

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учреждение образования
«Витебский государственный технологический университет»

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ УПРАВЛЕНИЯ В ЭНЕРГЕТИКЕ

**Методические указания по выполнению лабораторных работ
для студентов специальности
7-07-0712-02 «Теплоэнергетика и теплотехника»**

Витебск
2026

УДК 681.5

Составители:

А. А. Кузнецов, К. Н. Ринейский, А. М. Самусев

Одобрено кафедрой «Автоматизация производственных процессов»
УО «ВГТУ», протокол № 10 от 21.05.2026.

Рекомендовано к изданию редакционно-издательским советом
УО «ВГТУ», протокол № 9 от 29.05.2026.

Автоматизированные системы управления в энергетике : методические указания по выполнению лабораторных работ/ сост. А. А. Кузнецов, К. Н. Ринейский, А. М. Самусев. – Витебск : УО «ВГТУ», 2026. – 66 с.

Методические указания по выполнению лабораторных работ являются руководством к выполнению лабораторных работ по дисциплине «Автоматизированные системы управления в энергетике» для студентов специальности 7-07-0712-02 «Теплоэнергетика и теплотехника», содержат перечень лабораторных работ, освещают теоретические вопросы подготовки к их выполнению, приводят примеры построения автоматизированных систем управления на основе использования программируемого логического контроллера, программного обеспечения и средств автоматизации.

УДК 681.5

© УО «ВГТУ», 2026

СОДЕРЖАНИЕ

Введение	4
Лабораторная работа 1 «Типы программируемых логических контроллеров. Знакомство со стандартом IEC 61131-3. Цикл ПЛК»	5
Лабораторная работа 2 «Знакомство со средой программирования CoDeSys v2.3. Создание нового проекта. Типы данных переменных. Горячие клавиши»	7
Лабораторная работа 3 «Методика подключения и загрузки проекта в ПЛК. Работа с глобальными переменными»	11
Лабораторная работа 4 «Знакомство с языком SFC. Стандартные операторы CoDeSys»	15
Лабораторная работа 5 «Детекторы фронтов. Триггеры памяти»	28
Лабораторная работа 6 «Таймеры. Счётчики импульсов в среде CoDeSys»	32
Лабораторная работа 7 «Протокол обмена данными ModBus. Методика создания запросов в CoDeSys. Работа ПЛК в режиме Master»	39
Лабораторная работа 8 «Модули ввода-вывода Mx110. Модуль дискретного вывода МУ110-224.16Р. Работа ПЛК в режиме Master»	41
Лабораторная работа 9 «Настройка ПЛК для работы с TPM202 и TPM210 (ПЛК в роли Master)»	45
Лабораторная работа 10 «Настройка ПЛК для работы с TPM202 и TPM210 (ПЛК в роли Slave)»	47
Лабораторная работа 11 «Изучение принципов работы панели оператора с контроллером»	49
Лабораторная работа 12 «Настройка OPC-сервера»	57
Лабораторная работа 13 «Настройка обмена данных SCADA-системы»	60
Лабораторная работа 14 «Настройка интерфейса SCADA-системы»	62
Литература	65

ВВЕДЕНИЕ

Основная цель лабораторных работ – это развитие инженерных навыков по разработке систем автоматизации производственных процессов на основе современных технических средств контроля, управления на основе блочно-модульной автоматики и по созданию прикладного программного обеспечения на основе промышленных языков программирования стандарта МЭК 61131-3.

Разработка программного обеспечения имеет большое значение, так как является одной из важных составляющих в этапах разработки систем управления и формирования комплексных знаний инженера в сфере автоматизации.

Лабораторная работа 1

«Типы программируемых логических контроллеров. Знакомство со стандартом IEC 61131-3. Цикл ПЛК»

Цель работы: Ознакомится с линейкой программируемых логических контроллеров фирмы «ОВЕН». Рассмотреть ряд языков стандарта МЭК 61131-3. Знакомство с циклом ПЛК: особенности, принцип действия, типичные ошибки.

Теоретическая часть

Типы ПЛК. Для классификации огромного разнообразия существующих в настоящее время контроллеров рассмотрим их существенные различия. Основным показателем ПЛК является количество каналов ввода-вывода. По этому признаку ПЛК делятся на следующие группы: нано-ПЛК (менее 16 каналов), микро-ПЛК (более 16, до 100 каналов), средние (более 100, до 500 каналов), большие (более 500 каналов). По расположению модулей ввода-вывода ПЛК бывают: моноблочные, модульные, распределенные.

По конструктивному исполнению и способу крепления контроллеры делятся на: панельные (для монтажа на панель или дверцу шкафа) для монтажа на DIN-рейку внутри шкафа; для крепления на стене, стоечные, бескорпусные.

По области применения контроллеры делятся на следующие типы: универсальные общепромышленные, для управления роботами, для управления позиционированием и перемещением, коммуникационные, ПИД-контроллеры, специализированные. По способу программирования контроллеры бывают: программируемые с лицевой панели контроллера, программируемые переносным программатором, программируемые с помощью дисплея, мыши и клавиатуры, программируемые с помощью персонального компьютера.

Стандарт IEC 61131-3 (МЭК 61131-3). IEC 61131-3 – раздел международного стандарта IEC 61131, описывающий языки программирования для программируемых логических контроллеров. IEC 61131-3 – первый независимый от производителя стандартизированный язык программирования для промышленной автоматизации. Стандарт МЭК 61131-3 устанавливает пять языков программирования: три графических и два текстовых стандарта для языков программирования ПЛК. К графическим языкам программирования относятся: релейно-контактные схемы (LD), функциональные блочные диаграммы (FBD), последовательностные функциональные диаграммы (SFC). К текстовым языкам программирования относятся: лист инструкций (IL) и структурированный текст (ST).

Цикл ПЛК. Все ПЛК работают по методу периодического опроса входных данных (сканирования). ПЛК опрашивает входы, выполняет пользовательскую программу и устанавливает необходимые значения выходов (рис. 1.1).

Рабочий цикл ПЛК состоит из нескольких фаз:

1. Начало цикла.
2. Чтение состояния входов.

3. Выполнение кода программы пользователя.
4. Запись состояния выходов.
5. Обслуживание аппаратных ресурсов ПЛК.
6. Монитор системы исполнения.
7. Контроль времени цикла.
8. Переход на начало цикла.

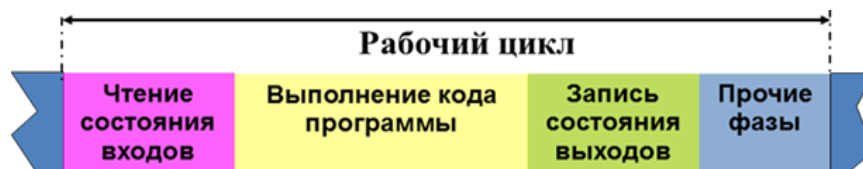


Рисунок 1.1 – Цикл ПЛК

Практическая часть

Ход работы. Ознакомление. В данной лабораторной работе необходимо следующее:

- 1) ознакомиться и рассмотреть основные типы ПЛК, критерии, по которым эти ПЛК разделяются;
- 2) привести примеры типов ПЛК в зависимости от его типа;
- 3) ознакомиться со стандартом МЭК 61131-3;
- 4) рассмотреть графические и текстовые языки программирования ПЛК;
- 5) ознакомиться с циклом ПЛК;
- 6) рассмотреть фазы, из которых состоит рабочий цикл ПЛК.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Описание типов ПЛК.
4. Примеры ПЛК в зависимости от их типа.
5. Стандарт МЭК 61131-3. Примеры языков данного стандарта.
6. Цикл ПЛК: описание, схема, последовательность.
7. Вывод.

Контрольные вопросы:

1. На какие типы подразделяются ПЛК?
2. По каким критериям ПЛК подразделяются на данные типы?
3. Что такое стандарт МЭК 61131-3?
4. На каких языках можно программировать ПЛК, исходя из стандарта?
5. Что такое цикл ПЛК?
6. Основные характеристики цикла ПЛК.

Лабораторная работа 2

«Знакомство со средой программирования CoDeSys v2.3. Создание нового проекта. Типы данных переменных. Горячие клавиши»

Цель работы: ознакомиться со средой программирования CoDeSys v2.3, создать новый проект в среде программирования, ознакомиться и изучить типы данных переменных.

Теоретическая часть

Знакомство со средой программирования CoDeSys. CoDeSys – это современный инструмент для программирования контроллеров (CoDeSys образуется от слов Controllers Development System). CoDeSys предоставляет программисту удобную среду для программирования контроллеров на языках стандарта МЭК 61131-3.

Создание нового проекта. Прежде всего нужно дать проекту новое имя, оно же послужит и названием файла проекта. Первый программный компонент (POU – Program Organization Unit) помещается в новый проект автоматически и получает название PLC_PRG.

Типы программного компонента. К программным компонентам в CoDeSys относятся: функция, функциональный блок и программа.

Функция (FUNCTION) – программный компонент, имеющий один или более входов, один выход, не имеющий внутренней памяти, который при каждом запуске работает аналогично и используется для комплексных вычислений.

Функциональный блок (FUNCTION_BLOCK) – программный компонент, имеющий произвольное число входов и выходов и внутреннюю память.

Программа (PROGRAM) – программный компонент, подобный функциональному блоку, но имеющий один глобальный экземпляр. **Важно помнить:** все используемые программные компоненты должны вызываться прямо или косвенно из главной программы PLC_PRG!

Описание методики создания запросов в CoDeSys. Для начала необходимо создать новый проект, затем идёт выбор и настройка целевой платформы, типа программного компонента и языка реализации.

При создании нового программного компонента, где типом POU является «Программа», переименовывать данный тип не рекомендуется (имя остаётся по умолчанию «PLC_PRG»).

Для корректной работы проекта необходимо увеличить минимальную длину опроса в 10 раз, перейдя во вкладку «Ресурсы → Конфигурация ПЛК → PLC110_30 → Параметры модуля»

Типы данных переменных. Существует два вида переменных: глобальные и локальные.

Глобальные переменные – это переменные, связанные с входами и выходами контроллера, которые объявляются во вкладке «ресурсы» → *Конфигурация ПЛК* → *PLC110_30*».

Локальные переменные (внутренние) объявляются при помощи сочетания клавиш *Shift+F2*, или вручную в окне объявления переменных (верхнее окно).

Важно помнить: *переменная объявляется только один раз, либо как локальная, либо как глобальная и обладает именем и типом!*

При наименовании переменной следует знать:

- переменная должна содержать буквы и цифры английского алфавита; переменная начинается только с буквы, где регистр не учитывается;
- переменная не должна содержать пробелов, но возможно использование одинарного подчеркивания;
- в имени переменной нельзя использовать зарезервированные слова (AND, PROGRAM, VAR и т. д.);
- нельзя использовать название и выходы функциональных блоков (RS, R, S, TP, CTU и т. д.);
- не должна содержать символы в имени (‘, <, ., > и т. д.).

Типы данных переменных. Для присвоения переменных используются следующие типы данных: логический (BOOL) – дискретные значения (только true или false), вещественный (INT) – целые, отрицательные или положительные числа (-32768...32767), вещественный неотрицательный (WORD) – целые, только положительные числа (0...65535), с плавающей точкой (REAL) – целые и дробные числа в широком диапазоне ($-1,2 \cdot 10^{-38} \dots 3,4 \cdot 10^{38}$), строковый (STRING) – произвольная последовательность символов, временной (TIME) – временные значения (пример T#5h10m35s15ms).

Горячие клавиши. Горячие клавиши предназначены для замены некоторых функций, что упрощает работу с проектом. Ниже представлен список горячих клавиш:

- **F1** – вызов справки. Для того чтобы вызвать справку для конкретного элемента, необходимо выделить его название и нажать F1;
- **F2** – ассистент ввода. Чтобы вызвать ассистент ввода, необходимо выделить знаки «???» и нажать F2. Помогает быстро и безошибочно присвоить ранее объявленную переменную;
- **Shift + F2** – объявление локальной переменной;
- **F5** – запуск программы;
- **F11** – компиляция. Проверка на ошибки;
- **F9** – точка останова;
- **Ctrl + F7** – записать значение.

Практическая часть

Ход работы. Создание проекта в среде CoDeSys. Пользователю необходимо создать проект в среде CoDeSys, выбрать тип программного компонента (программа) и язык реализации (CFC), добавить необходимые переменные, загрузить проект в ПЛК либо же симулировать его работу, проверить проект на работоспособность.

На примере простой задачи реализуем создание проекта. Допустим, у пользователя есть кнопка и лампочка. При нажатии на кнопку загорается лампочка (рис. 2.1).

