая организация CoDeSys Automation Alliance (CAA) – объединение компаний-производителей ПЛК, поддерживающих CoDeSys, в которое входят более семидесяти фирм.

2.1.2 Структура проекта CoDeSys.

При создании (открытии) проекта в CoDeSys появляется окно проекта (рисунок 2.1), в верхней части которого находится меню, а под ним − панель инструментов с учетом выбранного языка. В левой части окна находится вертикальная панель, под которой расположены четыре вкладки. Справа, последовательно сверху вниз, находятся область объявления переменных, область кода выбранного языка и область сообщений об ошибках и предупреждениях при компиляции проекта.

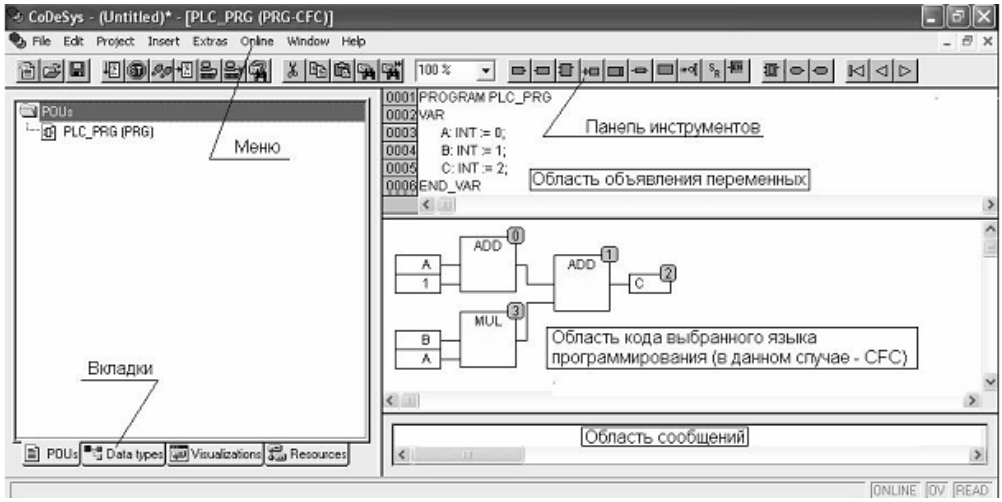


Рисунок 2.1 − Окно проекта CoDeSys

Вкладки проекта CoDeSys отдельно показаны на рисунке 2.2.

Рисунок 2.2 − Вкладки CoDeSys

Первая слева вкладка – POU, т. е. компоненты проекта. При входе в эту вкладку на находящейся над ней вертикальной панели (Object Organizer − список объектов) отображаются все компоненты проекта (на рисунке 2.1 компонент только один – программа (PRG) с именем PLC_PRG).

При щелчке левой кнопкой мыши по имени POU можно войти в редактор кода, соответствующий языку данного компонента. При щелчке правой кнопкой мыши по имени POU или по пустому полю вертикальной панели появляется меню, с помощью которого можно добавить новый компонент POU, переименовать или удалить уже имеющийся и т. д.

Следующая вкладка – Data types (Типы данных) – предназначена для определения пользовательских типов данных, например, перечисляемых типов данных и структур.

Вкладка Visualizations (Визуализации) предназначена для создания и просмотра операторских интерфейсов (визуализаций). При входе в эту вкладку на левой панели отображаются имеющиеся визуализации. При щелчке левой кнопкой мыши по имени визуализации можно войти в нее, при щелчке правой кнопкой − появляется контекстное меню.

Вкладка Resouces (Ресурсы) позволяет выполнить действия, связанные с ресурсами контроллера. Через нее можно войти в «Настройку целевой платформы», выполнить конфигурирование входов и выходов ПЛК, изменить прошивку ПЛК, войти в «Менеджер библиотек» и т. д. Работа с вкладками Resouces рассматривается ниже.

Часть пунктов меню, находящегося в верхней части окна проектов, универсальна для всех языков, встроенных в CoDeSys, часть из них специфична для конкретного языка. Рассмотрим здесь наиболее важные пункты, общие для всех языков. В пункте меню «Проект» (Project) имеются два первых элемента «Компилировать» (Build) и «Компилировать все» (Rebuild all) (рисунок 2.3).

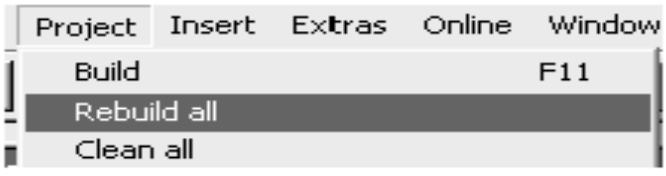


Рисунок 2.3 − Пункт меню «Проект»

При использовании элемента «Компилировать» компилируется только новая часть проекта. При обращении к элементу «Компилировать все» компилируется весь проект. В случае успешной компиляции в области сообщений появляется запись «0 Error(s), 0 Warning(s)» − нет ошибок и нет

предупреждений. Проект, в котором есть ошибки, не может быть запущен до их полного исправления. При наличии предупреждений проект может работать.

По окончании компиляции можно выполнить загрузку и пуск программы через раздел меню «Онлайн» (Online) (рисунок 2.4). Если предполагается загружать программу в ПЛК, имеющем связь с ПК, то сначала с помощью элемента «Подключение» (Login) производится их логическое соединение, затем можно загрузить программу, используя команду «Загрузить». Если программа является новой, выполняется действие «Загрузить новую программу». Если компьютер не соединен с ПЛК, то в меню «Онлайн» можно выбрать «Режим эмуляции» и выполнить программу без загрузки в ПЛК. Программа не начинает работу до принудительного пуска.

Пуск программы происходит при обращении к элементу «Старт» (Run) меню «Онлайн» (рисунок 2.5) или к одноименной иконке панели инструментов.

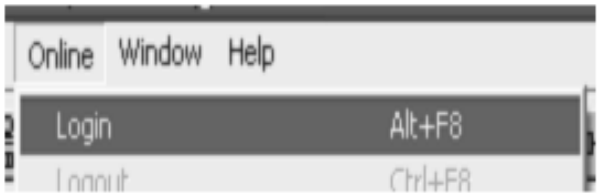


Рисунок 2.4 − Пункт меню «Онлайн»

Рисунок 2.5 − Пункт меню «Онлайн», запуск программы

Останов программы происходит при обращении к элементу «Стоп» (Stop) меню «Онлайн» (рисунок 2.6) или к одноименной иконке панели инструментов. В этом случае работа программы прекращается, но компьютер не отключается от контроллера. При необходимости отключиться от компьютера используется подпункт «Отключение» (Logout) (см. рисунок 2.6). Необходимо отметить, что если отключить компьютер от контроллера без останова программы, то в контроллере программа продолжит выполняться, что является нормальным режимом для производственных условий.

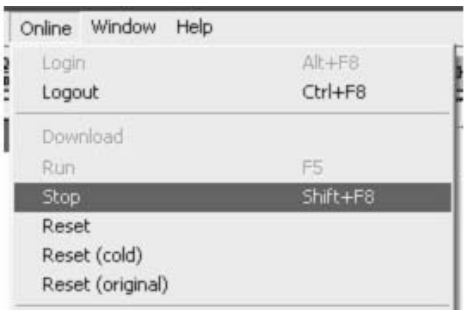


Рисунок 2.6 − Пункт меню «Онлайн», останов программы и отключение от контроллера

Пункт «Сброс» (Reset) перезапускает программу, заново инициализируя все переменные, за исключением RETAIN. Если начальные значения переменных определены, они будут присвоены переменным; если нет, переменным

присваиваются стандартные значения по умолчанию (например, 0 для целых типов). Данный сброс аналогичен выключению и включению питания ПЛК.

При выборе пункта «Сброс (холодный)» − Reset (cold) выполняются те же действия, что и по команде «Сброс» (Reset), и дополнительно присваиваются начальные значения переменным, хранящимся в энергонезависимой области памяти RETAIN.

Пункт «Сброс (заводской)» − Reset (original) удаляет программу пользователя, восстанавливая состояние контроллера, в котором он поступает с завода-изготовителя.

В некоторых ситуациях требуется изменить значения переменных, используемых в программе, не прекращая ее работы. Для этого необходимо открыть вкладку POU и кликнуть левой кнопкой мыши переменную в работающей программе. Появится диалоговое окно, представленное на рисунке 2.7. В поле «Новое значение» (New Value) вводится требуемое значение. Затем следует обратиться к пункту «Записать значения» меню «Онлайн» или нажать F7 (рисунок 2.8), после чего в программе будет учитываться новое значение переменной.

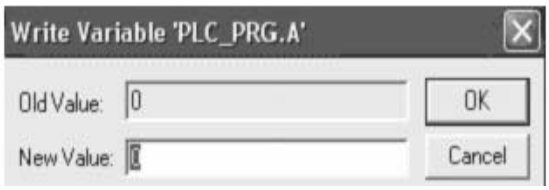


Рисунок 2.7 − Диалоговое окно для записи значений переменных

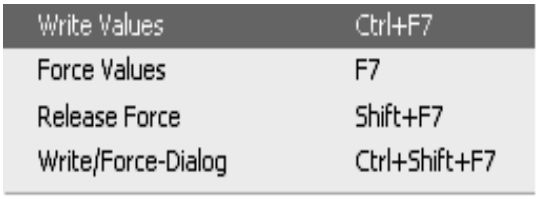


Рисунок 2.8 − Пункт меню «Онлайн», запись значений переменных

2.1.3 Создание окон проекта.

Проект в CoDeSys создается в виде файла xxx.pro с помощью компонентов организации программ, обозначаемых как POU (Program Organization Unit). Чтобы создать новый проект, следует войти в пункт меню File/New (Файл/Создать). Появится диалоговое окно Target Settings (Настройка целевой платформы) (рисунок 2.9), в котором необходимо указать тип контроллера, для которого создается проект. Если проект не предполагается загружать в контроллер, то выбирается пункт None. Чтобы тип контроллера появился в списке, необходимо предварительно установить его target-файл, как это будет рассмотрено ниже.

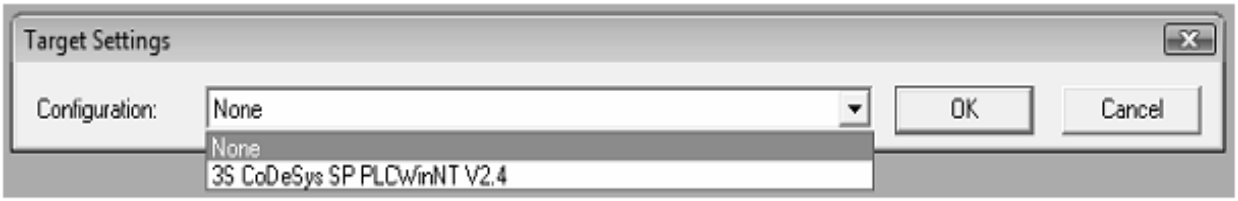


Рисунок 2.9 − Настройка целевой платформы

После выбора целевой платформы на экране появится окно New POU (Новый программный компонент (POU)) (рисунок 2.10), в котором требуется указать имя нового POU, выбрать его тип и язык реализации.