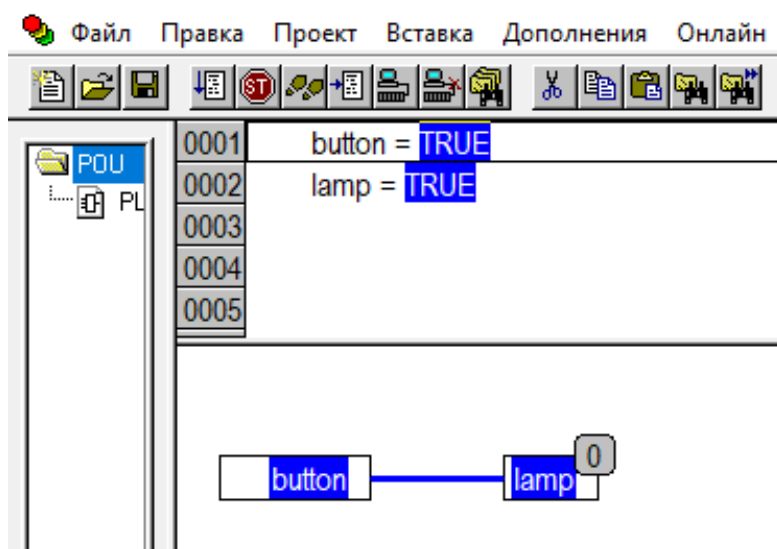


Рисунок 2.1 – Работа лампочки при нажатии на кнопку

Для проверки проекта на ошибки необходимо компилировать проект: проследовать по пути «Проект → Компилировать» либо воспользоваться горячей клавишей «F11». Для эмуляции работы контроллера необходимо перейти во вкладку «Онлайн → Режим эмуляции» и поставить галочку (рис. 2.2). После этого пользователь работает в режиме эмуляции. Затем, чтобы загрузить проект в контроллер необходимо перейти во вкладку «Онлайн → Подключение» (рис. 2.2), после чего проект загрузится в виртуальный контроллер. Далее для запуска проекта необходимо перейти во вкладку «Онлайн → Старт», затем пользователь сможет работать с этим проектом (рис. 2.2).

При старте программы переменные имеют значение «FALSE». Для изменения состояния переменной необходимо нажать на неё два раза левой клавишей мыши. Чтобы присвоить выбранное значение необходимо нажать сочетание горячих клавиш «Ctrl+F7». Для отключения от проекта необходимо перейти во вкладку «Онлайн → Отключение».

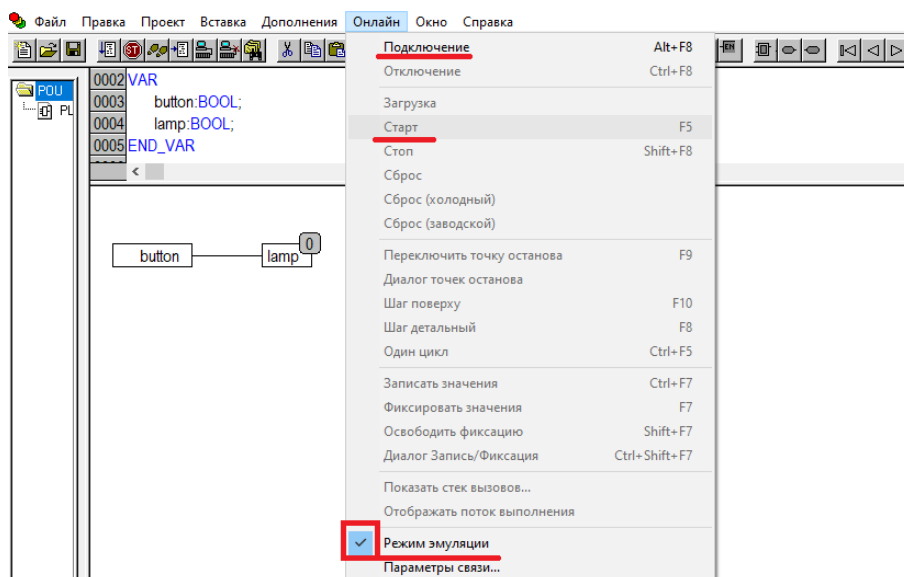


Рисунок 2.2 – Подключение проекта

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Знакомство со средой программирования CoDeSys v2.3.
4. Создание нового проекта.
5. Типы программного компонента.
6. Методика создания запросов в CoDeSys v2.3.
7. Типы данных переменных. Горячие клавиши.
8. Реализовать методику создания проекта в среде CoDeSys v2.3.
9. Вывод.

Контрольные вопросы:

1. Что такое «CoDeSys»?
2. Что такое «POU»?
3. Что такое «Программа»?
4. Что такое «Функциональный блок»?
5. Какие переменные называются «Глобальными»?
6. Какие переменные называются «Локальными»?
7. Перечислите типы данных переменных.
8. Какие горячие клавиши используются для записи значения?
9. Какие горячие клавиши используются для вызова «Ассистента ввода»?

Лабораторная работа 3

«Методика подключения и загрузки проекта в ПЛК. Работа с глобальными переменными»

Цель работы: ознакомиться и изучить методику подключения и загрузки проекта в ПЛК, работу с глобальными переменными.

Теоретическая часть

Для начала необходимо создать новый проект, затем идёт выбор и настройка целевой платформы, типа программного компонента и языка реализации. **Важно!** При создании нового программного компонента, где типом ROU является «Программа», переименовывать данный тип не рекомендуется (имя остаётся по умолчанию «PLC_PRG»).

Для корректной работы проекта необходимо увеличить минимальную длину опроса в 10 раз, перейдя во вкладку «Ресурсы → Конфигурация ПЛК → PLC110_30 → Параметры модуля» (рис. 3.1).

Добавление глобальных переменных (работа со входами и выходами ПЛК). Для работы с входами и выходами ПЛК пользователю необходимо перейти во вкладку «Ресурсы → Конфигурация ПЛК». При работе с ПЛК110-30 пользователю доступны следующие входы и выходы: 18 дискретных входов, 2 из которых быстрые; 12 дискретных выходов, 4 из которых быстрые. Для добавления необходимой переменной (входной или выходной) пользователю необходимо нажать «+» напротив соответствующего параметра. **Важно помнить!** При добавлении локальных и глобальных переменных имена у этих переменных не должны совпадать. При совпадении имени у двух переменных с разным нахождением, приоритет будет у локальной переменной, исходя из цикла ПЛК.

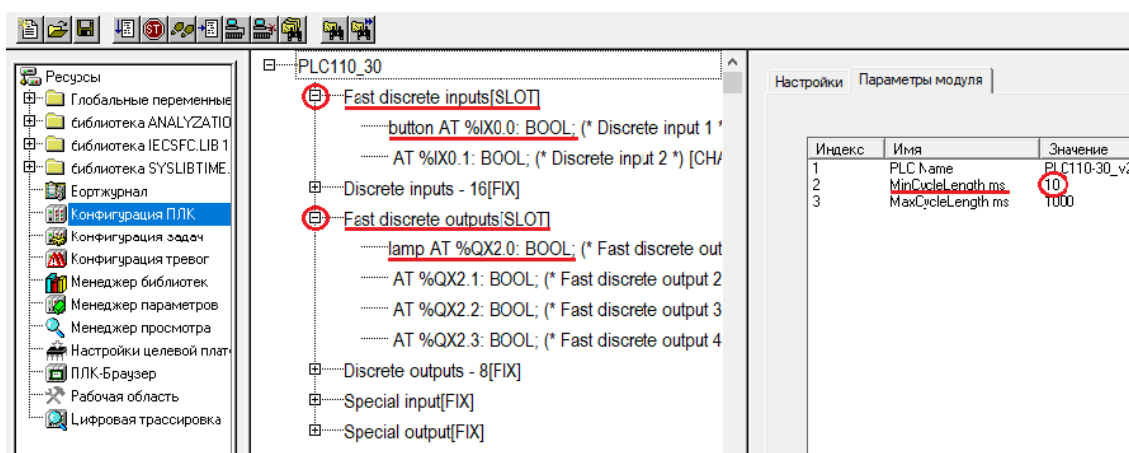


Рисунок 3.1 – Конфигурирование ПЛК. Добавление глобальных переменных

Добавление глобальных переменных (работа со входами и выходами ПЛК). Для работы с входами и выходами ПЛК пользователю необходимо перейти во вкладку «Ресурсы → Конфигурация ПЛК». При работе с ПЛК110-30 пользователю доступны следующие входы и выходы: 18 дискретных входов, 2 из которых быстрые; 12 дискретных выходов, 4 из которых быстрые. Для добавления необходимой переменной (входной или выходной) пользователю необходимо нажать «+» напротив соответствующего параметра (рис. 3.1). **Важно помнить!** При добавлении локальных и глобальных переменных имена у этих переменных не должны совпадать. При совпадении имени у двух переменных с разным нахождением, приоритет будет у локальной переменной, исходя из цикла ПЛК.

Настройка связи между CoDeSys и ПЛК. Для настройки связи между CoDeSys и ПЛК необходимо перейти во вкладку «Онлайн → Параметры связи». Перед пользователем появится окно настройки коммуникационного параметра (рисунок 3.2), посредством которого будет производится подключение к устройству. ПЛК включен в общую сеть Ethernet, поэтому настройку подключения пользователь будет настраивать по данному коммуникационному параметру.

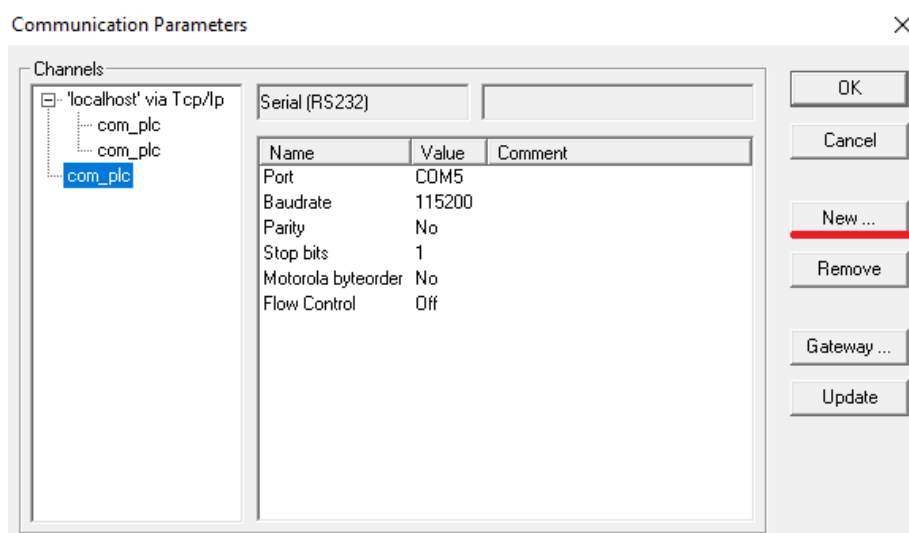


Рисунок 3.2 – Настройка подключения к ПЛК

Для добавления параметра необходимо нажать на кнопку «New» (рис. 3.2), перед пользователем откроется окно конфигурации нового параметра подключения (рис. 3.3). В данном окне необходимо обозвать создаваемый компонент (например, *Ethernet*) и выбрать устройство (*Tcp/Ip (Level2)*). Для подтверждения параметров пользователю необходимо нажать кнопку «Ок» (рис. 3.3).

Затем созданный параметр появится в дереве коммуникационных устройств (рис. 3.4). После необходимо настроить ранее созданный параметр, а именно в нашем случае прописать Ip-адрес устройства: 192.168.70.117. После завершения конфигурирования следует нажать клавишу «ОК». Созданный параметр будет выбран как способ подключения к устройству.

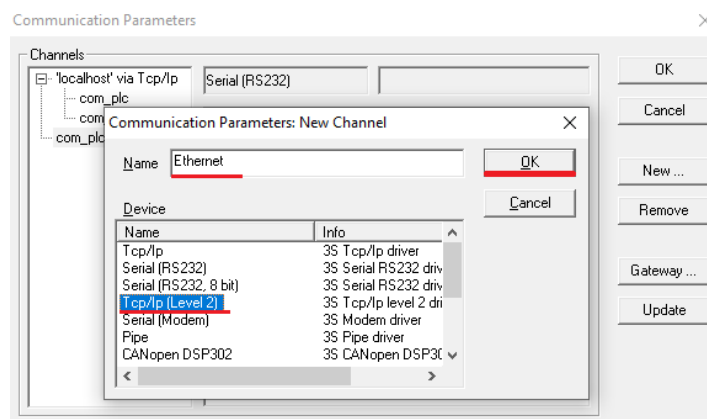


Рисунок 3.3 – Добавление нового коммуникационного параметра

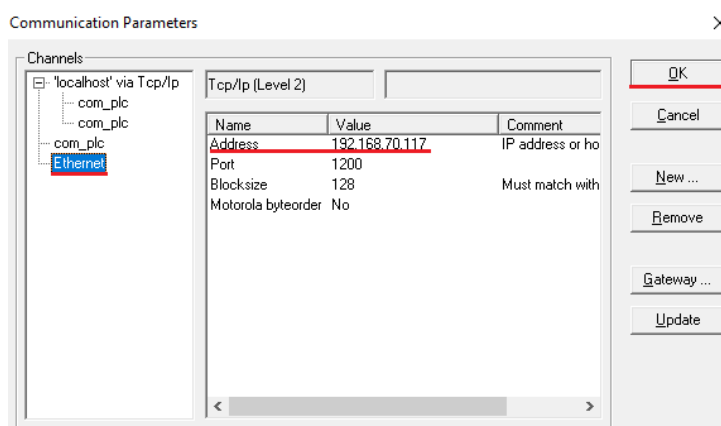


Рисунок 3.4 – Конфигурирование коммуникационного устройства

Практическая часть

Ход работы. Загрузка проекта в ПЛК. Для подключения и загрузки проекта в ПЛК необходимо перейти во вкладку «Онлайн → Подключение», либо воспользоваться сочетанием горячих клавиш «Alt+F8». **Важно!** При загрузке проекта в ПЛК необходимо убрать галочку перед «Режимом эмуляции», если этого не сделать, то программа будет эмулировать работу ПЛК, а не будет загружен в реальный; **подключить устройство к сети питания!** После подключения проекта необходимо перейти во вкладку «Онлайн → Старт», либо воспользоваться горячей клавишей «F5». После завершения работы с проектом необходимо отключить его от ПЛК путем перехода во вкладку «Онлайн → Отключение», либо воспользоваться сочетанием горячих клавиш «Ctrl+F8».