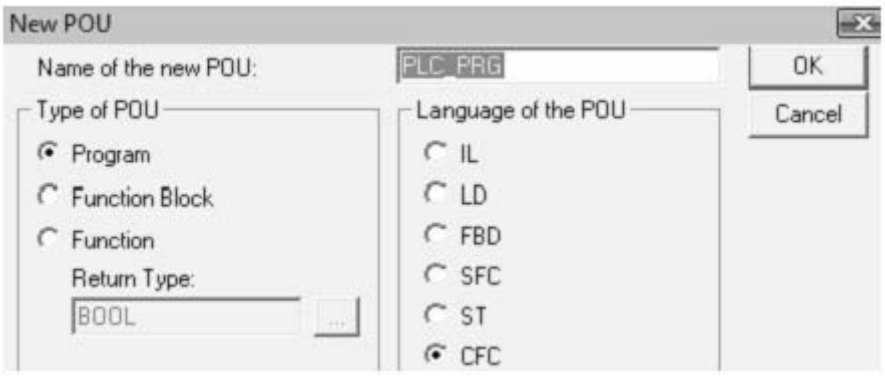


Рисунок 2.10 − Выбор вида программного компонента

В CoDeSys существует всего три типа POU: «Программа» (Program), «Функциональный блок» (Function Block) и «Функция» (Function). Если создаваемый проект предполагается загружать в контроллер, то в нем должен быть хотя бы один компонент «Программа» с именем PLC_PRG (это имя предлагается по умолчанию). Все остальные POU можно называть произвольно с учетом следующих требований: имя должно состоять только из латинских букв, цифр или знаков подчеркивания, при этом первым символом в имени должна быть буква. Имена POU в составе одного проекта не должны повторяться. Те же требования предъявляются к именам переменных и визуализаций.

Если создаваемый компонент является функцией, то нужно выбрать тип возвращаемого функцией значения и указать его в поле Return Type (тип возвращаемого значения). По умолчанию задается тип BOOL, т. е. логический. Можно нажать на кнопку слева от поля Return Type и выбрать тип из открывающегося списка. После ввода необходимой информации следует нажать кнопку «OK», при этом диалоговое окно New POU закроется и откроется редактор кода выбранного языка программирования.

2.1.4 Переменные. Объявление переменных.

Согласно общепринятому определению переменная – это поименованная или иным образом адресуемая область памяти. Имя (или адрес) этой области можно использовать для доступа к находящимся в ней данным, т. е. к значению переменной.

У любой переменной имеется имя, тип и область действия; переменные в CoDeSys могут быть локальными и глобальными. Областью действия локальной переменной является тот программный компонент, внутри которого она объявлена. Если переменная объявлена внутри функции, то область её действия – эта функция и т. д. Область действия глобальной переменной – весь проект.

В CoDeSys все переменные, связанные со входами или выходами контроллера, являются глобальными по умолчанию. В прочих случаях лучше

не пользоваться глобальными переменными без необходимости, т. к. сложно проследить, как изменяется значение данной переменной при работе программы в целом. В CoDeSys все используемые переменные должны быть объявлены. Локальные переменные объявляются в области объявления переменных редактора кода (рисунок 2.11).

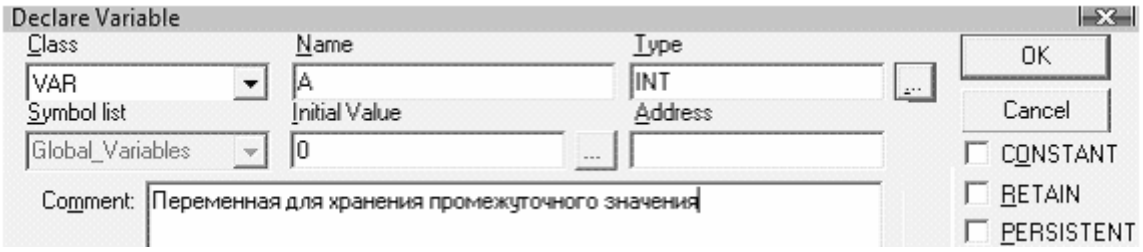


Рисунок 2.11 − Окно объявления переменной

CoDeSys отслеживает переменные, используемые в тексте, и, обнаружив новые необъявленные переменные, предлагает объявить их. При этом выводится окно объявления переменной «Declare Variable» (см. рисунок 2.11).

В этом окне необходимо ввести имя и тип переменной. По умолчанию указывается тип BOOL. Если переменная имеет другой тип, следует нажать на кнопку справа от поля Type. Откроется окно «Ассистент ввода» (Input assistant), в котором можно выбрать требуемый тип (рисунок 2.12). В поле Class можно оставить значение по умолчанию (при создании пользовательского функционального блока в этом поле необходимо указать, является ли переменная входной, выходной или внутренней). В поле Address ничего вводить не следует.

В поле Initial value (начальное значение) вводится начальное значение переменной, его можно не изменять. В поле Comment можно ввести комментарий. После нажатия кнопки «OK» введенные данные отображаются в области объявления переменных редактора кода (рисунок 2.13).

Рисунок 2.12 − Ассистент ввода

Рисунок 2.13 − Область объявления переменных редактора кода

Содержание области объявления переменных можно создать или заменить вручную.

Объявления лоальных переменных для компонента PROGRAM PLC_PRG всегда пишутся между ограничителями VAR и END_VAR.

Глобальные переменные не объявляют в области редактора кода (объявляют в Ресурсах). Имя глобальной переменной не должно нигде повторяться, в том числе и для локальной переменной. Иначе глобальная переменная будет не видна в том компоненте, где объявлена одноименная локальная, что может привести к неправильной работе программы. В таблице 2.1 приведены основные типы данных, используемые при составлении программ в среде CoDeSys и их краткие описания.

Таблица 2.1 − Основные типы данных, применяемые при составлении программ

Имя типа	Описание типа
INT	Целые числа в диапазоне от -32768 до 32767
REAL	Числа с плавающей точкой, 32 байта
BOOL	Логический тип данных, значения TRUE или FALSE
WORD	Целые числа в диапазоне от 0 до 65535
TIME	Время (Например, T#5m30s0ms – 5 мин 30 с)
STRING	Строка от 1 до 255 символов

2.1.5 Компоненты организации программ.

Проект в CoDeSys может содержать три вида компонентов организации программ (POU): программа (Program), функциональный блок (Function Block) и функция (Function).

Программа может возвращать несколько значений, ее можно вызывать из других программ или из функциональных блоков, из функций программы вызывать нельзя.

Функции и функциональные блоки могут быть созданы пользователем самостоятельно или взяты из библиотек. Функция отображает множество значений входных параметров на один выход, а функциональный блок – на множество выходов. Чтобы создать пользовательский компонент, следует вызвать вкладку POU, щелкнуть правой кнопкой мыши по левой вертикальной области и в появившемся контекстном меню (рисунок 2.14) выбрать пункт Add object (Добавить объект). Появится окно, показанное на рисунке 2.15 (Выбор нового программного компонента).

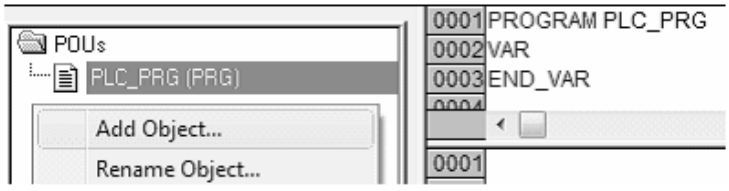


Рисунок 2.14 − Добавление нового пользовательского компонента

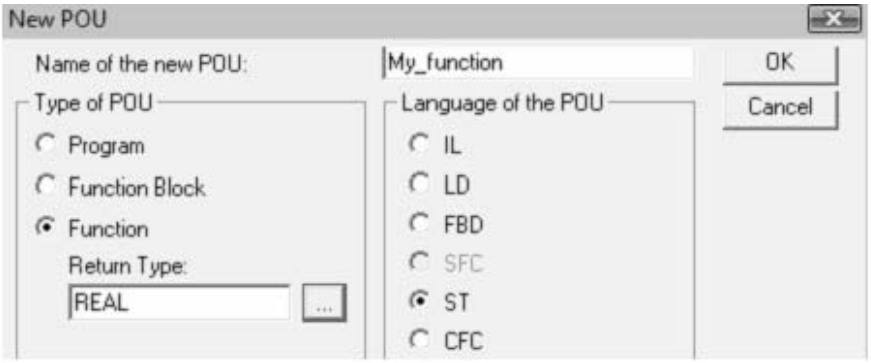


Рисунок 2.15 − Создание нового программного компонента Function

Если новый пользовательский компонент – это функция, то необходимо указать не только имя, тип нового компонента и язык реализации, но и тип возвращаемого значения (Return type). В примере на рисунке 2.15 тип возвращаемого значения – Real. При нажатии кнопки справа от поля ввода появится окно ассистента ввода, в котором можно выбрать требуемый тип. Кроме того, его можно ввести вручную. После создания нового программного компонента появляется окно редактора кода для выбранного языка. В рассматриваемом примере для функции с именем My_function и типом возвращаемого значения Real область объявления переменных показана на рисунке 2.16. Здесь в области объявления переменных для пользовательской функции есть две подобласти: VAR_INPUT…END_VAR и VAR…END_VAR. При объявлении переменных необходимо указать, к какому классу (VAR или VAR_INPUT) относится объявляемая переменная (рисунок 2.17).

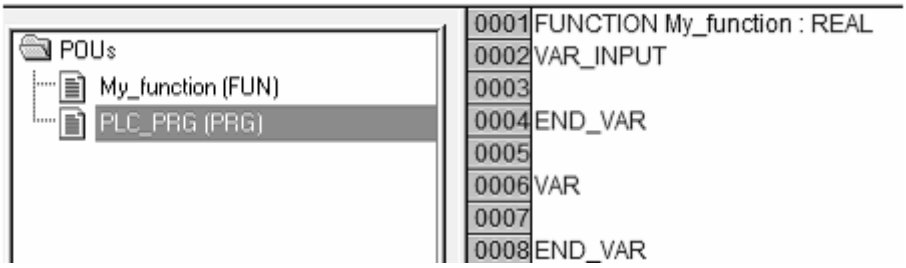


Рисунок 2.16 − Область объявления переменных для компонента Function

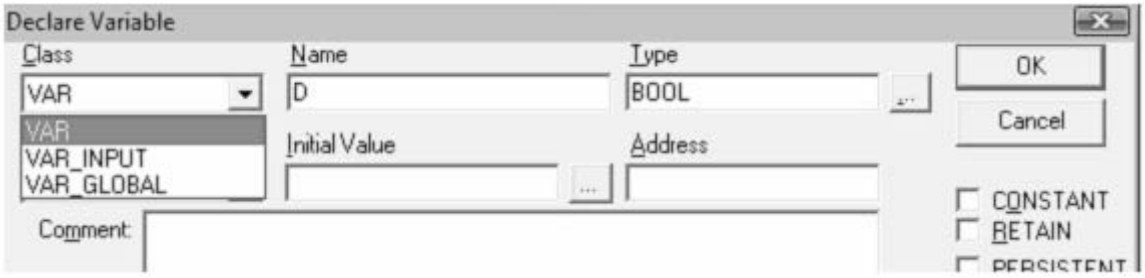


Рисунок 2.17 − Окно объявления переменных для компонента Function

Если переменная относится к классу VAR_INPUT, то эта переменная будет соответствовать одной из независимых переменных данной функции, т. е. одному из входных параметров. Если переменная относится к классу VAR, то она будет внутренней переменной данного компонента. Выходу, т. е. собственно значению функции, соответствует переменная с именем, совпадающим с именем функции.

Если новый компонент является функциональным блоком, то тип возвращаемого значения не указывается, т. к. у функционального блока может быть не один выход. Переменные, используемые внутри пользовательского функционального блока, могут относиться к классам VAR, VAR_INPUT и VAR_OUTPUT. Переменные из класса VAR_INPUT соответствуют входам блока, переменные из класса VAR_OUTPUT – выходам блока, а переменные из класса VAR – внутренние переменные.

Имена входов пользовательского блока будут соответствовать именам переменных класса VAR_INPUT, имена выходов – именам переменных класса VAR_OUTPUT. При этом совершенно не обязательно, чтобы имена переменных, связываемых с входами или выходами функционального блока извне, совпадали с именами его входов и выходов.

Все применяемые в программе пользовательские функциональные блоки и функциональные блоки из подключаемых библиотек должны иметь уникальные имена. К ним предъявляются такие же требования, как и к именам переменных. Имя функционального блока записывается на месте трех вопросительных знаков, находящихся над блоком.

Функциональный блок может быть написан на любом языке, встроенном в CoDeSys, функция – на любом языке, кроме SFC.

2.1.6 Установка target-файлов.