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Описание методики подключения и загрузки проекта в ПЛК.
4. Работа с глобальными переменными.

5. Настройка связи между CoDeSys и ПЛК.
6. Загрузка проекта в ПЛК.
7. Вывод.

Контрольные вопросы:

1. Что такое «CoDeSys»?
2. Где расположены глобальные переменные?
3. Какой тип данных имеют встроенные глобальные переменные?
4. Какие горячие клавиши используются для подключения и загрузки проекта?
5. Какие горячие клавиши используются для старта программы?
6. Какие горячие клавиши используются для отключения от проекта?

Лабораторная работа 4

«Знакомство с языком CFC. Стандартные операторы CoDeSys»

Цель работы: ознакомиться с графическим языком программирования CFC, изучить основные характеристики и принципы работы операторов CoDeSys: логики, арифметики, операторы выбора и ограничения, сравнения.

Теоретическая часть

Графический язык программирования CFC. Язык CFC (Continuous Flow Chart) – ещё один высокоуровневый язык визуального программирования. По сути, CFC – это дальнейшее развитие языка FBD. Этот язык был специально создан для проектирования систем управления непрерывными технологическими процессами (рис. 4.1).

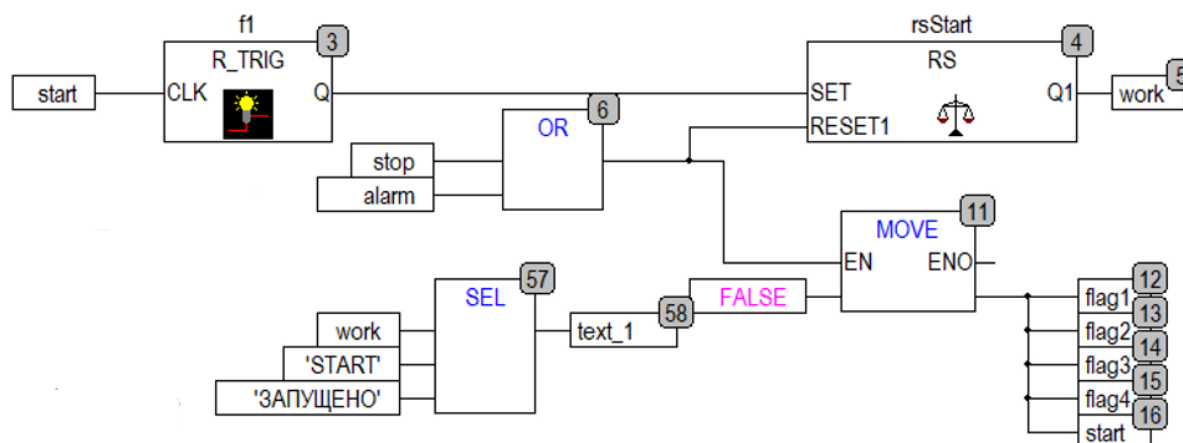


Рисунок 4.1 – Пример программы, реализованной на языке CFC

Логические операторы CoDeSys. Логическими операторами, используемыми в CoDeSys, являются: логическое «И», логическое «ИЛИ», логическое «исключающее ИЛИ», логическое «НЕ». Все логические операторы работают со следующими типами данных: BOOL, BYTE, WORD или DWORD.

Логическое «И». Логическое «И» представлено элементом «AND». В своей структуре имеет два входа и один выход (рис. 4.2). Логика работы элемента представлена в таблице 4.1.

Таблица 4.1 – Логика работы элемента логического «И»

1-ый вход	2-ой вход	выход
0	0	0
1	0	0
0	1	0
1	1	1

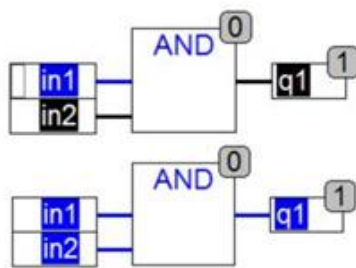


Рисунок 4.2 – Логическое «И»

Логическое «ИЛИ». Логическое «ИЛИ» представлено элементом «OR». В своей структуре имеет два входа и один выход (рис. 4.3). Логика работы элемента представлена в таблице 4.2.

Таблица 4.2 – Логика работы элемента логического «ИЛИ»

1-ый вход	2-ой вход	выход
0	0	0
1	0	1
0	1	1
1	1	1

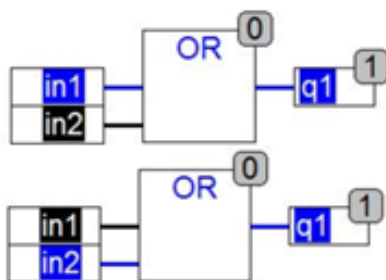


Рисунок 4.3 – Логическое «ИЛИ»

Логическое «исключающее ИЛИ». Логическое «исключающее ИЛИ» представлено элементом «XOR». В своей структуре имеет два входа и один выход (рис. 4.4). Логика работы элемента представлена в таблице 4.3.

Таблица 4.3 – Логика работы логического элемента «исключающее ИЛИ»

1-ый вход	2-ой вход	выход
0	0	0
1	0	1
0	1	1
1	1	0

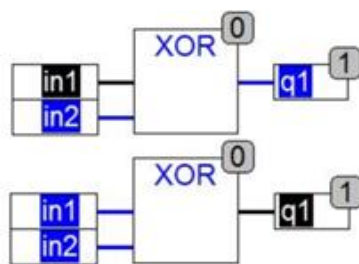


Рисунок 4.4 – Логическое «исключающее ИЛИ»

Логическое «НЕ». Логическое «НЕ» представлено элементом «NOT». В своей структуре имеет один вход и один выход (рис. 4.5). Логика работы элемента представлена в таблице 4.4.

Таблица 4.4 – Логика работы логического элемента «НЕ»

ВХОД	ВЫХОД
0	1
1	0

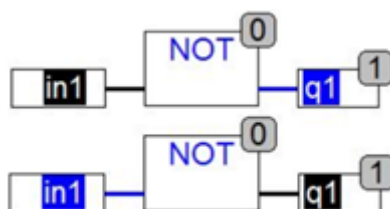


Рисунок 4.5 – Логическое «НЕ»

Арифметические операторы CoDeSys. Арифметическими операторами, используемыми в CoDeSys, являются: сложение, умножение, вычитание, деление, остаток от деления. Название блоков арифметических элементов может указываться как буквенно, так и графически.

Арифметическое «Сложение». Элемент «Сложение» в CoDeSys может быть представлено как в письменном виде (ADD), так и в графическом (+). В своей первоначальной структуре имеет два входа и один выход (рис. 4.6). Типы входных и выходных данных, с которыми используется элемент: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL и LREAL, TIME. Принцип работы: первое слагаемое (верхний вход) складывается со вторым слагаемым (нижний вход), в итоге получаем сумму чисел того же типа, что и на входе.

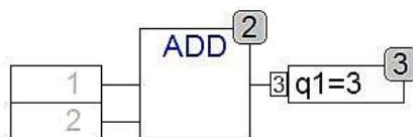


Рисунок 4.6 – Арифметический оператор «Сложение».

Арифметическое «Умножение». Элемент «Умножение» в CoDeSys может быть представлено как в письменном виде (MUL), так и в графическом. В своей первоначальной структуре имеет два входа и один выход (рис. 4.7). Типы входных и выходных данных, с которыми используется элемент: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT. Принцип работы: первый множитель (верхний вход) умножается со вторым множителем (нижний вход), в итоге получаем произведение чисел того же типа, что и на входе.



Рисунок 4.7 – Арифметический оператор «Умножение»

Арифметическое «Вычитание». Элемент «Вычитание» в CoDeSys может быть представлено как в письменном виде (SUB), так и в графическом (-). В своей структуре имеет два входа и один выход (рис. 4.8). Типы входных и выходных данных, с которыми используется элемент: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL и LREAL, TIME. Принцип работы следующий: уменьшаемое (верхний вход) вычитает вычитаемое (нижний вход), в итоге получаем разность чисел того же типа, что и на входе.

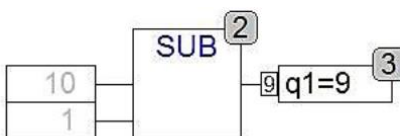


Рисунок 4.8 – Арифметический оператор «Вычитание»

Арифметическое «Деление». Элемент «Деление» в CoDeSys может быть представлено как в письменном виде (DIV), так и в графическом (/). В своей структуре имеет два входа и один выход (рис. 4.9). Типы входных и выходных данных, с которыми используется элемент: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT. Принцип работы следующий: делимое (верхний вход) делится на делитель (нижний вход), в итоге получаем частность чисел того же типа, что и на входе.

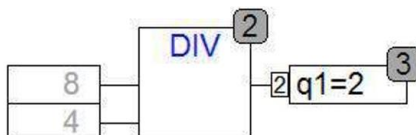


Рисунок 4.9 – Арифметический оператор «Деление».

Арифметический «Остаток от деления». Элемент «Остаток от деления» в CoDeSys обозначается писменном виде (MOD). В своей структуре имеет два входа и один выход (рис. 4.10). Типы входных и выходных данных, с которыми используется элемент: BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT. Принцип работы следующий: делимое (верхний вход) делится на делитель (нижний вход), в итоге получаем следующее:

- а) если делимое делится на делитель без остатка (например, $4/2$), то в результате на выходе блока мы получим значение, равное 0;
- б) если делимое делится на делитель с остатком, то выполняется следующие действия: находится число, которое меньше делителя и делится на делитель без остатка, затем от делимого отнимается это число, в результате получаем искомый результат.

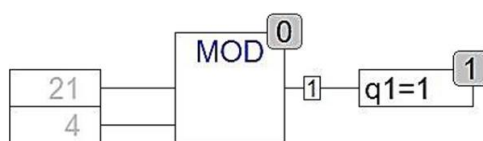


Рисунок 4.10 – Арифметический оператор «Деление».

Операторы выбора и ограничения. Селектор. Селектор – оператор побитной выборки. В своей структуре оператор имеет три входных элемента и один выходной. Первый вход селектора (верхний) имеет тип данных BOOL, остальные входы и выход могут быть любого типа данных. Однако, типы данных у оставшихся входов и выхода должны совпадать. Принцип работы : при подаче на первый вход логического нуля, выходному элементу присваивается значение второго входного элемента; если же на первый вход была подана логическая единица, то выходному элементу присваивается значение третьего входного элемента (рис. 4.11).

Операторы выбора наибольшего и наименьшего значения. Оператор выбора наибольшего значения (MAX) – оператор, который из приведённых значений выбирает наибольшее. В своей структуре имеет два входа и один выход (рис. 4.12).

Оператор выбора наименьшего значения (MIN) – оператор, который из приведённых значений выбирает наименьшее. В своей структуре имеет два входа и один выход.

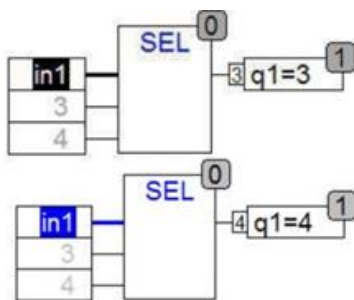


Рисунок 4.11 – Оператор «Селектор»

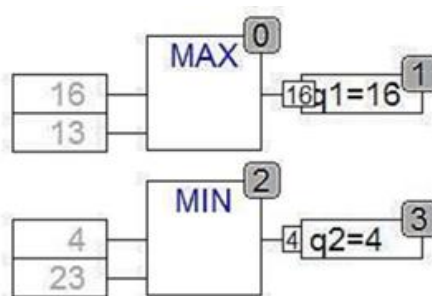


Рисунок 4.12 – Операторы выбора наименьшего и наибольшего значения

Ограничитель. Ограничитель (LIMIT) – оператор, который ограничивает значение переменной в заданном промежутке значений, который работает со всеми типами данных. В своей структуре оператор имеет три входа и один выход. Первому входу присваивается начальное значение промежутка, второй вход – переменная, которую мы ограничиваем, третий вход – конечное значение промежутка (рис. 4.13). Если значение лимитированной переменной выходит за указанный промежуток, то выходу присваивается значение промежутка, близкому к заданному.

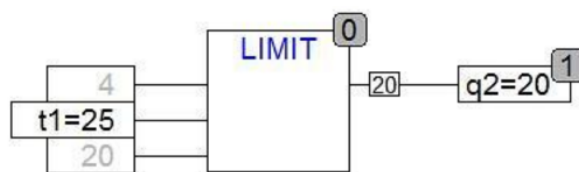


Рисунок 4.13 – Оператор «Ограничитель»

Оператор «Присвоение». Присвоение (MOVE) – оператор, который присваивает значение входной переменной выходной. Классический оператор в своей структуре имеет один вход и один выход (рис. 4.14). По сути, классический оператор выполняет функцию прямого присвоения и в своей классической компоновке не используется.

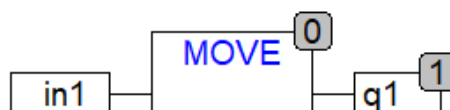


Рисунок 4.14 – Классическое представление оператора сравнения.

В качестве оператора блок «Присвоение» используется с дополнительными входами условиями срабатывания (EN/ENO) (рис. 4.15). Вход «EN» служит для условия срабатывания блока присвоения (рис. 4.16) и имеет тип данных BOOL. Принцип работы следующий: если на вход запуска работы блока присвоения (EN) приходит логическая единица, то значению выхода присваивается значения входа, иначе выход имеет свое состояние, которое ему было

присвоено ранее. Стоит заметить, что типы данных входного и выходного элемента должны совпадать.

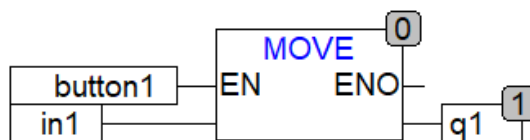


Рисунок 4.15 – Оператор «Присвоение» с дополнительным входом условия

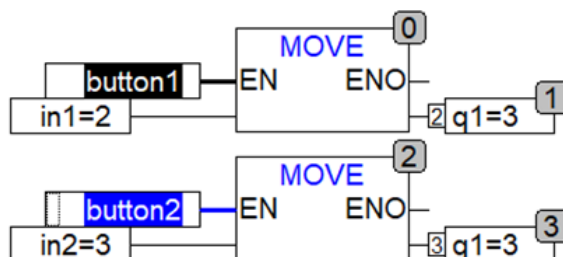


Рисунок 4.16 – Работа оператора «Присвоение»

Мультиплексор. Мультиплексор – это устройство с ячейками памяти, в которых хранится информация, в зависимости от выбора ячейки может передаваться на исполнительное устройство. В своей первоначальной структуре мультиплексор имеет 4 входа (один управляющий, три – хранящие данные) и один выход (рис. 4.17). Верхний вход является управляющим входом, имеющий тип данных INT, который в свою очередь указывает на регистр (ячейку памяти), в которой хранится информация (порядок регистров начинается с «0»). Последующие входы являются ячейками памяти (регистрах), в которых хранится необходимая информация. Типы данных входных элементов (регистров) и выходного элемента должны совпадать.

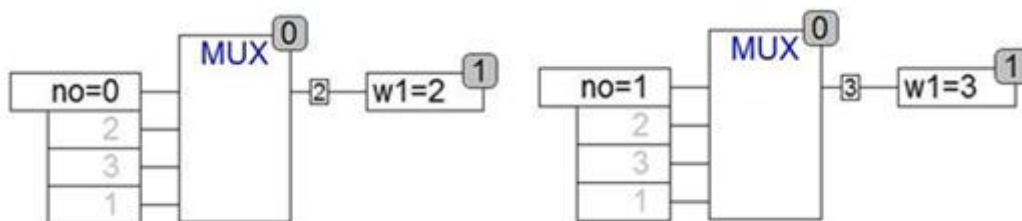


Рисунок 4.17 – Оператор «Мультиплексор»

Операторы сравнения. Операторами сравнения в CoDeSys являются: «Больше», «Меньше», «Больше или равно», «Меньше или равно», «Равно», и «Неравно». Операторы сравнения в своей структуре имеют два входа и один выход. На входе оператора переменные могут быть любого типа данных, главное, чтобы типы у этих переменных совпадали. Выходная переменная может иметь только один тип данных: BOOL.

Оператор сравнения «Больше». Оператор сравнения «Больше» может обозначаться письменно как «GT», так и символьно «>». Работает следующим образом: если значение первого входа больше значения второго – результатом будет логическая единица, иначе – логический ноль (рис. 4.18).

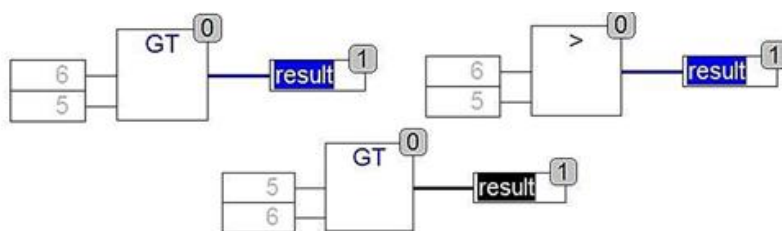


Рисунок 4.18 – Оператор сравнения «Больше»

Оператор сравнения «Меньше». Оператор сравнения «Меньше» может обозначаться письменно как «LT», так и символьно «<». Работает следующим образом: если значение первого входа меньше значения второго – результатом будет логическая единица, иначе – логический ноль (рис. 4.19).

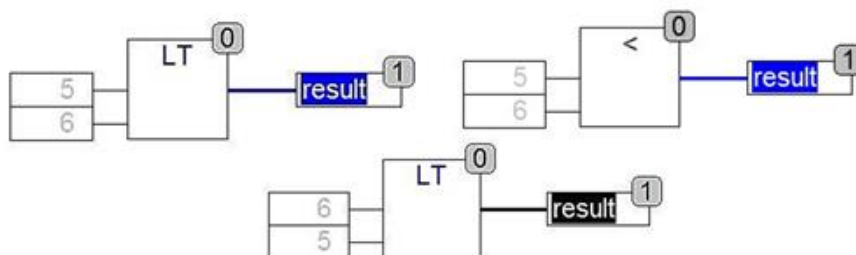


Рисунок 4.19 – Оператор сравнения «Меньше»

Оператор сравнения «Больше или равно». Оператор сравнения «Больше или равно» может обозначаться письменно как «GE», так и символьно «>=». Работает следующим образом: если значение первого входа больше или равно значению второго – результатом будет логическая единица, иначе – логический ноль (рис. 4.20).

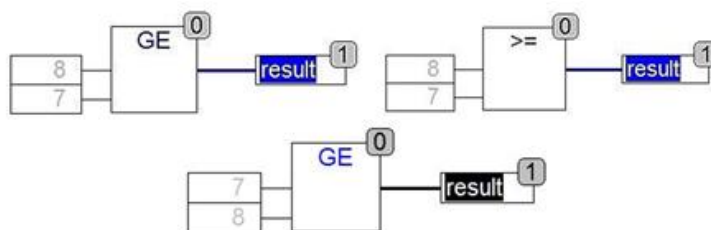


Рисунок 4.20 – Оператор сравнения «Больше или равно»

Оператор сравнения «Меньше или равно». Оператор сравнения «Меньше или равно» может обозначаться письменно как «LE», так и символьно «<=».

Работает следующим образом: если значение первого входа меньше или равно значению второго – результатом будет логическая единица, иначе – логический ноль (рис. 4.21).

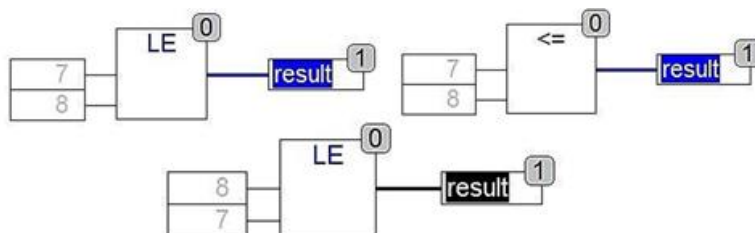


Рисунок 4.21 – Оператор сравнения «Меньше или равно»

Оператор сравнения «Равно». Оператор сравнения «Равно» может обозначаться письменно как «EQ», так и символьно «=». Работает следующим образом: если значение первого входа равно значению второго – результатом будет логическая единица, иначе – логический ноль (рис. 4.22).

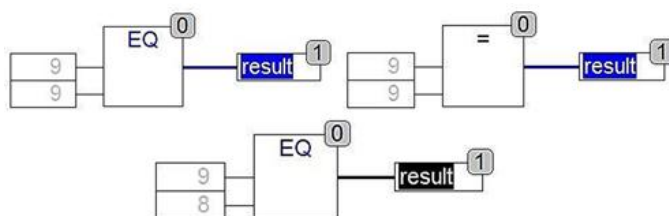


Рисунок 4.22 – Оператор сравнения «Равно»

Оператор сравнения «Не равно». Оператор сравнения «Не равно» может обозначаться письменно как «NE», так и символьно «<>». Выполняет противоположные функции относительно оператора «Равно» (рис. 4.23).

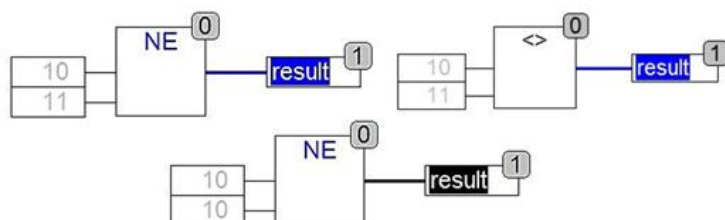


Рисунок 4.23 – Оператор сравнения «Не равно»

Практическая часть

Ход работы. *Реализация изменения состояния объекта управления с использованием нескольких устройств управления.* Объектом управления выступит лампочка (out1), устройствами управления – тумблеры (in7, in8, in9)

(рис. 4.24). Пользователь имеет возможность менять состояние объекта управления при помощи трёх вышеупомянутых устройств управления. Объект управления изменяет своё состояние при одновременной работе двух или более устройств управления (рис. 4.25). Реализовать программу на языке CFC в среде CoDeSys с использованием логических операторов.

Найти решение уравнению. Пользователю дано четыре входных переменных (R1–R4) и одна выходная переменная (R5), который имеет право изменять значения входных переменных.

Вычислить значение выходной переменной, исходя из уравнения (рис. 4.26):

$$R5 = \frac{(R1 - 10)}{(R2 + 11)} \cdot (R3 - 4) + R4.$$

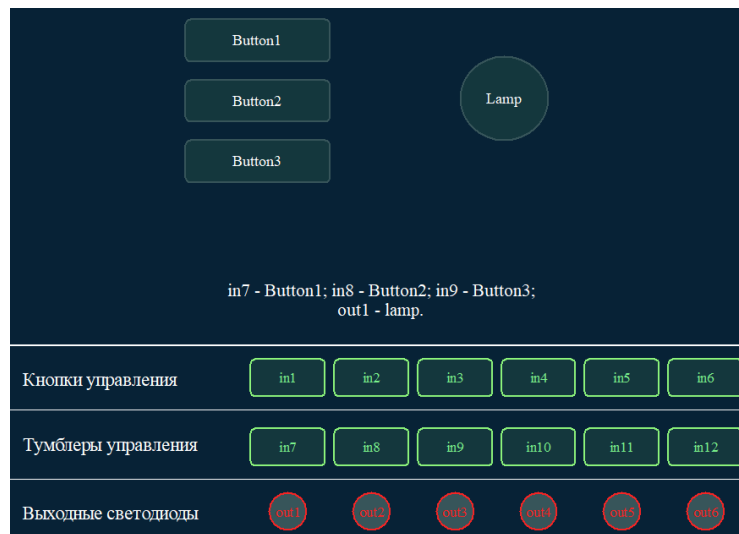


Рисунок 4.24 – Изменение состояния объекта управления с использованием нескольких устройств управления



Рисунок 4.25 – Реализация управления над объектом.

Результат уравнения доступен при нажатии на кнопку in1 (рис. 4.27). Для проверки правильного написания уравнения рекомендуется ввести следующие значения для входных параметров: R1 = 60, R2 = 89, R3 = 20, R4 = 4. Необходимо реализовать данную задачу с использованием арифметических операторов.

?

$$R5 = \frac{(R1-10)}{(R2+11)} * (R3-4) + R4$$

Установить значение R1 R1 = 60.00

Установить значение R2 R2 = 89.00

Установить значение R3 R3 = 20.00

Установить значение R4 R4 = 4.00

in1 - показать ответ.

Кнопки управления in1 in2 in3 in4 in5 in6

Тумблеры управления in7 in8 in9 in10 in11 in12

Рисунок 4.26 – Исходные данные

?

$$R5 = \frac{(R1-10)}{(R2+11)} * (R3-4) + R4$$

Установить значение R1 R1 = 60.00

Установить значение R2 R2 = 89.00

Установить значение R3 R3 = 20.00

Установить значение R4 R4 = 4.00

Ответ: R5 = 12.00

in1 - показать ответ.

Кнопки управления in1 in2 in3 in4 in5 in6

Тумблеры управления in7 in8 in9 in10 in11 in12

Рисунок 4.27 – Результат решения уравнения

Реализация конфигурирования объектом управления (вентилятором) в зависимости от влияющих факторов (температуры) с использованием операторов выбора и ограничения, сравнения. Пользователю дан объект управления – вентилятор. Вентилятор имеет **5 режимов** работы в зависимости

от температуры: **1 режим** – скорость вентилятора равна 0, при температуре от 0 до 20 °С включительно (рис. 4.28); **2 режим** – скорость вентилятора равна 1, при температуре от 21 до 40 °С включительно; **3 режим** – скорость вентилятора равна 2, при температуре от 41 до 60 °С включительно (рис. 4.29); **4 режим** – скорость вентилятора равна 3, при температуре от 61 до 80 °С включительно; **5 режим** – скорость вентилятора равна 4, при температуре от 81 до 100 °С включительно.