С помощью комплекса CoDeSys можно программировать любой контроллер, в котором его производителем установлена система исполнения CoDeSys SP. Кроме того, для данного типа контроллера должен быть target-файл от фирмы-производителя. В этом файле находится информация о ресурсах контроллера.

При необходимости написать для данного типа контроллеров проект с помощью CoDeSys следует установить соответствующий target-файл. Это делается с помощью программы InstallTarget, которую можно найти по следующему пути: Пуск/Все программы/3S Software/CoDeSys V2.3/InstallTarget (рисунок 2.18). При запуске программы InstallTarget появится диалоговое окно, показанное на рисунке 2.19. В левом поле Possible Targets указаны target-файлы, которые можно установить (с жесткого или съемного диска), их можно выбрать, нажав кнопку «Open». В правом поле Installed Targets указаны инсталлированные файлы.

Перед тем, как проинсталлировать файл, необходимо в поле Installation directory указать папку, в которую он будет установлен. Можно выбрать папку, нажав кнопку справа от поля ввода и отметив требуемую директорию в

открывшемся окне Choose Installation Directory (Выбор папки для инсталляции (рисунок 2.21)).

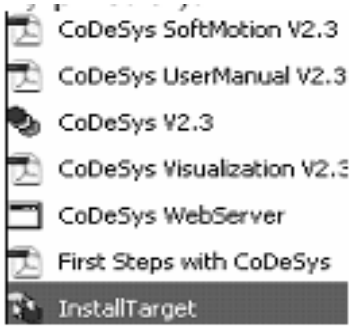


Рисунок 2.18 − Расположение программы InstallTarget

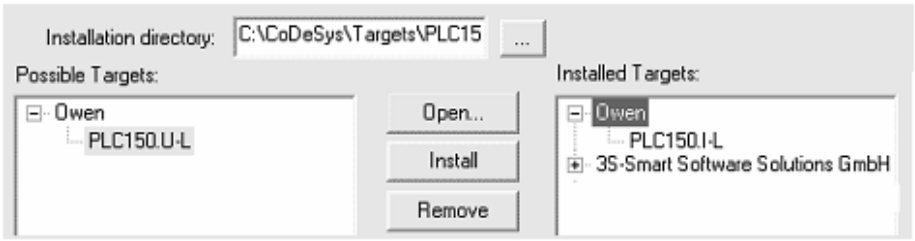


Рисунок 2.19 − Окно программы InstallTarget

На рисунке 2.20 для инсталляции выбран target-файл PLC150.U-L, который будет установлен в папку с адресом «C:\CoDeSys\Targets…». Для начала установки необходимо выбрать target-файл в левом окне и нажать кнопку «Install». В результате инсталляции данный файл появится в поле Installed Targets (см. рисунок 2.21). Если требуется удалить target-файл, следует выбрать его в правом окне и нажать кнопку «Remove».

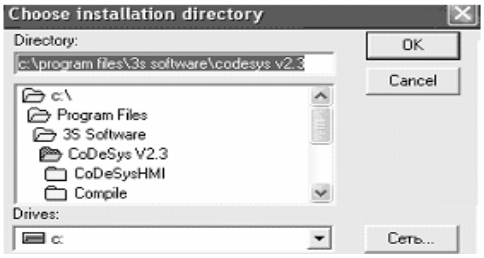


Рисунок 2.20 − Выбор папки для инсталляции target-файла

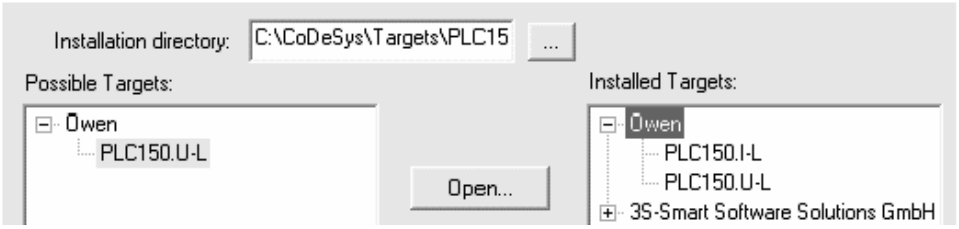


Рисунок 2.21 − Результат инсталляции target-файла

2.1.7 Настройка связи компьютера с контроллером.

Программирование промышленных логических контроллеров ОВЕН осуществляется с помощью персонального компьютера, который соединяется с контроллером специальным кабелем посредством одного из стандартных интерфейсов.

Программируемые контроллеры подключаются к компьютеру тремя способами − с помощью COM-порта (интерфейс RS-232) кабелем КС1, входящим в комплект поставки контроллера, с помощью интерфейса локальной вычислительной сети Ethernet восьмижильным кабелем на основе витых пар, либо с помощью USB-интерфейса (последнее имеют не все модели) стандартным кабелем типа A-B.

Настройка подключения происходит после создания нового проекта в среде CoDeSys.

После запуска программы CoDeSys на экране появится основное окно системы, в котором можно открыть уже созданный проект или создать новый.

Создать новый проект можно, нажав на крайнюю левую кнопку панели инструментов (под главным меню), или выбрав «Файл» − «Новый». После этого появится окно выбора целевой платформы для создания проекта, где нужно выбрать, для какого вида контроллеров будет создаваться проект.

После выбора целевой платформы (например, PLC150.U-L) и подтверждения выбора кнопкой «ОК» в текущем окне на экране появится новое окно, в котором будут содержаться основные параметры и настройки выбранной платформы ПЛК (адреса сегментов памяти, тактовая частота процессора, тип процессора, количества входов и выходов, значения некоторых системных переменных). Некоторые параметры пользователь может изменять.

После подтверждения настроек контроллера и выбора языка программирования можно настраивать связь с контроллером. Для этого выбрать в главном меню команду «Онлайн» − «Параметры связи», в результате появится окно (рисунок 2.22).

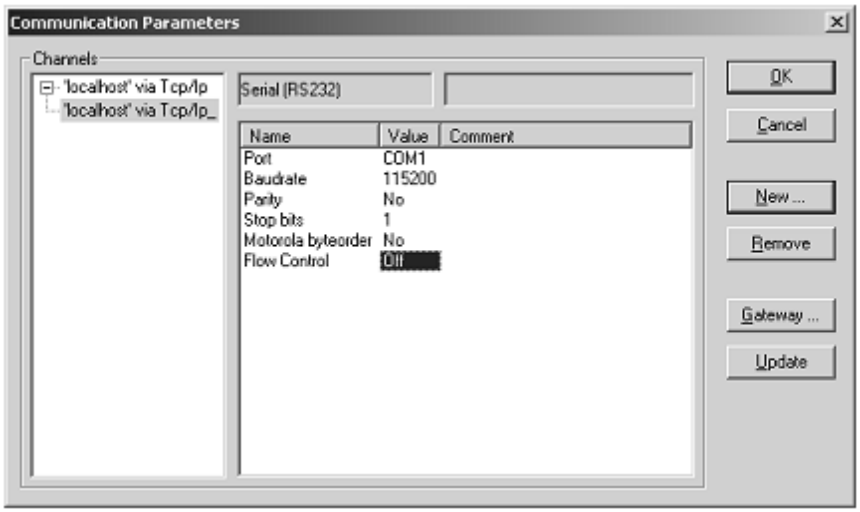


Рисунок 2.22 − Окно настройки связи с ПЛК

На рисунке 2.22 показаны уже существующие настройки. Если таковых не имеется и в иерархическом дереве слева есть только одна строка localhost via Tcp/Ip, то нужно создать новое подключение.

Для этого надо нажать кнопку «New», в правой части окна и в появившемся окне (рисунок 2.23) выбрать вид соединения с контроллером (в нашем случае − Serial (RS232)), затем нажать кнопку «OK».

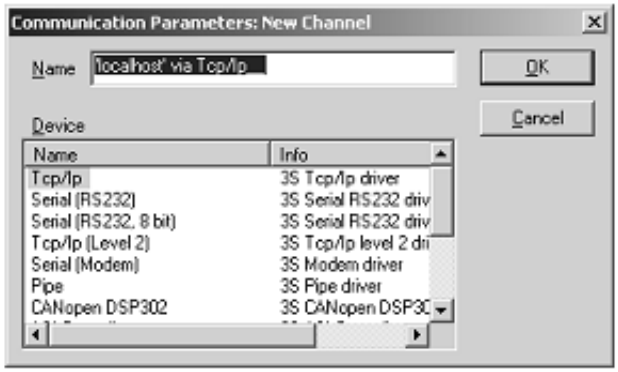


Рисунок 2.23 − Выбор соединения компьютера с ПЛК

В средней части окна настроек связи появится список параметров для выбранного соединения (см. рисунок 2.23). Значения параметров будут установлены по умолчанию. При необходимости для выбранной целевой платформы контроллера параметры должны быть скорректированы в соответствии с руководством по эксплуатации.

Проверить связь с контроллером можно, создав простейшую программу и записав ее в контроллер.

Перед загрузкой программы в контроллер нужно проверить его подключение к компьютеру с помощью кабеля (соединение интерфейса RS-232 на лицевой панели ПЛК с COM-портом ПК), при необходимости подключение производится с соблюдением требований, указанных в руководстве по эксплуатации.

2.1.8 Конфигурация памяти ввода / вывода контроллера.

Конфигурация ПЛК определяется составом аппаратных средств контроллера, совокупностью модулей, каналов ввода/вывода и значением их параметров. Информация о конфигурации хранится в области памяти ввода/вывода контроллера.

Область памяти ввода/вывода ПЛК (%I и %Q) включает дискретные и аналоговые входы и выходы, модули расширения (в том числе организующие обмен информацией между ПЛК и отдельными приборами и устройствами, связанными с ПЛК по сети). Внешние устройства обмениваются данными с пользовательской программой ПЛК также через эту область памяти.

Размер памяти ввода/вывода определяется типом лицензии CoDeSys контроллера ОВЕН ПЛК.

На основе описания конфигурации ПЛК CoDeSys проверяет правильность

задания МЭК-адресов, используемых в программах, на их соответствие фактически имеющимся аппаратным средствам.

В процессе создания и отладки проекта необходимо настроить конфигурацию входов, выходов и интерфейсов связи ПЛК с внешними устройствами.

Настройка конфигурации выполняется в окне редактора «Конфигурация ПЛК (PLC Configuration)» ПО CoDeSys.

Для входа в режим редактирования конфигурации ПЛК следует перейти на вкладку «Ресурсы» Организатора объектов. В «Дереве ресурсов» следует выбрать пункт «Конфигурация ПЛК (PLC Configuration)». В рабочей области главного окна откроется окно редактора (рисунок 2.24).

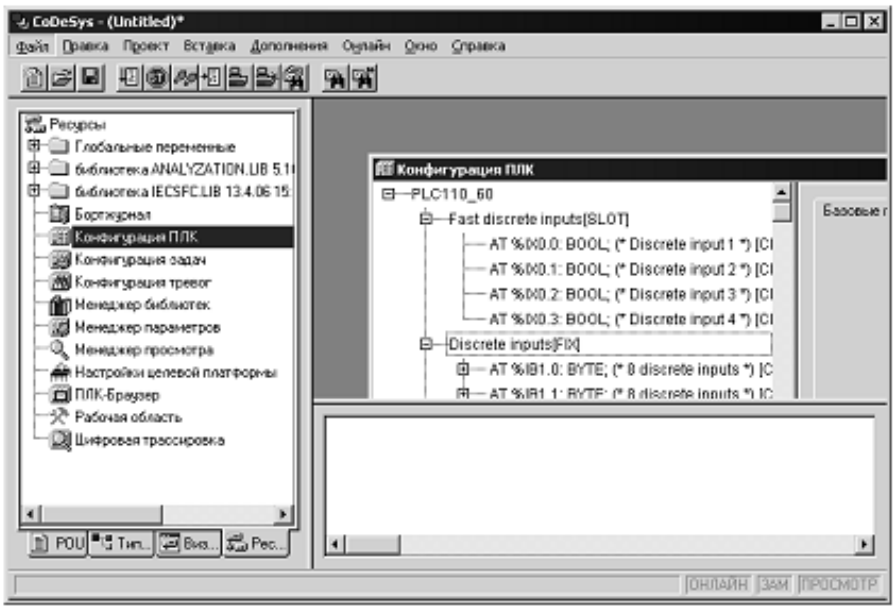


Рисунок 2.24 – Вход в режим «Конфигурация ПЛК (PLC Configuration)»

Окно редактора конфигуратора ПЛК разделено на две части. В левой части окна отображается дерево конфигурации, древовидная структура, отображающая ресурсы контроллера.