Рисунок 4.28 – Режим работы вентилятора при температуре 15 °С

Температура ограничена и находится в промежутке от 0 до 100 °С. Поставленную задачу необходимо реализовать с использованием операторов, изученных ранее.



Рисунок 4.29 – Режим работы вентилятора при температуре 55 °С

Дополнительное задание. В зависимости от режимов работы вентилятора выходные элементы out1–out5 (светодиоды) меняют свое состояние: out1 загорается при первом режиме работы, out2 – при втором, out3 – при третьем, out4 – при четвертом, out5 – при пятом (рис. 4.29).

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Описание графического языка программирования CFC.
4. Логические и арифметические операторы CoDeSys.
5. Особенности операторов «Сложение» и «Умножение».
6. Операторы выбора и ограничения CoDeSys.
7. Операторы сравнения CoDeSys.
8. Реализация изменения состояния объекта управления с использованием нескольких устройств управления.
9. Найти решение уравнению.
10. Реализация конфигурирования объектом управления (вентилятором) в зависимости от влияющих факторов (температуры) с использованием операторов выбора и ограничения, сравнения.
11. Вывод.

Контрольные вопросы:

1. Графический язык программирования CFC.
2. Логические операторы CoDeSys: описание, структура, обозначение, примеры.
3. Арифметические операторы CoDeSys: описание, структура, обозначение, примеры.
4. Какие типы данных имеют входы и выходы у логических операторов?
5. Какие типы данных имеют входы и выходы у арифметических операторов?
6. Операторы сравнения CoDeSys: описание, структура, обозначение, примеры.
7. Какие типы данных имеют входы и выходы у операторов выбора и ограничения?
8. Какие типы данных имеют входы и выходы у операторов сравнения?
9. Особенности операторов «Мультиплексор» и «Селектор».

Лабораторная работа 5

«Детекторы фронтов. Триггеры памяти»

Цель работы: изучить принцип работы детекторов фронтов: детектора переднего фронта (r-trigger) и детектора заднего фронта (f-trigger), ознакомиться и изучить принцип работы триггеров памяти: триггер памяти с приоритетом на уставку значения – SR-триггер, триггер памяти с приоритетом на сброс значения – RS-триггер.

Теоретическая часть

Детекторы фронтов. Детекторы фронтов – элементы, которые способствуют уловить начало или окончание процесса. В своей структуре детекторы фронта имеют один вход и один выход, которые работают только с одним типом данных – BOOL. Более подробно рассмотрим каждый элемент.

Детектор переднего фронта «R-trigger». Детектор переднего фронта «R-trigger» – это триггер, который улавливает передний фронт входного сигнала (начальный момент работы) и выдаёт единичный импульс на выходе. В CoDeSys детектор переднего фронта обозначается как «r_trig» и выглядит так, как показано на рисунке 5.1.

Детектор заднего фронта «F-trigger». Детектор заднего фронта «F-trigger» – это триггер, который улавливает задний фронт входного сигнала (момент окончания работы) и выдаёт единичный импульс на выходе. В CoDeSys детектор заднего фронта обозначается как «f_trig» (рис. 5.2).

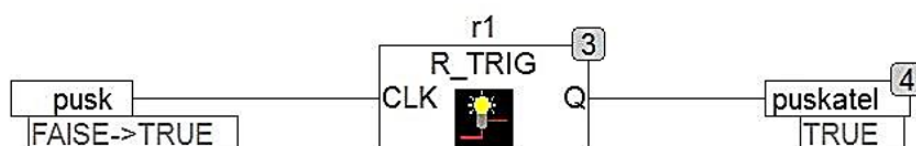


Рисунок 5.1 – Внешний вид и принцип работы R-триггера

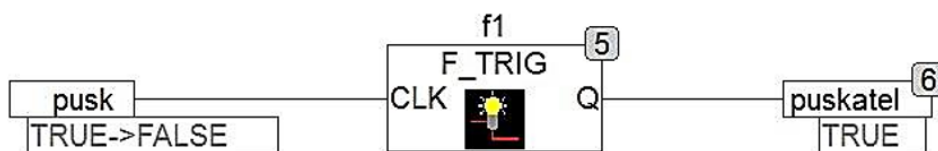


Рисунок 5.2 – Внешний вид и принцип работы F-триггера

Триггеры памяти. Триггеры памяти используются для установки или сброса значения выходной переменной с последующим запоминанием состояния переменной. По принципу работы триггеры памяти подразделяются на: триггер памяти с приоритетом на уставку значения (SR-триггер) и триггер па-

мнотри с приоритетом на сброс значения (RS-триггер). В своей структуре триггеры памяти имеют два входа: первый вход (SET) выполняет присвоение выходу логической единицы (установка значения), второй (RESET) – присвоение выходу логического нуля (сброс значения), и один выход (Q). Входы и выход данных элементов имеют тип данных, такой как BOOL.

Триггер памяти с приоритетом на установку значения – SR-триггер. SR-триггер предназначен для установления и сброса логической единицы на выходе с приоритетом на установку значения. При одновременной подаче логической единицы на входы SET и RESET триггера, на выходе Q получим логическую единицу (рис. 5.3).

Триггер памяти с приоритетом на сброс значения – RS-триггер. RS-триггер предназначен для установления и сброса логической единицы на выходе с приоритетом на сброс значения. При одновременной подаче логической единицы на входы SET и RESET триггера, на выходе Q получим логический ноль (рис. 5.4).

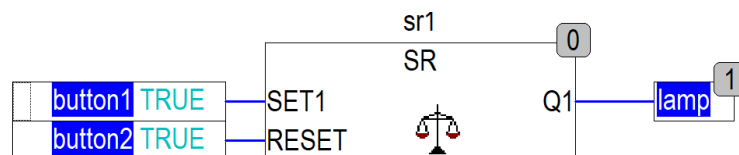


Рисунок 5.3 – Принцип работы SR-триггера

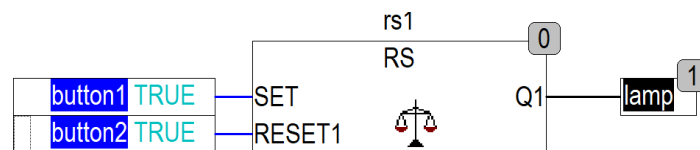


Рисунок 5.4 – Принцип работы RS-триггера

Практическая часть

Ход работы. Реализация изменения состояния объекта управления (двигателя) с использованием органов управления при возникновении нештатных происшествий (пожара либо аварии).

Пользователю дан объект управления – «двигатель» (out1) и органы управления – кнопки «Пуск» (in1) и «Стоп» (in2). При нажатии на кнопку «Пуск» двигатель запускается (рис. 5.5) (активизируется выход out1) и продолжает свою работу.

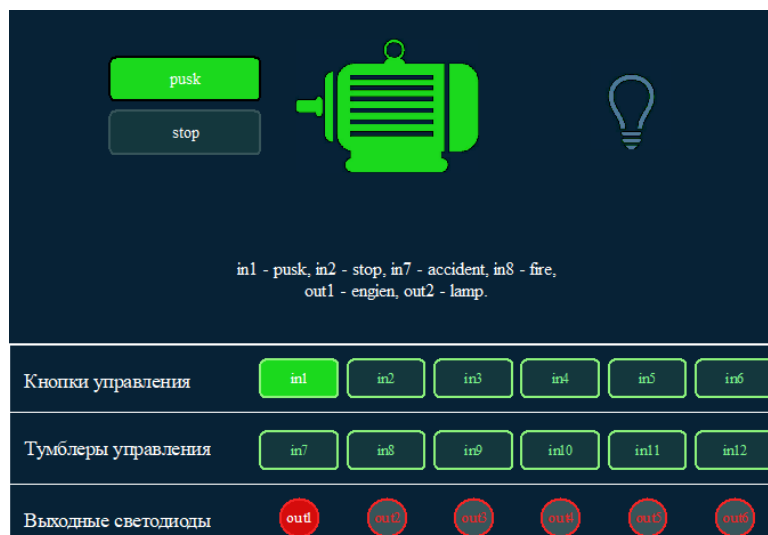


Рисунок 5.5 – Запуск двигателя при нажатии на кнопку «Пуск»

При нажатии на кнопку «Стоп» двигатель останавливается (выход out1 деактивируется).

При возникновении нештатных ситуаций: пожара (модуляция при помощи нажатия на тумблер in7) или аварии (модуляция при помощи нажатия на тумблер in8), двигатель автоматически выключается и при этом загорается лампа, сигнализирующая о нештатной ситуации (out2) (рисунок 5.6). Запуск двигателя не возможен до устранения неисправности. После устранения неисправности система работает в обычном режиме. Реализовать задачу с использованием элементов, рассмотренных в теоретической части.



Рисунок 5.6 – Обработка срабатывания нештатной ситуации

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.

3. Операторы выбора и ограничения CoDeSys.
4. Операторы сравнения CoDeSys.
5. Детекторы фронта CoDeSys.
6. SR-триггер: описание, структура, принцип действия.
7. RS-триггер: описание, структура, принцип действия.
8. Реализация изменения состояния объекта управления (двигателя) с использованием органов управления при возникновении нештатных происшествий (пожара либо аварии).
9. Вывод.

Контрольные вопросы:

1. Какие типы данных имеют входы и выходы у детекторов фронтов?
2. Принцип работы детекторов фронтов.
3. Детекторы фронтов: описание, структура, обозначение, примеры.
4. SR-триггер: описание, структура, обозначение, примеры.
5. RS-триггер: описание, структура, обозначение, примеры.
6. Какие типы данных имеют входы и выходы у детекторов фронтов?
7. Какие типы данных имеют входы и выходы у триггеров?
8. Принцип работы триггеров памяти.

Лабораторная работа 6 «Таймеры. Счётчики импульсов в среде CoDeSys»

Цель работы: ознакомиться и изучить принцип работы таймеров: таймер с задержкой включения, таймер с задержкой выключения, генератор одиночного импульса; изучить принцип работы счётчиков: инкрементного, декрементного и универсального.

Теоретическая часть

Таймеры. Таймер – устройство, предназначенное для отсчёта заданного времени. В среде CoDeSys существует три вида таймера: таймер с задержкой включения, таймер с задержкой выключения, генератор единичного импульса (рис. 6.1). По своей структуре таймер включает в себя два входа и два выхода. Первый вход таймера (IN) имеет тип данных BOOL и предназначен для запуска таймера, второй вход (PT) имеет тип данных TIME и предназначен для установки времени работы таймера. Первый выход таймера (Q) имеет тип данных BOOL и является результирующим элементом отработки таймера, второй выход (ET) имеет тип данных TIME и служит для отображения работы таймера в реальном времени (сколько отработал таймер в данный момент).

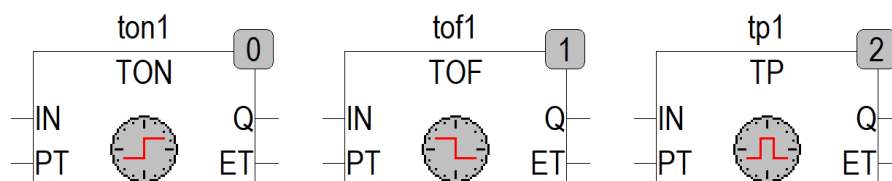


Рисунок 6.1 – Таймеры: с задержкой включения (TON), с задержкой выключения (TOF) и генератор одиночного импульса (TP)

Таймер с задержкой включения. Таймер с задержкой включения (TON) служит для задержки сигнала на выходе относительно входа на заданное значение (время). Работает таймер по переднему фронту входного сигнала. Схема работы таймера представлена на рисунке 6.2. Принцип работы следующий: при подаче сигнала на вход запускается таймер. Как только таймер отсчитал положенное время (вход PT), на выходе таймера получаем логическую единицу. После снятия сигнала со входа, сигнал на выходе также пропадает. Если убрать сигнал до того, как таймер досчитал до конца, то таймер обнулится, состояние выхода будет неизменным (логический ноль).

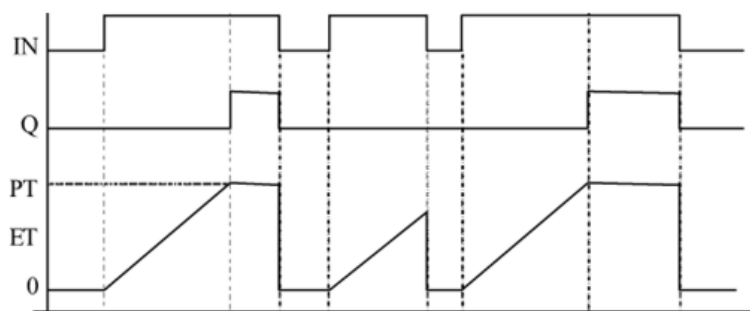


Рисунок 6.2 – Схема работы таймера TON

Таймер с задержкой выключения. Таймер с задержкой выключения (TOF) служит для поддержания сигнала на выходе после отключения сигнала на входе на заданное значение (время). Работает таймер по заднему фронту входного сигнала. Схема работы таймера представлена на рисунке 6.3. Принцип работы следующий: при подаче сигнала на вход мы получаем сигнал на выходе (логическую единицу). Как только мы убираем сигнал со входа, запускается работа таймера. Как только таймер отсчитал положенное время (вход РТ), на выходе таймера получаем логический ноль. Тем самым этот таймер «продлевает жизнь» выходному сигналу. Если убрать сигнал на входе, а затем снова подать его до того, как таймер досчитал до конца, то таймер обнулится, состояние выхода будет неизменным (логическая единица).

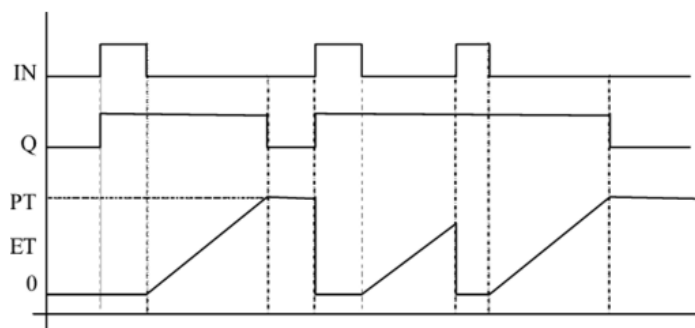


Рисунок 6.3 – Схема работы таймера TOF

Генератор одиночного импульса. Генератор одиночного импульса (TP) служит для генерирования одиночного импульса на заданное значение (время). Работает таймер по переднему фронту входного сигнала. Схема работы таймера представлена на рисунке 6.4. Принцип работы следующий: при подаче сигнала на вход мы получаем сигнал на выходе (логическую единицу) и запускается работа таймера. Как только таймер отсчитал положенное время (вход РТ), на выходе таймера получаем логический ноль. Тем самым этот таймер генерирует одиночный импульс выходному сигналу, который равен заданному времени. Особенности работы данного таймера: если неоднократно подавать сигнал на вход таймера в течение его работы (момент запуска первым сигналом), состоя-

ние выхода будет неизменным (логическая единица) в течение заданного времени, т. е. таймер запускается по первому входному сигналу, остальные сигналы он игнорирует (сигналы, которые подаются во время работы).

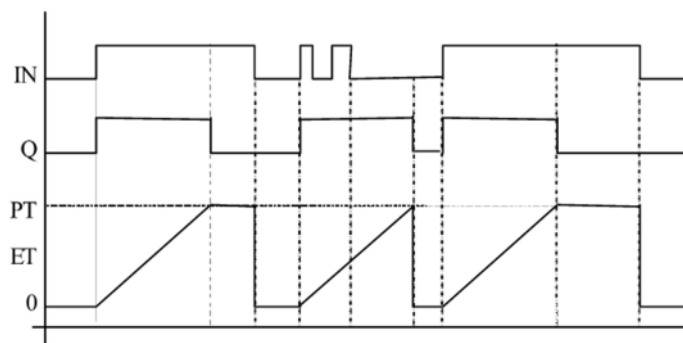


Рисунок 6.4 – Схема работы таймера ТР

Счётчики. Счётчик – устройство для подсчёта импульсов, поступающих на его вход. Различают три вида счётчиков: инкрементный (CTU), декрементный (CTD) и универсальный (CTUD). Рассмотрим каждый из счётчиков.

Инкрементный счётчик. Инкрементный счётчик (CTU) – это счётчик, который ведёт счёт на увеличение. Данный счётчик имеет 3 входа и два выхода (рис. 6.5). Принцип работы следующий: при подаче сигнала на вход CU счётчик считает этот импульс и запоминает его ($CV = 1$). С последующей подачей импульсов на вход счётчика выход CV меняет свое значение на единицу больше (+1). При достижении счётчиком уставки (PV), выход счётчика (Q) даёт логическую единицу и сохраняет сигнал. При подаче сигнала на вход RESET счётчик обнуляется ($CV = 0$, $Q = \text{FALSE}$). *Примечание:* счётчик работает только до достижения максимального значения используемого типа данных (WORD). Переполнения не происходит.

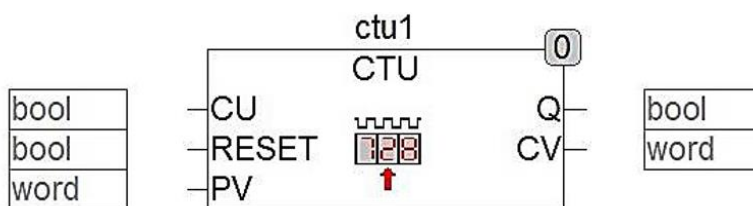


Рисунок 6.5 – Инкрементный счётчик CTU

Декрементный счётчик. Декрементный счётчик (CTD) – это счётчик, который ведёт счёт на уменьшение. Данный счётчик имеет 3 входа и два выхода (рис. 6.6). Принцип работы следующий: необходимо загрузить в счётчик данные (уставку) счётчику, при подаче сигнала на вход CD счётчик считает этот импульс и отнимает от загруженного значения. С последующей подачей импульсов на вход счётчика выход CV меняет свое значение на единицу мень-

ше (-1). При достижении счётчиком нуля ($CV = 0$), выход счётчика (Q) даёт логическую единицу. *Примечание:* в счетчик в параметр уставки нельзя загрузить значение, которое превышает максимальное значение используемого типа данных (WORD).

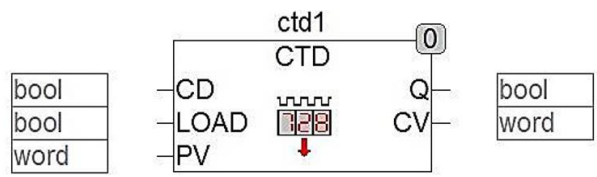


Рисунок 6.6 – Декрементный счётчик CTD

Универсальный счётчик. Универсальный счётчик – это счётчик, который работает как на увеличение, так и на уменьшение счёта. В своей структуре, счётчик имеет элементы как инкрементного, так и декрементного счётчиков (рис. 6.7). Принцип работы аналогичен двум вышеупомянутым счётчикам.

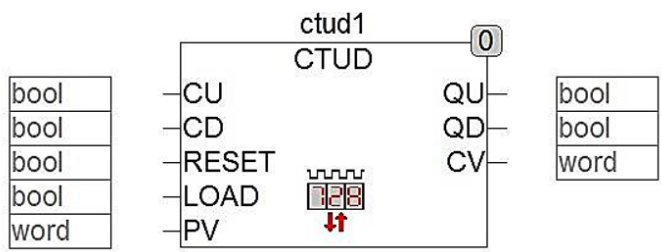


Рисунок 6.7– Универсальный счётчик

Практическая часть

Ход работы. Управление системой запуска котла. Система включает в себя два исполнительных механизма: вентилятор (out1) и котёл (out2) (рис. 6.8).

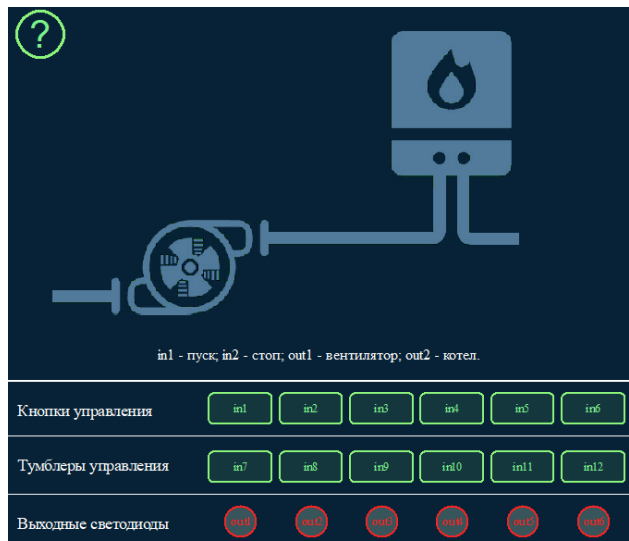


Рисунок 6.8 – Исходный вид системы

Запуск системы осуществляется путём нажатия на кнопку «пуск» (in1). После нажатия на кнопку пуск начинается подготовка к работе системы: включается вентилятор (рис. 6.9).

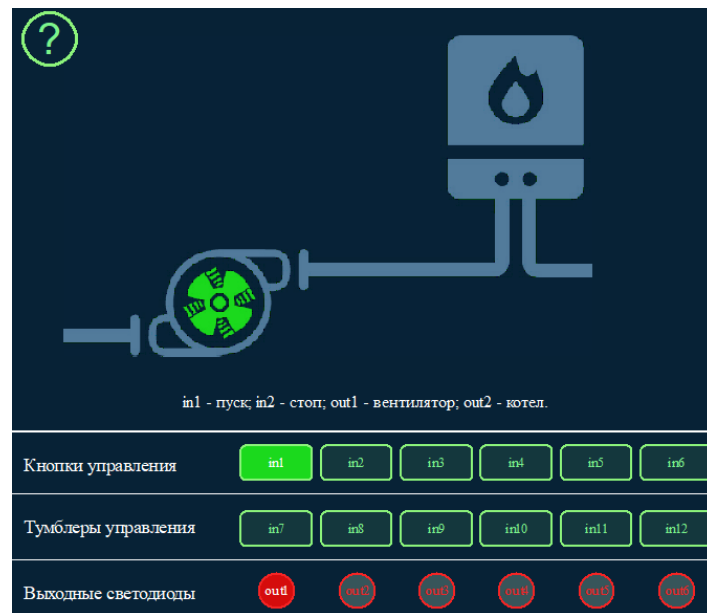


Рисунок 6.9 – Запуск вентилятора путем нажатия на кнопку «пуск»

Затем с истечением 5 секунд включается насос и система работает в штатном режиме. При нажатии на кнопку «стоп» (in2) основная система останавливается: выключается котёл (рис. 6.10), после прохождения 3 секунд останавливается вентилятор, что свидетельствует о полной остановке системы. Реализовать задание с использованием элементов, изученных ранее.

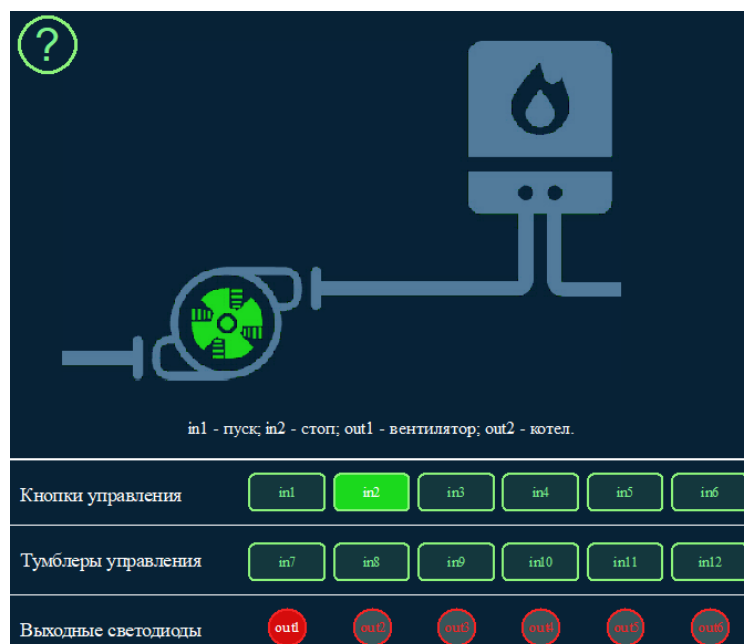


Рисунок 6.10 – Нажатие на кнопку «стоп» и остановка котла

Управление автоматическим освещением внутри помещения в зависимости от нахождения людей. Необходимо посчитать количество людей в комнате (рис. 6.11).



Рисунок 6.11 – Объект управления освещением

Если в комнате находятся люди, то необходимо включить свет (out1). На входе установлены два датчика: внешний (in7) и внутренний (in8). Когда человек заходит в комнату, срабатывает внешний датчик, затем внутренний – количество людей в комнате увеличивается на 1 (рис. 6.12).

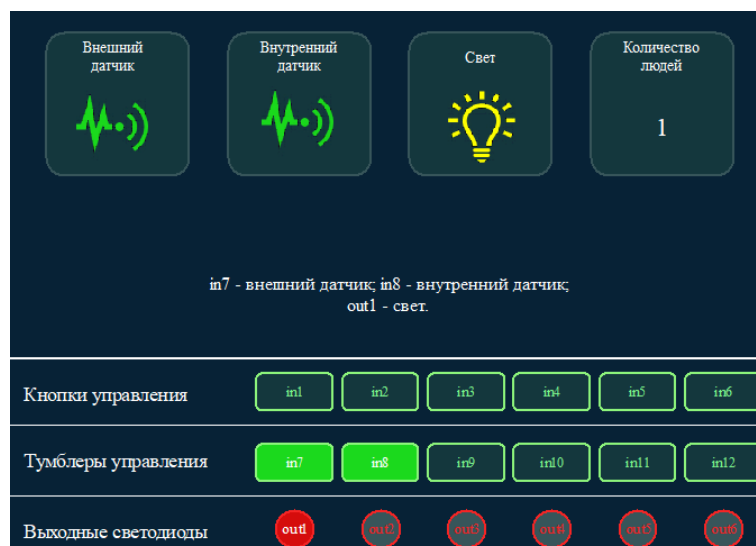


Рисунок 6.12 – Отработка условия входа человека в комнату

Далее перестает работать внешний датчик, затем внутренний. Когда человек выходит из комнаты, срабатывает внутренний датчик, затем внешний –

количество людей уменьшается на 1. Далее перестаёт работать внутренний датчик, затем внешний.