Структура и компоненты дерева определяются файлом настроек целевой платформы конфигурации, но могут быть изменены пользователем CoDeSys.

В дереве конфигурации отображаются следующие элементы.

1 Модуль (элемент конфигурации): независимая единица аппаратных средств. Модуль включает набор каналов ввода-вывода. Модуль (как и каждый отдельный канал) может иметь параметры. Каждый тип модуля имеет уникальный идентификатор.

2 Канал: это собственно данные ввода-вывода. Как правило, модуль имеет фиксированный набор каналов или подмодулей. Каждый канал имеет определенный МЭК-тип и адрес. Для каждого канала автоматически выделяется определенное пространство памяти. Каждый канал имеет уникальный в пределах данной конфигурации ПЛК идентификатор.

3 Битовый канал: идентификатор отдельного бита в канале.

В дереве конфигурации задается распределение адресов входов/выходов контроллера, что определяет привязку проекта к аппаратным средствам.

В правой части окна отображаются диалоги конфигурации, доступные для текущего (выделенного) элемента дерева конфигурации. Диалоги отображаются в виде одной или нескольких табличных вкладок (рисунок 2.25).

В полях, расположенных на вкладках диалогов, задаются требуемые значения параметров канала или модуля. Значение параметра устанавливается до компиляции проекта.

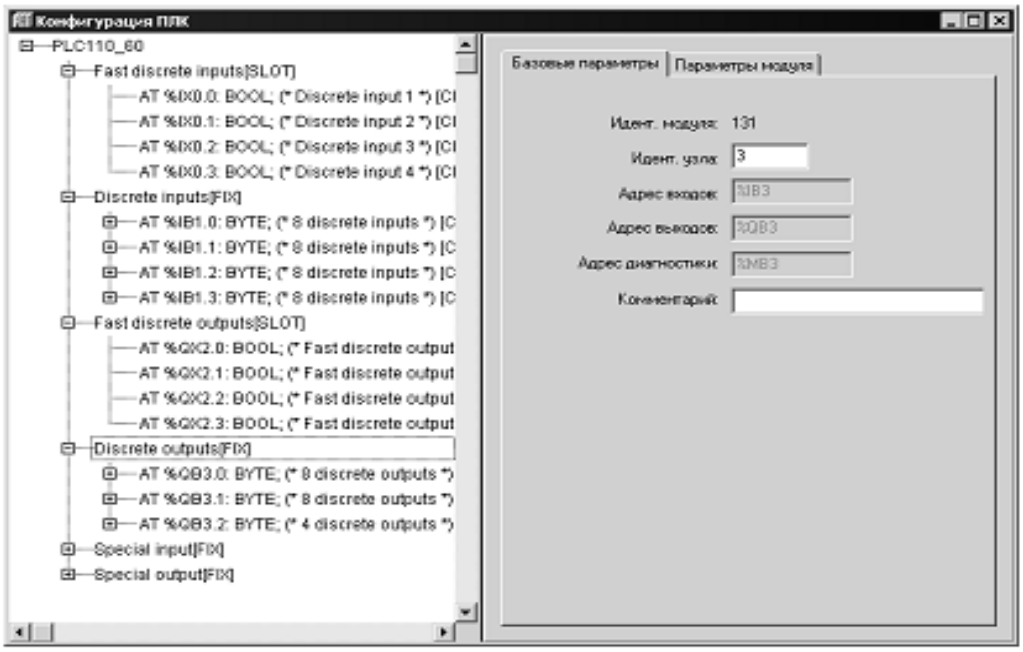


Рисунок 2.25 – Окно режима «Конфигурация ПЛК (PLC Configuration)»

2.1.9 Привязка переменных к входам и выходам ПЛК.

У ПЛК могут быть аналоговые и дискретные входы и выходы, информация о которых хранится в его target-файле. Для конфигурирования входов и выходов нужно войти во вкладку «Ресурсы», а в ней – в «Конфигурацию ПЛК». Входы и выходы ПЛК описаны в таблице 2.2.

Таблица 2.2 – Входы и выходы ПЛК

Английский	Русский (примеры)
Discrete input	Дискретный вход (сигналы от датчиков-реле)
Discrete output	Дискретный выход (сигналы включения нагревателя или исполнительного механизма постоянной скорости)
Unified signal sensor	Аналоговый вход (сигналы аналоговых датчиков, например, преобразователей давления)
Analog output	Аналоговый выход (сигнал к позиционеру)

Рассмотрим для примера конфигурирование аналогового входа, к которому подключен термометр сопротивления. Для настройки следует кликнуть

кнопкой мыши выбранный аналоговый вход (нумерация сверху вниз), выбрать в появившемся меню «Заменить элемент» и выбрать RTD sensor (рисунок 2.26).

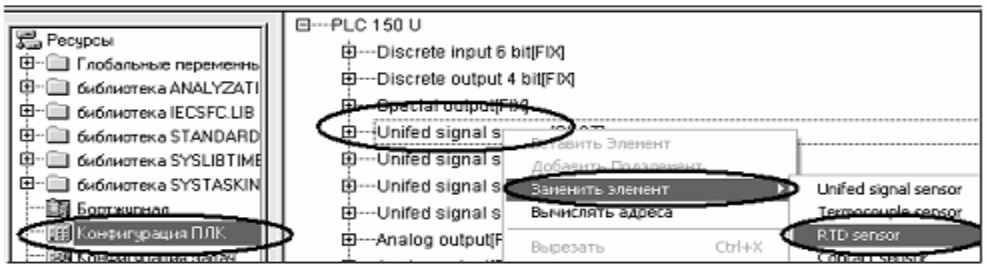


Рисунок 2.26 – Конфигурирование аналоговых входов. Выбор типа датчика

Далее следует кликнуть «AT» в первой строке настраиваемого входа и ввести имя переменной (PV, рисунок 2.27, *а*). Эта переменная будет глобальной. Затем кликнуть вид датчика (RTD sensor), открыть «Параметры модуля» (рисунок 2.27, *б*) и выбрать тип датчика (r428_50). Здесь можно вводить поправки в трех выбранных точках с линейной интерполяцией. Кроме того, указывается время цикла измерения переменной по настраиваемому входу.

а) *б*)

Рисунок 2.27 – Конфигурирование аналоговых входов. Продолжение

Рассмотрим добавление и конфигурирование широтно-импульсной модуляции (ШИМ). Для этого открываем дискретные выходы, открываем контекстное меню кликом кнопкой правой кнопки мыши и выбираем «Вставить Pulse-Width Modulator». Появится группа «Pulse-width modulator» под всеми дискретными входами. Далее открываем группу ШИМ и присваиваем имя переменной (на рисунке 2.28 – to_pwm).

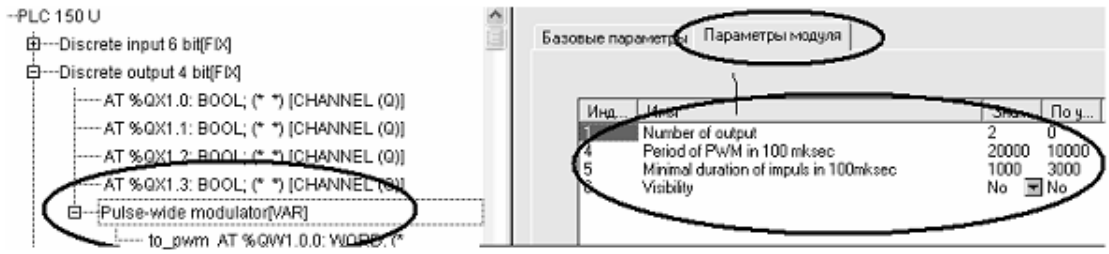


Рисунок 2.28 – Конфигурирование ШИМ

Затем входим во вкладку «Параметры модуля» и указываем в первой строке выход, к которому подключен ШИМ (следует иметь в виду, что нумерация в CoDeSys начинается с нуля), во второй строке – период ШИМ в сотнях микросекунд, в третьей – минимальную длительность импульса в сотнях микросекунд.

2.2 Порядок проведения лабораторной работы

1 Изучить состав системы программирования CoDeSys, назначение компонентов.

2 Изучить порядок установки системы и инсталляции target-файлов.

3 Изучить состав проекта, порядок создания нового проекта, создание и редактирования компонентов проекта.

4 Изучить способы и последовательность объявления переменных.

5 Изучить методы отладки проекта.

6 Изучить последовательность подключения контроллера к компьютеру и настройки связи.

7 Изучить порядок конфигурации модулей ПЛК.

8 Подготовить ответы на контрольные вопросы.

9 Составить отчет по работе.

При выполнении работы дополнительную информацию можно получить, из «Руководства пользователя по программированию ПЛК в CoDeSys 2.3 » или используя справочную систему среды программирования CoDeSys.

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Описание структуры проекта и основных правил его создания.

4 Описание компонентов POU, которые могут содержаться в проекте.

5 Описание операндов, используемых в системе программирования.

6 Описание классов переменных, используемых в компонентах проекта.

7 Таблица 2.2.

8 Фото окон создания проекта в среде CoDeSys для конкретного типа контроллера.

9 Вывод по лабораторной работе.

10 Ответы на контрольные вопросы.

Контрольные вопросы

1 Какие области содержит окно проекта в системе CoDeSys?

2 Какая информация отображается в статусной строке окна проекта в системе CoDeSys?

3 Какие файлы формируются при создании проекта и его компиляции?

4 Как выбрать целевую платформу и каким образом ее настроить?

5 Как выбрать язык программирования при создании проекта и можно ли его изменить в процессе работы над пректом?

6 Как подключить библиотеку к проекту?

7 Что такое экземпляр функционального блока и чем функция отличается от функционального блока?

8 Каким образом можно открыть проект и загрузить его в контроллер?

9 Какие методы отладки программ используются в системе CoDeSys?

10 Какие типы данных используются в языках программирования системы CoDeSys?

11 Каким способом объявляются переменные в языках программирования системы CoDeSys?

12 Какая информация содержится в target-файлах?

13 Как переменную связать с конкретным входом контроллера?

14 Какие параметры интерфейса RS-232 устанавливаются при настройке связи?

15 Какие модули могут входить в состав изучаемого контроллера?

3 Лабораторная работа № 3. Создание нового проекта в среде CoDeSys для программирования контроллеров

Цель работы: изучение методики создания проектов, настройки среды программирования.

Задание

Создание нового проекта в среде CoDeSys, обеспечивающего контроль оператором движения некоторого механизма:

– оператор должен периодически подтверждать правильность функционирования механизма; в противном случае, необходимо выдать предупреждение, а затем остановить работу;

– рабочий орган машины совершает циклическое движение по периметру прямоугольника.

3.1 Рекомендации к выполнению задания

Для выполнения задания необходимо обратиться к руководству фирмы ОВЕН «Первые шаги с CoDeSys. Smart Software Solutions» (на компьютере: раздел 6 Документация, 04 CoDeSys).

3.1.1 Этапы создания проекта.

Этап 1. Запуск CoDeSys.

Пуск -> Программы -> 3S Software -> CoDeSys V2.3 -> CoDeSys V2.3.

Этап 2. Создание нового проекта.

Создать новый проект командой

File -> New.

Настройка целевой платформы (Target Settings).