Рассматриваем идеальную: человек проходит без разворотов и остановки, в один момент проходит только 1 человек. Необходимо реализовать задачу с использованием элементов, изученных ранее.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Таймеры: описание, структура, принцип действия.
4. Инкрементный счётчик: описание, структура, принцип действия.
5. Декрементный счётчик: описание, структура, принцип действия.
6. Универсальный счётчик: описание, структура, принцип действия.
7. Реализация управлением системой запуска котла.
8. Реализация автоматического освещения внутри помещения в зависимости от нахождения людей.
9. Вывод.

Контрольные вопросы:

1. Таймер с задержкой включения TON: описание, структура, обозначение, особенности работы.
2. Таймер с задержкой выключения TOF: описание, структура, обозначение, особенности работы.
3. Генератор одиночного импульса TP: описание, структура, обозначение, особенности работы.
4. Инкрементный счётчик: описание, структура, принцип действия.
5. Декрементный счётчик: описание, структура, принцип действия.
6. Универсальный счётчик: описание, структура, принцип действия.
7. Особенности работы инкрементного счётчика.
8. Особенности работы декрементного счётчика.

Лабораторная работа 7

«Протокол обмена данными ModBus. Методика создания запросов в CoDeSys. Работа ПЛК в режиме Master»

Цель работы: ознакомиться и изучить методику создания запросов в CoDeSys,

Теоретическая часть

Протокол обмена данными ModBus. Modbus – открытый коммуникационный протокол, основанный на архитектуре ведущий-ведомый (англ. *master-slave*; в стандарте Modbus используются термины *client-server*). Широко применяется в промышленности для организации связи между электронными устройствами. Может использоваться для передачи данных через последовательные линии связи RS-485, RS-422, RS-232 и сети TCP/IP (Modbus TCP).

Для передачи пакета по физическим линиям связи PDU помещается в другой пакет, содержащий дополнительные поля. Этот пакет носит название ADU (Application Data Unit). Формат ADU зависит от типа линии связи. Существуют три варианта ADU, два для передачи данных через асинхронный интерфейс и один – через TCP/IP сети:

- 1) Modbus ASCII – для обмена используются только ASCII символы;
- 2) Modbus RTU – компактный двоичный вариант;
- 3) Modbus TCP – для передачи данных через TCP/IP-соединение.

Практическая часть

Ход работы. Методика создания запросов в CoDeSys. Для настройки ПЛК в режиме Master необходимо перейти в складку «Ресурсы → Конфигурация ПЛК», выбрать (PLC110_30 и добавить подэлемент «ModBus(Master)» (рис. 7.1).

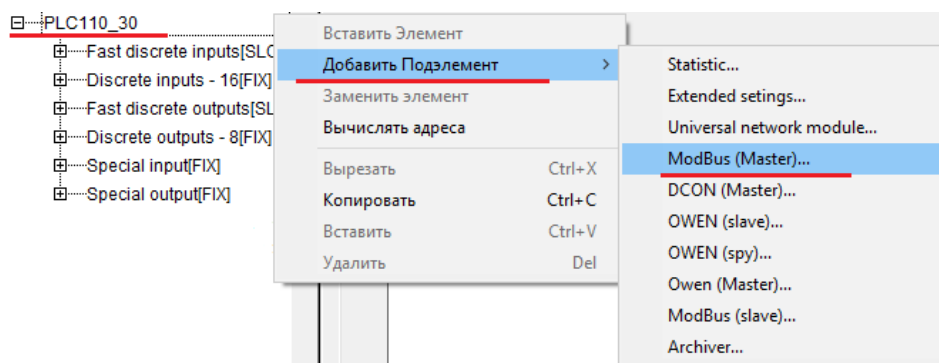


Рисунок 7.1 – Добавление подэлемента

Затем необходимо сконфигурировать подэлемент: настроить параметры модуля, выбрать интерфейс связи и произвести его конфигурирование (рис. 7.2). Далее происходит заполнение элемента, исходя от выбранного устройства.

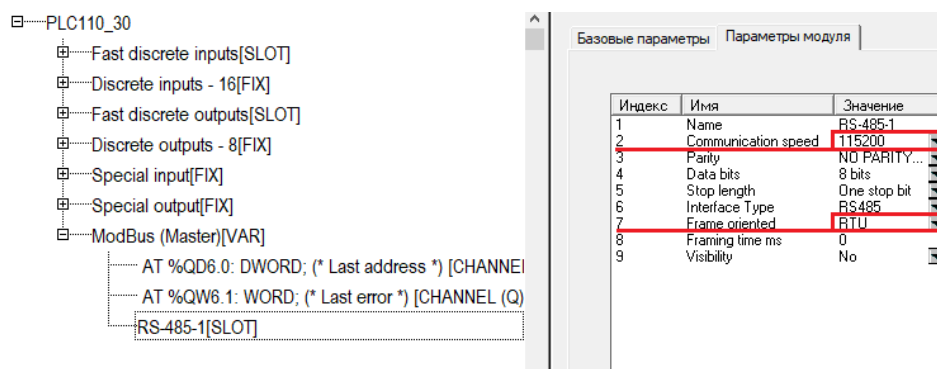


Рисунок 7.2 – Пример конфигурирования подэлемента

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Протокол обмена данными ModBus.
4. Методика создания запросов в CoDeSys.
5. Вывод.

Контрольные вопросы:

1. Протокол обмена данными ModBus: описание, структура, принцип действия.
2. Методика создания запросов в CoDeSys: основные этапы при создании запросов.
3. Интерфейс подключения RS-232.
4. Интерфейс подключения RS-485.
5. В чём отличия интерфейса подключения RS-485 от RS-232?

Лабораторная работа 8

«Модули ввода-вывода Mx110. Модуль дискретного вывода МУ110-224.16Р. Работа ПЛК в режиме Master».

Цель работы: ознакомиться и изучить характеристики модулей ввода-вывода Mx110, изучить методику конфигурирования модуля дискретного вывода МУ110-224.16Р, изучить методику настройки ПЛК в режиме Master.

Теоретическая часть

Модули ввода-вывода Mx110. Модули ввода-вывода Mx110 предназначены для управления по сигналам из сети RS-485 встроенными дискретными ВЭ, используемыми для подключения исполнительных механизмов с дискретным управлением, и сбора данных с дискретных входов модуля с передачей их в сеть RS-485. Коммуникационные возможности модулей:

- интерфейс – RS-485;
- поддерживаемые протоколы – Modbus RTU, Modbus ASCII, OVEN, DCON;
- скорость обмена данными по RS-485 – 2400...115200 бит/с.

Особенности модулей ввода-вывода:

- встроенные входы могут работать в режиме счетчика импульсов частотой до 1 кГц;
- встроенные выходы могут работать в режиме генерации ШИМ-сигналов до 1 Гц;
- дополнительная логика работы дискретных входов и выходов (прямая / «НЕ» / «И» / «ИЛИ» / один импульс / ШИМ / триггер);
- автоматическое определение протокола;
- обновление встроенного программного обеспечения по RS-485;
- поддержка облачного сервиса OwenCloud (при использовании сетевого шлюза ПМ210).

Модуль дискретного вывода МУ110-224.16Р. Модуль дискретного вывода МУ110-224.16Р предназначен для управления по сигналам из сети RS-485 встроенными дискретными ВЭ, используемыми для подключения исполнительных механизмов с дискретным управлением, и сбора данных с дискретных входов модуля с передачей их в сеть RS-485

Электрическая схема подключения к выводам модуля представлена на рисунке 8.1.

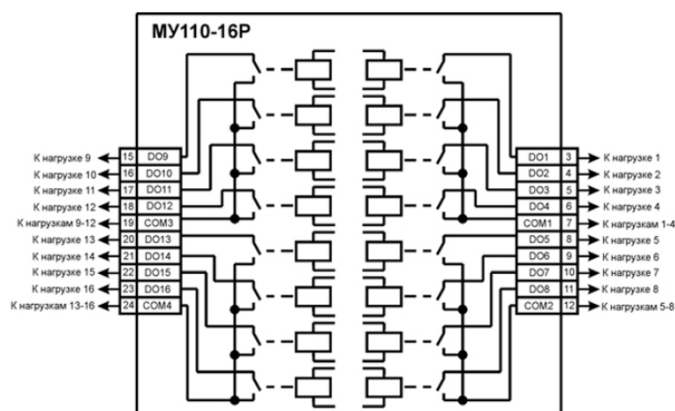


Рисунок 8.1 – Подключение нагрузки выводам модуля

Технические характеристики модуля дискретного вывода МУ110-224.16Р:

- количество выходов – 16;
- тип выходов – электромагнитное реле;
- максимальная нагрузочная способность дискретных выходов – 3 А (при переменном напряжении не более ~250 В 50 Гц и $\cos \varphi > 0,4$) или 3 А (при постоянном напряжении не более 30 В);
- тип питания – универсальное ~230 В/ 24 В;
- напряжение питания – переменное: ~90...264 В (номинальное ~230 В) частотой 47...63 Гц или постоянное: 18...30 В (номинальное – 24 В);
- потребляемая мощность – не более 12 ВА;

Методика конфигурирования при подключении модуля дискретных выходов. Основные этапы методики конфигурирования описаны в лабораторной работе 10. После настройки базовых принципов необходимо добавить универсальное Modbus устройство (рис. 8.2), произвести его конфигурирование (TCP-порт – 502, NetMode – Serial, адрес slave – 11, режим работы – By pool time), затем наполнить подэлемент необходимыми переменными (*Register output module*). Далее настроить данные переменные (рис. 8.3). Конфигурирование модуля закончено. **Важно! Переменным можно присваивать только уникальное значение регистров (не должны повторяться).**

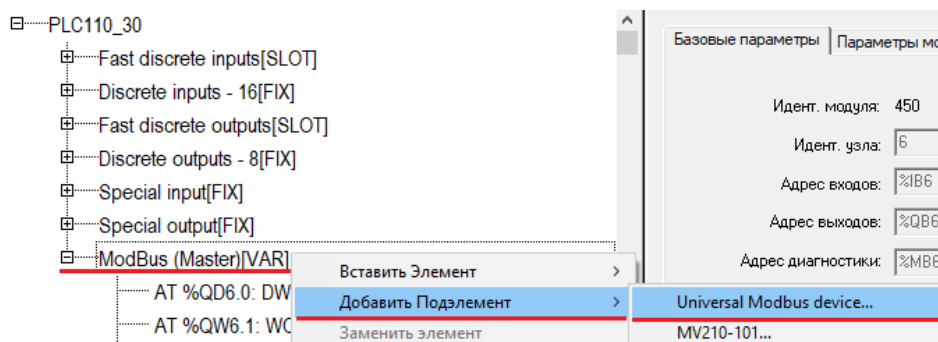


Рисунок 8.2 – Добавление универсального Modbus-устройства

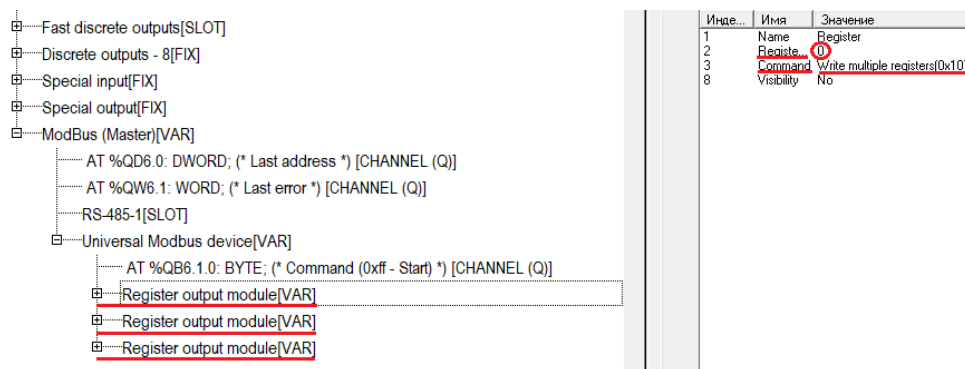


Рисунок 8.3 – Конфигурирование устройства переменными

Практическая часть

Ход работы. Реализация управлением группой фонтанов. Пользователь с помощью кнопок старт (in1) и стоп (in2) запускает и останавливает работу шести фонтанов (рисунок 8.4). Фонтаны работают в следующей последовательности тактов: 0 – все фонтаны выключены, 1 – работает только 1-ый фонтан (out1), 2 – работает только 2-ой фонтан (out2), 3 – работает только 3-ий фонтан (out3), 4 – работает только 4-ый фонтан (out4), 5 – работает только 5-ый фонтан (out5), 6 – работает только 6-ой фонтан (out6); 7 – работает только 1-ый фонтан (out1); 8 – работают 1-ый и 2-ой фонтаны (out1 и out2); 9 – работают 1–3-ий фонтаны (out1–out3); 10 – работают 1–4-ый фонтаны (out1–out4); 11 – работают 1–5-ый фонтаны (out1–out5); 12 – работают все фонтаны (out1–out6); 13 – все фонтаны выключены; 14,16 – повтор 12 такта; 15,17 – повтор 13 такта.