Проект является машинно-независимым, его можно опробовать в режиме эмуляции. Выбираем для определенности конкретный контроллер. На страничке диалогового окна **Configuration** устанавливаем CoDeSys SP for Windows NT Realtime и подтвердим ввод – OK.

Создание главной программы PLC_PRG POU.

Диалоговое окно определяет тип первого программного компонента (New POU).

Необходимо выбрать язык реализации (language of the POU) FBD и сохранить предложенные по умолчанию тип компонента – программа (Type of the POU Program) и имя – Name PLC_PRG.

PLC_PRG – это особый программный компонент (POU). В однозадачных проектах он циклически вызывается системой исполнения.

Объявление – Переключатель подтверждения.

В первой цепи графического FBD-редактора выделить строку вопросов «???» и ввести наименование первой переменной. Пусть это будет Observer (наблюдатель). Нажать на клавиатуре стрелку вправо. В появившемся диалоге определения переменной сохранить наименование (Name Observer) и логический тип (Type BOOL). Изменить класс переменной (Class) на глобальный (VAR_GLOBAL). Подтвердить определение – OK. Определение переменной Observer должно появиться в окне глобальных переменных проекта (Global_Variables):

```
VAR_GLOBAL
        Observer: BOOL;
END_VAR
```

Создание – Детектор переднего фронта.

Оператор должен подтверждать работу именно переключением клавиши, а не просто находиться с постоянно нажатой клавишей подтверждения. Чтобы разделить эти ситуации, необходимо определить моменты нажатия и отпускания, т. е. переходы значения логической переменной их нуля (FALSE) в единицу (TRUE) и наоборот.

Необходимо вернуться в окно редактора PLC_PRG и выделить позицию справа от переменной Observer. Появится маленький пунктирный прямоугольник. Щелкнуть по нему правой клавишей мыши. В контекстном меню ввода задать команду Box.

По умолчанию, вставляется элемент AND. Необходимо воспользоваться

ассистентом ввода: нажать клавишу F2. В диалоговом окне (слева) выбать категорию: «Стандартные функциональные блоки» (Standard Function Blocks). Из триггеров (trigger) стандартной библиотеки (standard.lib) выберете R_TRIG. R_TRIG формирует логическую единицу по переднему фронту на входе.

Необходимо задать имя для нового экземпляра функционального блока R_TRIG. Щелкнуть мышью над изображением триггера и ввести имя Trigl. В диалоге определения переменных должен быть указан класс Class VAR (локальные переменные), имя (Name) Trigl и тип (Type R_TRIG). Нажать «OK».

Создание – Детектор заднего фронта.

Выделить вход функционального блока Trigl и вставить (как было описано выше) элемент AND и переименовать его в OR (логическое ИЛИ). Выделить свободный вход OR функционального и вставить перед ним экземпляр функционального блока F_TRIG под именем Trig2. На вход F_TRIG с помощью ассистента ввода (F2) подать (категория Global Variables) переменную Observer.

Для выполнения последующих пунктов этапа **Создание нового проекта** необходимо обратиться к руководству фирмы ОВЕН «Первые шаги с CoDeSys. Smart Software Solutions» (на компьютере: раздел 6 Документация, 04 CoDeSys).

Контроль времени, первый интервал.

Выход – Предупреждение.

Формирование – Стоп Сигнал по второму интервалу времени.

Размещение POU управления механизмом.

Определение последовательности работы механизма.

Программирование первого шага.

Программирование следующих шагов.

Определение переходов.

Останов механизма.

Вызов POU управления механизмом.

Компиляция проекта.

Этап 3. Визуализация.

Перейти на страничку визуализации. В контекстном меню ввести команду добавления объекта Add object. Присвоить новому объекту имя Observation.

Для выполнения последующих пунктов этапа **Визуализация** необходимо обратиться к руководству фирмы ОВЕН «Первые шаги с CoDeSys. Smart Software Solutions» (на компьютере: раздел 6 Документация, 04 CoDeSys).

В конце работы окно визуализации будет выглядеть согласно рисунку 3.1.

Этап 4. Запуск целевой системы.

Запустите систему исполнения (обратите внимание, что CoDeSys SP RTE работает только в Windows NT 4.0, Windows 2000 или Windows XP). Теперь в панели задач вы увидите иконку CoDeSys SP RTE. Щелкните по ней правой клавишей и дайте команду на старт системы (Start System).

Этап 5. Настройка канала и соединение.

Если вы в первый раз подключаете контроллер к CoDeSys, необходимо выполнить определенные настройки.

В меню Online открыть диалог Communication parameters. Нажать клавишу «New» для настройки нового соединения. Желательно присвоить ему некоторое осмысленное имя.

Рисунок 3.1 – Визуализация.

В простейшем случае CoDeSys SP RTE работает на том же компьютере, что и среда программирования CoDeSys. Это означает, что можно применить способ соединения посредством разделяемой памяти (Shared memory (Kernel)).

Настройка подтверждается клавишей «OK».

Этап 6. **Запуск проекта**.

Соединение с контроллером устанавливается командой Online -> Login из среды программирования CoDeSys. Если используется удаленное соединение, CoDeSys попросит вас подтвердить загрузку (download) кода проекта.

Команда запускает Online -> Run проект. Перейти в окно визуализации и проверить работу механизма.

Для запуска проекта в режиме эмуляции установите флажок в меню Online -> Simulation. Далее перейти в режим online и запустить проект, как описано выше.

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Этапы создания проекта.

4 Описание структуры проекта и основных правил его создания.

5 Фото окна программы «Создание детектора переднего фронта».

6 Фото окна программы «Выход Предупреждение».

7 Фото окна программы «Формирование Стоп Сигнала».

8 Фото окна программы «Выход Предупреждение».

9 Фото окон программы «Всех шагов» согласно алгоритма.

10 Фото окна программы «Визуализация».

11 Вывод по лабораторной работе.
12 Ответы на контрольные вопросы.

Контрольные вопросы

1 Как обозначается главная программа проекта?
2 Как формируются переменные в новом проекте?
3 Сколько фаз работы механизма должно присутствовать в программе?
4 С чего начинается программирование первого шага?
5 Как программируются следующие шаги?
6 Как обеспечить останов механизма?
7 Какое условие перехода на следующий шаг?
8 Что такое компиляция проекта? Как осуществляется?
9 Как настроить первый элемент визуализации?
10 Какие модули могут входить в состав изучаемого контроллера?

4 Лабораторная работа № 4. Программирование логических контроллеров на языке LD в системе CoDeSys

Цель работы: изучить принципы составления прикладных программ для промышленных логических контроллеров (ПЛК) на языке LD пакета CoDeSys. Приобретение навыков программирования на языке LD в системе CoDeSys.

Задание

1 Используя схему управления асинхронным двигателем, составить программу на языке LD, обеспечивающую управление по заданному алгоритму.
2 Отладить программу в режиме эмуляции.
3 Записать в контроллер и проверить ее работу.

4.1 Рекомендации к выполнению задания

Для выполнения задания необходимо обратиться к руководству фирмы ОВЕН (на компьютере: раздел 6 Документация, 04 CoDeSys) «Руководство пользователя по программированию ПЛК в CoDeSys 2.3». (UserManual_V23_RU.pdf) и оперативной справочной системе CoDeSys.

4.1.1 Общие сведения о языке лестничных диаграмм LD (*Ladder Diagram*).

Язык лестничных диаграмм LD (Ladder Diagram), или язык релейно-контактных схем (РКС), является достаточно популярным в силу своей наглядности и позволяет решать широкий круг задач комбинационной и событийно-управляемой логики.

Язык LD – графический язык, в котором программа выглядит, как релейная схема в стандарте промышленной автоматизации. Две вертикальные линии

слева и справа образуют линии питания. Между ними располагаются контактные цепи в виде горизонтальных линий по аналогии с обычными электрическими цепями релейной автоматики (по общему виду программы и дано название языка программирования). Слева по линии располагаются контакты (соответствуют входным переменным логического типа и дискретным входам). Справа – катушки реле (соответствуют выходным переменным логического типа и дискретным выходам).

Каждому контакту соответствует логическая переменная. Если переменная имеет значение ИСТИНА, то контакт считается замкнутым, если переменная имеет значение ЛОЖЬ, то контакт считается разомкнутым.

Каждой катушке также соответствует логическая переменная. Если контактная цепь от левой линии до катушки состоит из замкнутых контактов, то реле считается включенным и соответствующей переменной присваивается значение ИСТИНА, иначе реле выключается и соответствующей переменной присваивается значение ЛОЖЬ. Если катушка инверсная (обозначается символом (/)), тогда в соответствующую логическую переменную копируется инверсное значение.

Предполагается, что контакты можно соединять в любом порядке и последовательности, определяя логику работы цепи, а катушки – только параллельно, как в релейных схемах подобных устройств.

Цепь из двух последовательно соединенных контактов соответствует логической операции «И», а цепь из двух параллельно соединенных – соответствует логической операции «ИЛИ». Операции «НЕ» соответствует нормально замкнутый контакт, который размыкает цепь (фактически, дает логический нуль) при включении (подаче логической единицы) и наоборот.

Кроме последовательного и параллельного соединения нормально разомкнутых или замкнутых контактов и катушек, язык LD позволяет:

– включать фиксируемые Set- / Reset- катушки;

– осуществлять переходы по цепям;

– включать в цепи функциональные блоки;

– управлять работой блоков по входам EN;

– записывать комментарии.

Программа на языке LD обрабатывается циклически слева направо, сверху вниз.

Порядок ввода, редактирования и отладки управляющих программ на языке LD в системе CoDeSys подробно описан в Руководстве пользователя по программированию ПЛК в CoDeSys 2.3.

4.1.2 Программирование ПЛК на LD.

Используя схему управления асинхронным двигателем, составить программу на языке LD, обеспечивающую управление по заданному алгоритму. Отладить программу в режиме эмуляции. Записать в контроллер и проверить ее работу.

Для того чтобы запускать электродвигатель в прямом и обратном направлении, применяется реверсивная схема управления на магнитном пускателе

(рисунок 4.1).

Подобную схему управления электродвигателем можно реализовать на базе ПЛК. Для этого к дискретным входам контроллера необходимо подключить кнопки управления, к выходам – через промежуточные реле пускатели. Принципиальная схема подключения элементов к условному контроллеру представлена на рисунке 4.2.

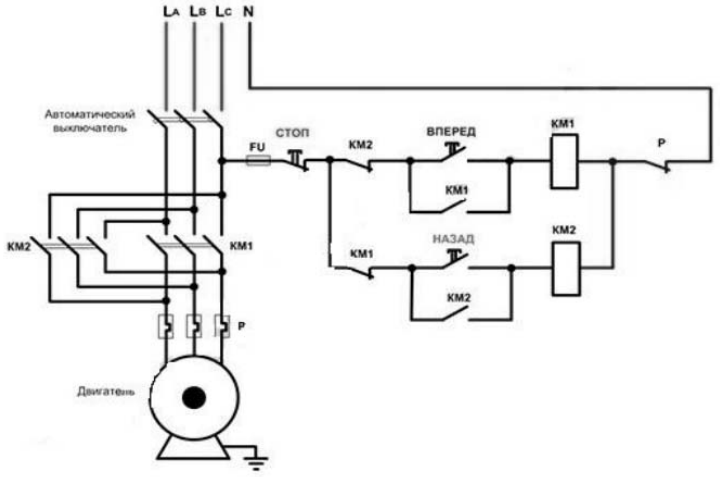


Рисунок 4.1 – Принципиальная схема работы реверсивного пускателя

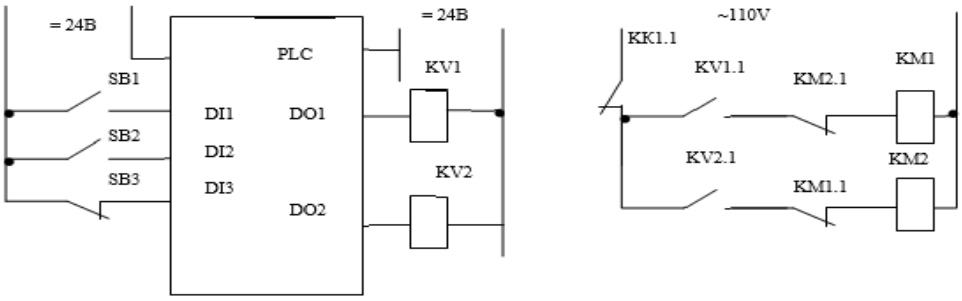


Рисунок 4.2 – Принципиальная схема подключения ПЛК

Входные и выходные сигналы ПЛК и соответствующие им переменные приведены в таблице 4.1.

Управляющая программа на языке LD, обеспечивающая запуск электрического двигателя в прямом и обратном направлении, т. е. функционально соответствующая схеме на рисунке 4.1, представлена на рисунке 4.3.

Таблица 4.1 – Таблица входных и выходных сигналов ПЛК

Номер входа	Подключаемый элемент	Тип сигнала	Символьное обозначение	Адрес
DI1	SB1	BOOL	VPERED	%IX0.0
DI2	SB2	BOOL	NAZAD	%IX0.1
DI3	SB3	BOOL	STOP	%IX0.2
DO1	KM1	BOOL	FRW	%QX1.0
DO2	KM2	BOOL	REV	%QX1.1

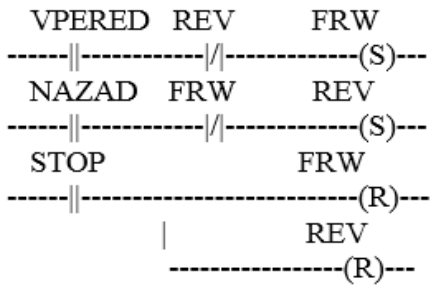


Рисунок 4.3 – Управляющая программа для ПЛК

4.2 Порядок проведения лабораторной работы

При выполнении работы необходимо дополнительно использовать Руководство пользователя по программированию ПЛК в CoDeSys 2.3.

1 Изучить основные правила составления управляющих программ на языке LD в системе CoDeSys.

2 Изучить порядок ввода редактирования и отладки управляющих программ на языке LD в системе CoDeSys.

3 Подготовить ответы на контрольные вопросы.

4 Изучить задание к лабораторной работе.

5 Разработать принципиальную схему управления двигателем на базе изучаемого программируемого логического контроллера.

6 Составить управляющую программу, реализующую управление двигателем в соответствии с заданием.

7 Проверить работу управляющей программы в режиме эмуляции.

8 Записать программу в память контроллера и проверить ее выполнение.

9 Составить отчет по работе.

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Список инструкций языка программирования LD.

4 Принципиальную схему управления двигателем на базе изучаемого программируемого контроллера.

5 Таблицу входных и выходных сигналов.

6 Фото листинга программы.

7 Вывод по лабораторной работе.

8 Ответы на контрольные вопросы.

Контрольные вопросы

1 Какие функциональные блоки могут вставляться в цепи языка LD системы CoDeSys?

2 Каким образом организуются переходы в управляющих программах, написанных на языке LD?

3 Как обозначается элемент, соответствующий дискретному входу или выходу контроллера в языке программирования LD?

4 Как вставить комментарии в управляющей программе на языке LD?

5 Какие позиции может занимать курсор при составлении программы на языке программирования LD?

6 Как изменить размеры элементов в редакторе LD?

7 Как изменить наименование элемента в редакторе LD?

8 Как изменить последовательность элементов в цепи программы на языке LD?

9 Как определить состояние элемента при отладке программы на языке LD?

10 Как выполняется программа на языке LD в пошаговом режиме?

5 Лабораторная работа № 5. Разработка управляющих программ с использованием типовых функциональных блоков в среде CoDeSys

Цель работы: освоить основные принципы разработки управляющих программ на языке LD с использованием типовых функциональных блоков: таймеров, триггеров, счетчиков, детекторов фронтов.

Задание

Используя схему управления асинхронным двигателем, разработанную при выполнении лабораторной работы № 4, составить программу на языке LD, обеспечивающую управление по заданному алгоритму.

Кратковременное нажатие первой кнопки запускает цикл: включение двигателя вперед на 2 с – останов – пауза 2 с – включение назад 3 с – останов – пауза 4 с – переход на начало цикла. Время работы 30 с, после этого останов. Кратковременное нажатие второй кнопки – останов в конце цикла и выдача прерывистого звукового сигнала по 1 с 3 раза. Нажатие третьей кнопки – останов немедленно и выдача звукового сигнала 3 с – отпускание кнопки – отключение звукового сигнала.

Цикл работы реализовать в виде функционального блока. Отладить программу в режиме эмуляции. Записать в контроллер и проверить ее работу.

5.1 Рекомендации к выполнению задания

Существует набор типовых функциональных блоков, реализующий функции, часто используемые в программировании ПЛК. Это счетчики, таймеры, триггеры, детекторы фронтов и т. д. В стандартной библиотеке подпрограмм Standart.lib, входящей в комплект CoDeSys, к таким блокам относятся:

– таймеры (TON, TOF, TP);

– счетчики (CTU, CTD, CTUD);

– триггеры (RS, SR);

– детекторы фронтов (R_TRIG, F_TRIG).

Эти функциональные блоки могут использоваться в программах на языке LD совместно с типовыми элементами этого языка. Информация о работе данных блоков содержится в Руководстве пользователя по программированию ПЛК в CoDeSys 2.3 и в Справочной системе комплекса программирования CoDeSys.

5.1.1 Графическое изображение типовых функциональных блоков: *таймеров*, *счетчиков*, *триггеров, детекторов фронтов и временные диаграммы, поясняющие их работу.*

1 Таймеры (TON, TOF, TP) – рисунки 5.1–5.3.

2 Счетчики (CTU, CTD, CTUD) – рисунок 5.4.

3 Триггеры (SR RS) – рисунок 5.5.

4 Детекторы фронтов (R_TRIG, F_TRIG) – рисунок 5.6.

Функциональный блок «таймер с задержкой включения»

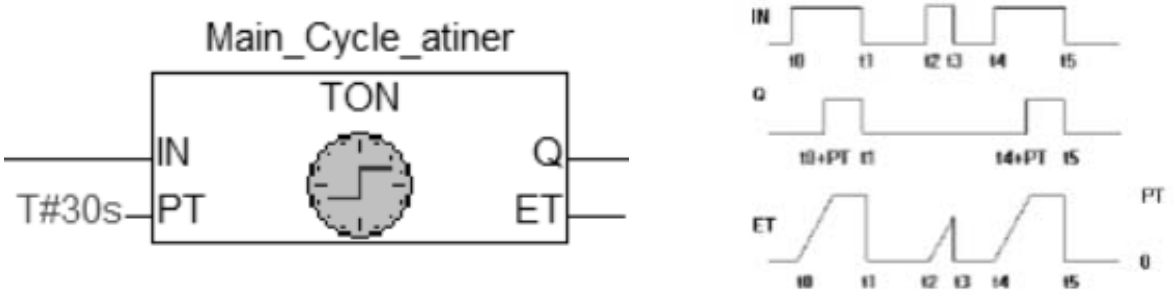


Рисунок 5.1 – Графическое изображение и временные диаграммы таймера TON

Функциональный блок «таймер с задержкой выключения»

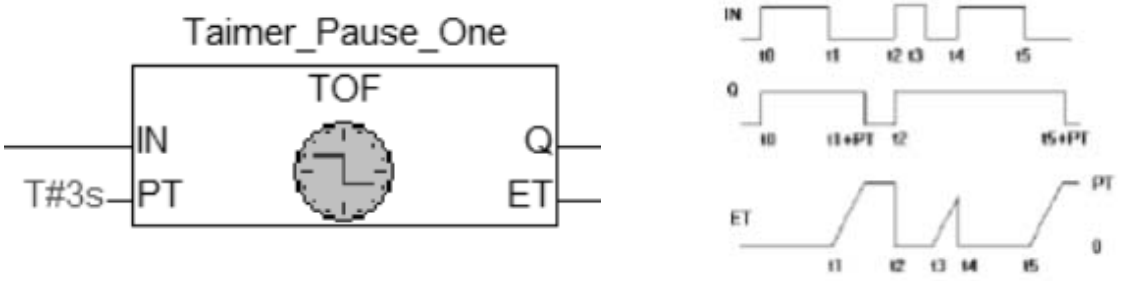


Рисунок 5.2 – Графическое изображение и временные диаграммы таймера TOF

Функциональный блок «таймер»

Рисунок 5.3 – Графическое изображение и временные диаграммы таймера TP

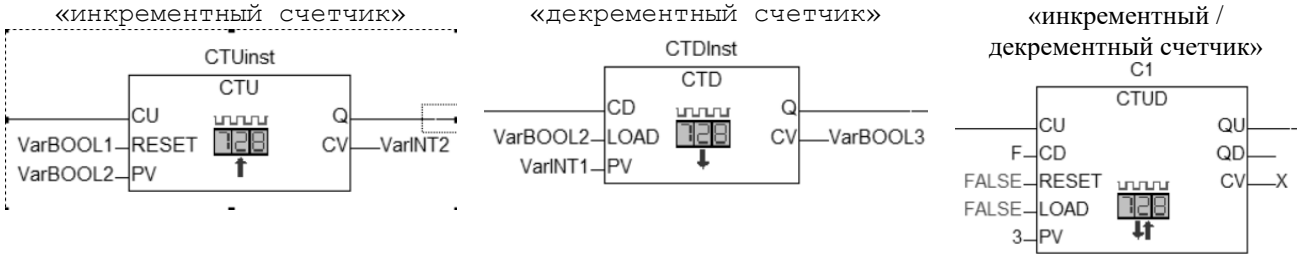


Рисунок 5.4 – Графическое изображение счетчиков CTU, CTD, CTUD

Рисунок 5.5 – Графическое изображение триггеров SR,

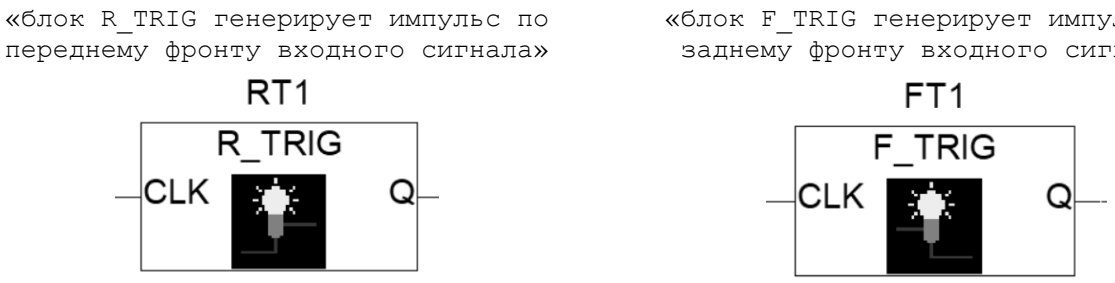


Рисунок 5.6 – Графическое изображение детекторов фронта R_TRIG, F_TRIG

5.2 Порядок проведения лабораторной работы

При выполнении работы необходимо дополнительно использовать Руководство пользователя по программированию ПЛК в CoDeSys 2.3 и Справочную систему комплекса программирования CoDeSys.

Требуется:

– изучить работу типовых блоков, реализующих функции таймеров, счетчиков, триггеров, детекторов фронтов и т. д., входящих в библиотеку Standart.lib системы CoDeSys;

– составить управляющую программу, реализующую управление двигателем по заданном в задании алгоритму;

– проверить работу управляющей программы в режиме эмуляции;

– записать программу в память контроллера и проверить ее выполнение в ПЛК.

5.3 Составление программы

Листинг программы представлен на рисунке 5.7.

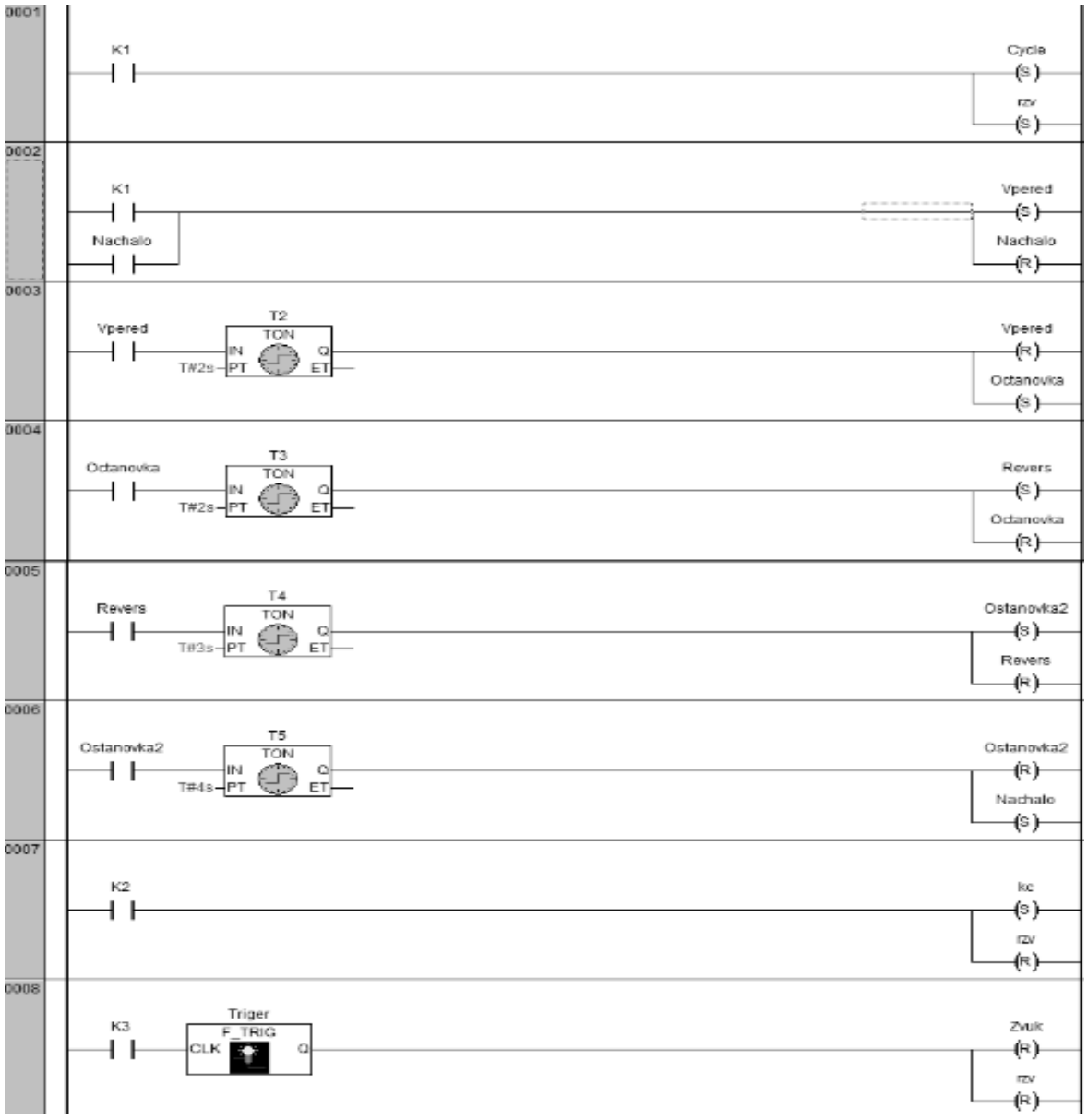
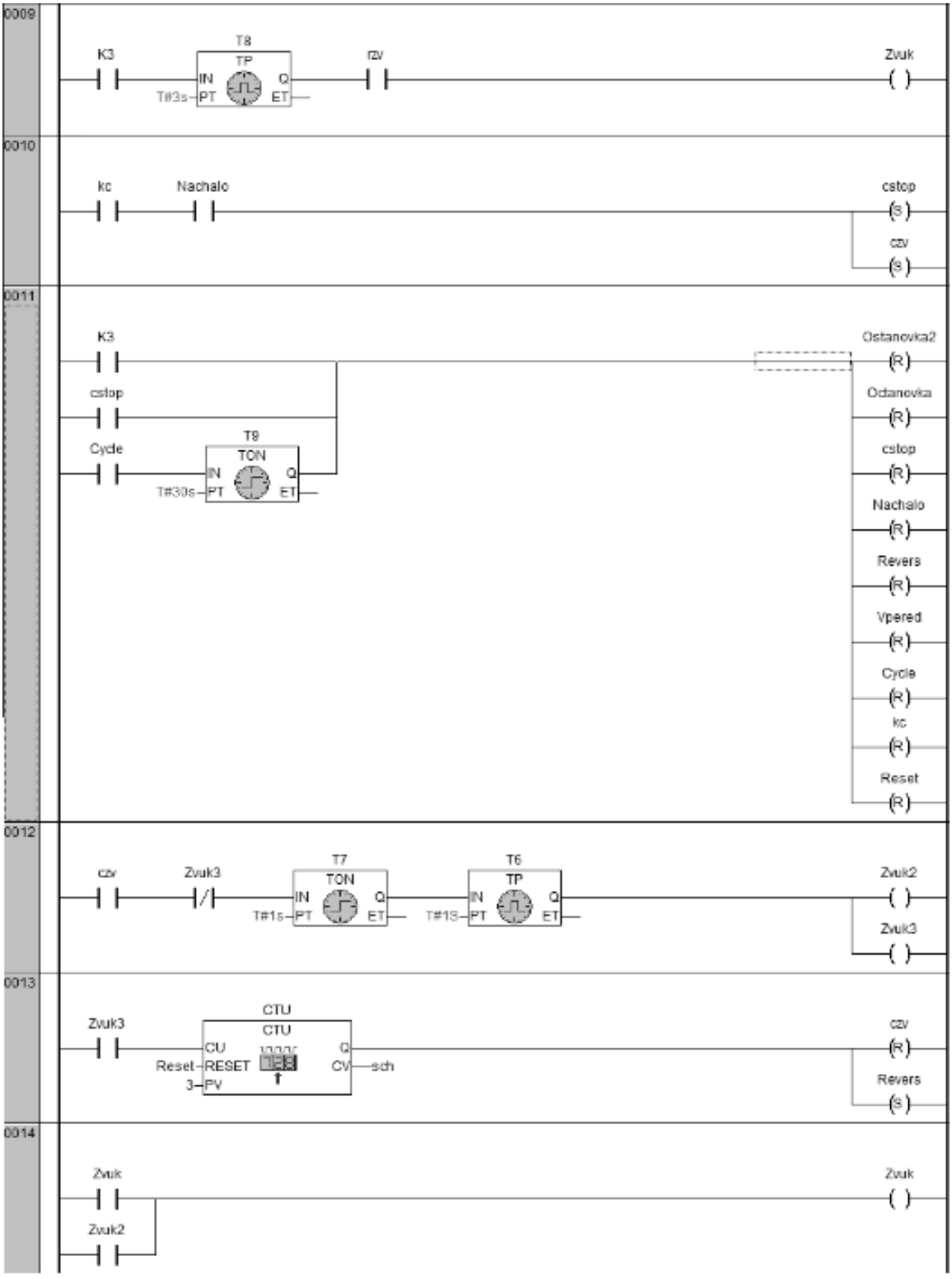


Рисунок 5.7 – Листинг программы на языке LD с применением функциональных блоков



Окончание рисунка 5.7

На рисунке 5.8 представлены обозначения входных и выходных сигналов.

	Имя	Адрес	Тип	Иниц.	Комментарий
0001	T1		TP		Реле времени 1
0002	Vpered		BOOL		Запуск двигателя вперед
0003	T2		TON		Реле времени(2 сек)
0004	Octanovka		BOOL		Остановка первая
0005	T3		TON		Реле времени(2 сек)
0006	Revers		BOOL		Реверс(Назад)
0007	Ostanovka2		BOOL		Остановка вторая
0008	T4		TON		Реле времени(3 сек)
0009	Nachalo		BOOL		Возвращение в начало цикла
0010	T5		TON		Реле времени(4 сек)
0011	T6		TP		Реле времени(1 сек)
0012	T7		TON		Реле времени(1 сек)
0013	kc		BOOL		
0014	Triger		F_TRIG		Детектор заднего фронта
0015	T8		TP		Реле времени(3 сек)
0016	rzv		BOOL		
0017	T9		TON		Реле времени(30 сек)
0018	Cycle		BOOL		
0019	cstop		BOOL		
0020	CTU		CTU		Функциональный блок 'инкрементный счетчик'
0021	Zvuk3		BOOL		Звуковой сигал
0022	Zvuk2		BOOL		Звуковой сигал
0023	sch		INT		
0024	czv		BOOL		Выдача прерывистого звукового сигнала три раза
0025	K1		BOOL		Первая кнопка
0026	K2		BOOL		Вторая кнопка
0027	K3		BOOL		Третья кнопка
0028	Reset		BOOL		Сброс
0029	Zvuk		BOOL		

Рисунок 5.8 – Обозначения входных и выходных сигналов разработанной программы

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Задание.

4 Графическое изображение типовых функциональных блоков: таймеров, счетчиков, триггеров, детекторов фронтов и временные диаграммы, поясняющие их работу.

5 Блок-схему алгоритма управления двигателем.

6 Список управляющих кнопок и исполнительных устройств с указанием адресов входов-выходов, к которым они подключены.

7 Листинг программы.

8 Фото реализации программы в динамике.

9 Вывод по лабораторной работе.

10 Ответы на контрольные вопросы.

Контрольные вопросы

1 Как функциональные блоки, имеющие несколько входов и выходов, вставляются в цепи языка LD системы CoDeSys?

2 В каких единицах и в каком диапазоне задаются интервалы времени?

3 Каким образом объявляются переменные, определяющие интервалы времени?

4 Чем отличается RS- триггер от SR- триггера?

5 Как реализовать таймер типа TOF, используя таймер типа TON?

6 Какая максимальная разрядность счетчика, реализованного в виде функционального блока CTU?

7 Какой длительности сигнал формируется детектором перехода R_TRIG и как ее можно изменить?

6 Лабораторная работа № 6. Разработка управляющей программы с использованием комбинаций языков программирования в среде CoDeSys

Цель работы: изучение методики создания управляющей программы с использованием языков программирования SFC, FBD, IL.

Задание

Создание блока управления движением на перекрестке, имеющем светофоры для двух пересекающихся направлений движения. Светофоры должны иметь два противоположных состояния – красный и зеленый. Чтобы избежать несчастных случаев, необходимо добавить общепринятые переходные стадии: желтый и желто-красный. Последняя стадия должна быть длиннее предыдущей.

6.1 Рекомендации к выполнению задания

Для выполнения задания необходимо обратиться к руководству фирмы ОВЕН (на компьютере: раздел 6 Документация, 04 CoDeSys) «Руководство пользователя по программированию ПЛК в CoDeSys 2.3» (UserManual_V23_RU.pdf) , глава 3.

6.1.1 Состав управляющей программы.

Управляемые по времени процессы можно представить средствами языков стандарта МЭК 61131-3, которые легко можно комбинировать в среде CoDeSys.

Управляющая программы состоит из трех блоков:

1) блок на языке Sequential Function Chart (SFC) с именем SEQUENCE;

2) функциональный блок на языке Function Block Diagram (FBD) с именем TRAFFICSIGNAL;

3) блок WAIT на языке Список Инструкции (IL).

6.1.2 Создание управляющей программы блока управления движением на перекрестке (*управления светофором*).

1 Создать компонент проекта POU.

Для этого необходимо запустить CoDeSys и выбрать «Файл» – «Создать» (File – New).

В окне диалога определить первый POU. По умолчанию он получает наименование PLC_PRG. Тип этого POU, безусловно, должен быть – программа. Каждый проект должен иметь программу с таким именем. В качестве языка программирования данного POU выбрать язык Continuous Function Chart (CFC).

2 Необходимо создать еще три объекта. Воспользоваться командой «Проект» – «Объект – Добавить» (Project – Object Add) в системном или в контекстном (нажать правую кнопку мыши в Организаторе объектов) меню.

Создать: программу на языке Sequential Function Chart (SFC) с именем SEQUENCE, функциональный блок на языке Function Block Diagram (FBD) с именем TRAFFICSIGNAL и еще один аналогичный блок – WAIT, который будет описан на языке Список Инструкции (IL).

Блок TRAFFICSIGNAL предназначен для сопоставления определенных стадий процесса соответствующим цветам. То есть необходимо удостоверится, что красный свет зажжен в красной стадии и в желто-красной стадии, желтый свет в желтой и желто-красной стадии и т. д.

Блок WAIT предназначен для создания таймера, который на вход получает длину стадии в миллисекундах и на выходе выдает состояние ИСТИНА по истечении заданного периода времени.

Блок SEQUENCE предназначен для объединения так, чтобы нужные огни зажигались в правильное время и на нужный период времени.

В **PLC_PRG** вводится входной сигнал включения, разрешающий начало работы светофора и «цветовые команды» каждой лампы связаны с соответствующими выходами аппаратуры.

3 Объявления блока TRAFFICSIGNAL.

В редакторе объявлений определить входную переменную (между ключевыми словами VAR_INPUT и END_VAR) по имени STATUS типа INT. STATUS будет иметь четыре возможных состояния, определяющие соответствующие стадии – зеленая, желтая, желто-красная и красная. Поскольку блок TRAFFICSIGNAL имеет три выхода, нужно определить еще три переменных RED, YELLOW и GREEN.

На рисунке 6.1 представлен функциональный блок TRAFFICSIGNAL, раздел объявлений.

4 Программирование блока TRAFFICSIGNAL.

Необходимо описать, что связывает вход STATUS с выходными переменными. Для этого перейти в раздел кода POU (body). Щелкнуть на поле слева от

первой цепи (серая область с номером 1). Выбрана первая цепь. Дадим команду меню «Вставка» – «Элемент» (Insert – Box).

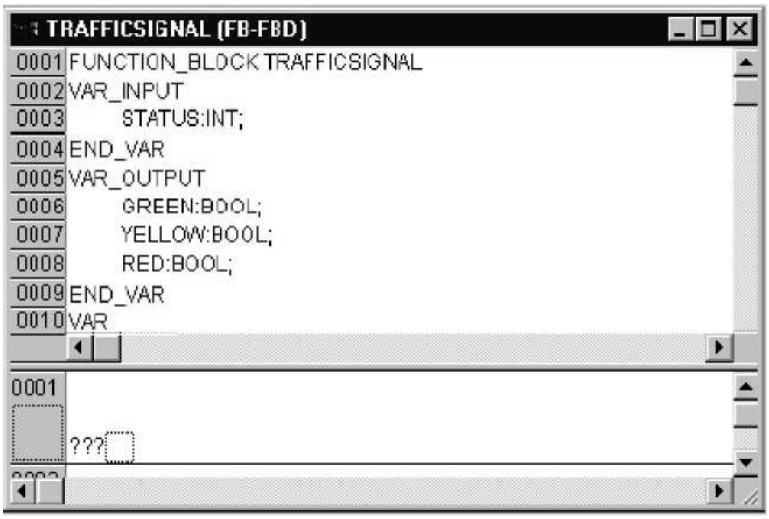
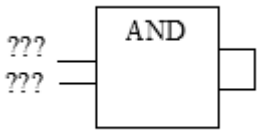


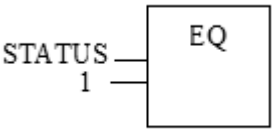
Рисунок 6.1 – Функциональный блок TRAFFICSIGNAL, раздел объявлений

В первой цепи будет вставлен прямоугольник с оператором AND и двумя входами.



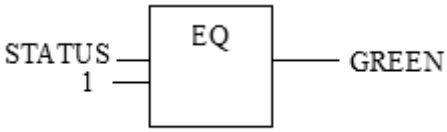
Щелкнуть мышкой на тексте AND и замените его на EQ.

Три знака вопроса около верхнего из двух входов заменить на имя переменной STATUS. Для нижнего входа вместо трех знаков вопроса нужно поставить 1. В результате получен следующий элемент.



Щелкнуть теперь на месте позади прямоугольника EQ. Теперь выбран выход EQ. Выполните команду меню «Вставка» – «Присваивание» (Insert – Assign).

Изменить три вопроса ??? на GREEN. Создана цепь следующего вида.



STATUS сравнивается с 1, результат присваивается GREEN. Таким образом, GREEN будет включен, когда STATUS равен 1.

Для других цветов TRAFFICSIGNAL понадобятся еще две цепи. Создать их командой «Вставка» – «Цепь (после)» (Insert – Network (after)). Законченный POU должен выглядеть как на рисунке 6.2.

Чтобы вставить оператор перед входом другого оператора, необходимо выделить сам вход, а не текст (выделяется прямоугольником). Далее используйте команду «Вставка» – «Элемент» (Insert – Box).

Теперь первый POU закончен. Как и планировалось, TRAFFICSIGNAL будет управлять включением выходов, руководствуясь значением переменной STATUS.

5 Подключение standard.lib.

Для создания таймера в POU WAIT понадобится POU из стандартной библиотеки. Необходимо открыть менеджер библиотек командами «Окно» – «Менеджер библиотек» (Window – Library Manager). Выбрать «Вставка» – «Добавить библиотеку» (Insert – Additional library). Должно открыться диалоговое окно выбора файлов. Выбрать standard.lib из списка библиотек.

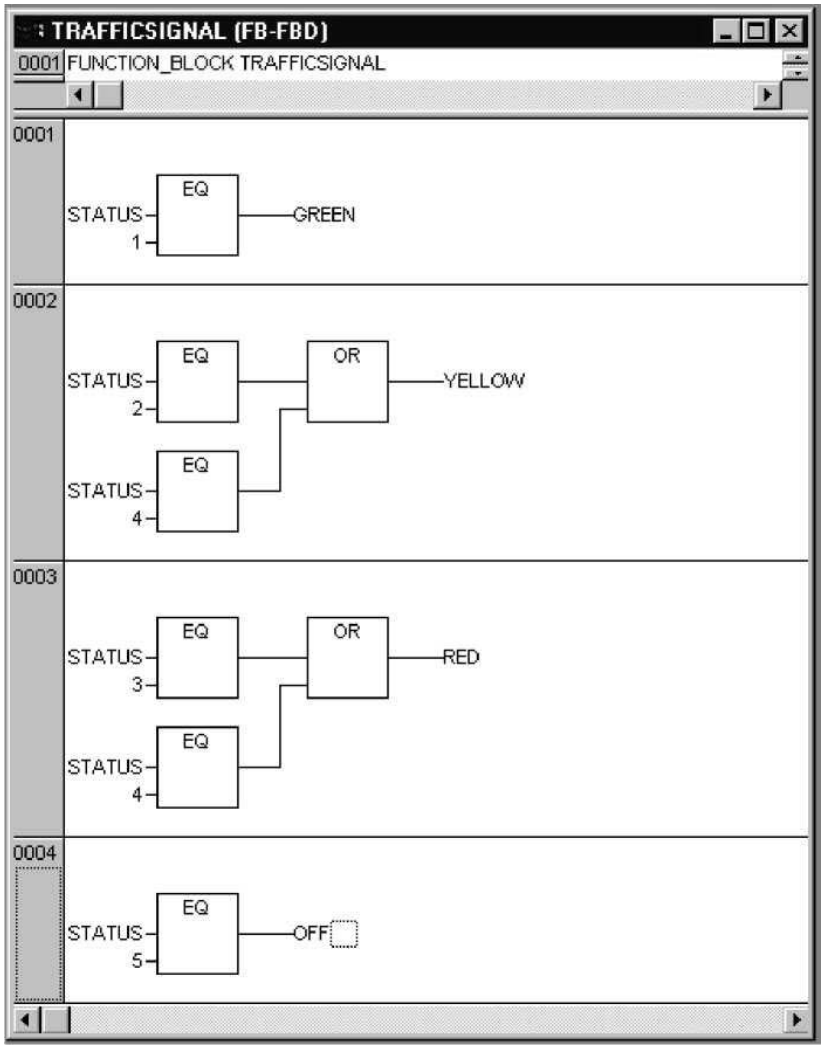


Рисунок 6.2 – Функциональный блок TRAFFICSIGNAL

6 Программирование блока TRAFFICSIGNAL.

7 Программирование блока WAIT.

8 Программирование блока SEQUENCE.

9 Визуализация.

С помощью визуализации можно быстро и легко увидеть работу блока управления светофором в динамике. На рисунке 6.3 представлены два светофора и их выключатель, который позволит включать и выключать блок управления светофором.

Для выполнения пп. 6–9 необходимо обратиться к руководству фирмы ОВЕН (на компьютере: раздел 6 Документация, 04 CoDeSys) «Руководство пользователя по программированию ПЛК в CoDeSys 2.3» (UserManual_V23_RU.pdf) , глава 3.

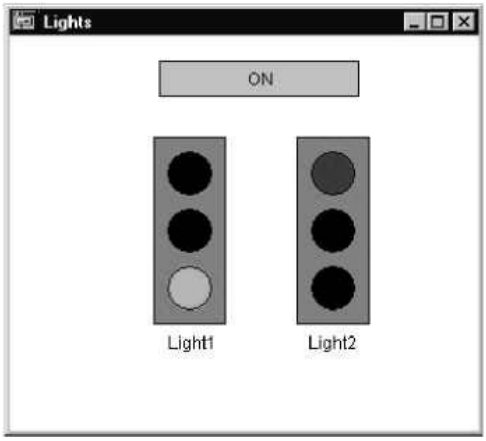


Рисунок 6.3 – Визуализация блока управления светофором

Содержание отчета

Отчет по лабораторной работе должен содержать следующее.

1 Титульный лист установленного образца.

2 Цель работы.

3 Задание.

4 Листинг программы создания блока TRAFFICSIGNAL.

5 Листинг программы создания блока WAIT.

6 Листинг программы создания блока SEQUENCE.

7 Фото реализации работы программы блока управления светофором в динамике (визуализация).

8 Вывод по лабораторной работе.

9 Ответы на контрольные вопросы.

Контрольные вопросы

1 Какие особенности программирования на языке Sequential Function Chart (SFC) (язык последовательных функциональных блоков, назначение, фрагмент программы)?

2 Какие особенности программирования на языке Function Block Diagram (FBD) (язык функциональных блоковых диаграмм)?

3 Какие особенности программирования на языке Instruction List (IL) (язык инструкций, назначение, фрагмент программы)?

4 Какое назначение блока TRAFFICSIGNAL?

5 Какое назначение блока WAIT?

6 Какое назначение блока SEQUENCE?

7 Как создать визуализацию блока управления светофором?

Список литературы

1 **Шишов, О. В.** Программируемые контроллеры в системах промышленной автоматизации: учебник / О. В. Шишов. – М. : ИНФРА-М, 2023. – 365 с.

2 **Волков, М. А.** Управление техническими и технологическими системами : учеб. пособие / М. А. Волков, А. Ю. Постыляков, Д. В. Исаков. – М. ; Вологда : Инфра-Инженерия, 2022. – 252 с.

3 Информационные системы и цифровые технологии : учеб. пособие: в 2 ч. / В. В. Трофимов, М. И. Барабанова, В. И. Кияев [и др.] ; под общ. ред. проф. В. В. Трофимова и В. И. Кияева. – М. : ИНФРА-М, 2021. – Ч. 1. – 253 с.

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

5 Общие сведения о CoDeSys. – URL: http://www.3s-software.ru /publications (дата обращения: 05.04.2025).

6 Каталог продукции фирмы ОВЕН. – URL: http://www.owen.ru/catalog (дата обращения: 05.04.2025).