Длительность одного такта составляет 1 секунду. Процесс циклический. За выходы out1–out3 принять выходы ПЛК, за выходы out4–out6 – выходы модуля МУ110-224.16Р. Реализовать задание с использованием материала, изученного ранее.

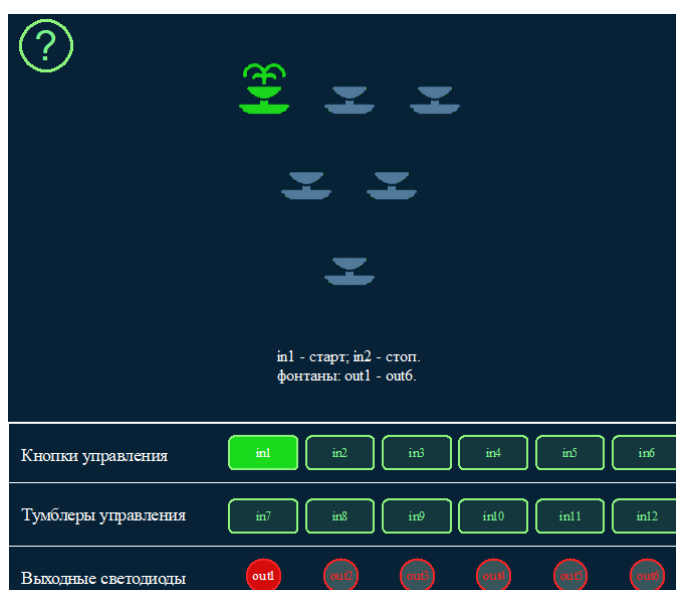


Рисунок 8.4 – Запуск системы

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Модули ввода-вывода Мх110.
4. Модуль дискретного вывода МУ110-224.16Р.
5. Методика конфигурирования при подключении модуля дискретных выходов.
6. Реализация управлением группой фонтанов.
7. Вывод.

Контрольные вопросы:

1. Для чего предназначены модули ввода-вывода Мх110?
2. В чём особенности модулей ввода-вывода Мх110?
3. Для чего предназначен модуль дискретного вывода МУ110-224.16Р?
4. Основные технические характеристики модуль дискретного вывода.
5. В чём состоит методика конфигурирования при подключении модуля дискретных выходов?
6. Какая функция используется при работе с переменными модуля?

Лабораторная работа 9

«Настройка ПЛК для работы с TPM202 и TPM210 (ПЛК в роли Master)»

Цель работы: изучение принципов формирования карты регистров ПЛК в режиме ModBus (Master).

Теоретическая часть

Контроллеры модификаций ПЛК110 выпускаются в конструктивном исполнении для крепления на DIN-рейке или на щите. По боковым продольным сторонам контроллера под прозрачными откидными крышками расположены съемные клеммные колодки, служащие для подключения дискретных датчиков, исполнительных механизмов, интерфейсов RS-485 и клеммы встроенного источника постоянного напряжения 24 В. Порядок разъединения/соединения клеммной колодки, подключения дискретных датчиков и исполнительных механизмов описан в документации к данному контроллеру. На верхней боковой стороне относительно лицевой панели ПЛК110 расположен соединитель интерфейса Ethernet типа RJ45. Светодиодный индикатор красного (или оранжевого) цвета в соединителе интерфейса Ethernet свидетельствует об установлении связи, работа зеленого светодиода свидетельствует о приеме либо передаче данных. На лицевой панели ПЛК110 расположены соединители интерфейсов RS-232, Debug RS-232. Порт Debug RS-232 предназначен для программирования контроллера, но также может быть использован для подключения Hayes-совместимых модемов (в том числе GSM), а также устройств, работающих по протоколам Modbus, OVEN или DCON. На лицевой панели ПЛК110 расположен соединитель интерфейса USB Device. Схема подключения терморегуляторов с ПЛК представлена на рисунке 9.1.

На передней панели имеются три кнопки:

- кнопка «F1» может применяться как дополнительный дискретный вход контроллера, исполняющий функцию, заданную пользовательской программой;
- кнопка «Старт/Стоп» предназначена для запуска и остановки пользовательской программы контроллера, она также может быть настроена для использования в качестве дополнительного дискретного входа контроллера;
- скрытая кнопка «Сброс» предназначена для перезагрузки контроллера, нажать ее возможно только тонким заостренным предметом диаметром не более 3 мм.

Внимание! После записи в контроллер пользовательской программы при любом его включении программа начинает выполняться автоматически (нажимать на кнопку «Старт/Стоп» не требуется).

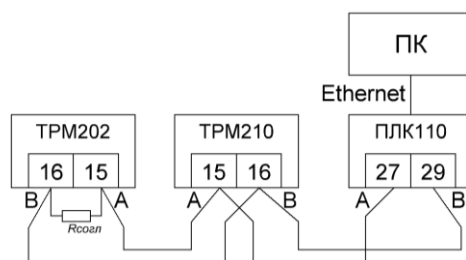


Рисунок 9.1 – Последовательность соединения приборов

Практическая часть

Ход работы *Настройка ПЛК для работы с TPM202 и TPM210 (ПЛК в роли Master)*. Для настройки ПЛК необходимо выполнить следующее:

1. Запустить на рабочем столе среду программирования CoDeSys v2.3.
2. Создать новый проект.
3. Выбрать целевую платформу.
4. Выбрать тип программного компонента и язык реализации.
5. Перейти во вкладку конфигурация.
6. Настроить конфигурацию:
 - 6.1 Добавить управляющее устройство (Modbus).
 - 6.2 Выбрать интерфейс подключения.
 - 6.3 Создать карту регистров (только для функции чтения).
7. Сохранить проект.
8. Включить лабораторного стенда в сеть.
9. Включить тумблеры автоматов на лабораторном стенде, после чего загорится индикатор «ВКЛ» на блоке питания БП30Б-ДЗ.
10. Включить устройства.
11. Загрузить проект в ПЛК.
12. Снять данные, полученные при подключении к устройствам.
13. Обесточить устройства.
14. Выключить лабораторный стенд.

Требования к содержанию отчёта.

1. Название лабораторной работы.
2. Цель работы.
3. Скриншот заполненной конфигурации.
4. Скриншот полученных данных.
5. Вывод о проделанной работе.

Контрольные вопросы.

1. Состав программы CoDeSys.
2. Назначение ModBus. Какие функции поддерживает ModBus?
3. Перечислить типы переменных и их характеристики.
4. Перечислить поддерживаемых стандартных кодов ошибок ModBus и их характеристики.

Лабораторная работа 10

«Настройка ПЛК для работы с ТРМ202 и ТРМ210 (ПЛК в роли Slave)»

Цель работы: изучение принципов формирования карты регистров ПЛК в режиме ModBus (Slave).

Теоретическая часть

Контроллеры ОВЕН ПЛК110 – линейка программируемых моноблочных контроллеров с дискретными входами/выходами на борту для автоматизации средних систем (рис. 10.1). Оптимальны для построения систем автоматизации среднего уровня и распределенных систем управления.

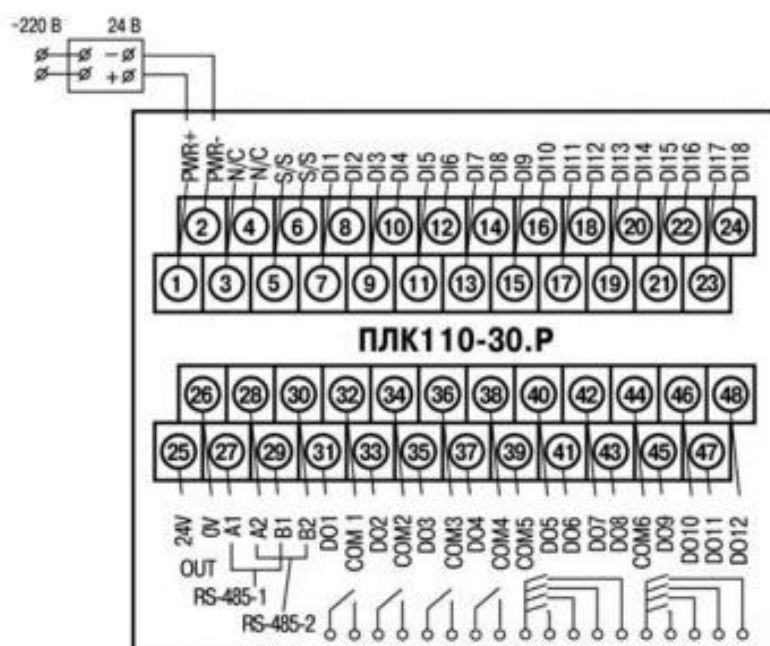


Рисунок 10.1 – Схема расположения и назначение клемм на ПЛК110

ПЛК110 содержит цифровые (дискретные) входы и выходы, количество которых различно в разных исполнениях контроллера. Обработка значений с входов осуществляется пользовательской программой ПЛК. На переднюю панель контроллера выведена светодиодная индикация о состоянии дискретных входов и выходов, наличии питания, наличии связи со средой программирования CoDeSys и о работе контроллера. Свечение индикатора «ПИТАНИЕ» отображает наличие питания контроллера. Индикатор «СВЯЗЬ» отображает состояние подключения контроллера к среде программирования CoDeSys. При наличии связи со средой CoDeSys, индикатор светится. Для связи контроллера со средой CoDeSys может использоваться один из каналов – RS232 (Debug), Ethernet или USB Device. Индикатор «РАБОТА» отображает состояние пользовательской программы. Индикатор светится, если пользовательская программа

выполняется. Индикаторы входов и выходов отображают состояние соответствующих дискретных входов и выходов контроллера. Индикаторы состояния входов светятся, если соответствующий вход замкнут.

Практическая часть

Ход работы. Настройка ПЛК для работы с TPM202 и TPM210 (ПЛК в роли slave). Для настройки ПЛК необходимо выполнить следующее:

1. Открыть ранее созданный проект.
2. Перейти во вкладку конфигурация.
3. Настроить конфигурацию:
 - 3.1. Добавить подчиняющееся устройство (Slave).
 - 3.2. Выбор интерфейса подключения.
 - 3.3. Создание карты регистров (только для функции чтения).
4. Организовать передачу данных внутри ПЛК (основная программа).
5. Сохранить проект.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Скриншот заполненной конфигурации.
4. Вывод о проделанной работе.

Контрольные вопросы:

1. Цели и задачи ModBus (slave).
2. Какие функции поддерживает ModBus?
3. Перечислить типы переменных и их характеристики.
4. Какие основные типы данных используются в ModBus (slave)?
5. Перечислить поддерживаемых стандартных кодов ошибок ModBus и их характеристики.

Лабораторная работа 11

«Изучение принципов работы панели оператора с контроллером»

Цель работы: изучение методики и принципов работы панели оператора с контроллером.

Теоретическая часть

Общие сведения о панелях операторов СПЗхх. Панель оператора с сенсорным экраном СПЗхх представляет собой устройство класса «человеко-машинный интерфейс», предназначенное для загрузки управляющей программы (проекта) функционирования программируемого логического контроллера (ПЛК) или др. приборов, к которым подключается панель, мониторинга функционирования и редактирования значений параметров функционирования (рис. 11.1). Позволяет отображать на экране ход выполнения технологического процесса и редактировать значения параметров, отвечающих за функционирование системы. Логика работы панели СПЗхх определяется пользователем в процессе конфигурирования на ПК с использованием программного обеспечения «Конфигуратор СП300».

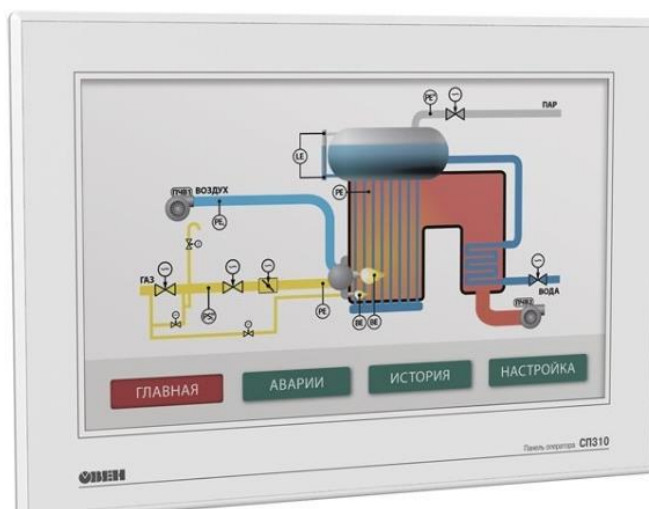


Рисунок 11.1 – Лицевая сторона панели СПЗхх

Панель СПЗхх предназначена для выполнения следующих функций:

- отображение состояния управляемого объекта в режиме реального времени, с использованием графических пиктограмм (индикаторы, графики, линейки, условные обозначения оборудования и т. д.);
- отображение сенсорных элементов, при помощи которых оператор осуществляет непосредственное управление функционированием объекта;

- управление функционированием ПЛК и/или других приборов; запись и чтение значений регистров ПЛК и/или других приборов, к которым подключается панель;
- оперативное изменение режима работы (изменение внешнего вида экрана и интерфейса управления, параметров управления и пр.) путем загрузки нового проекта;
- работа в режиме Master или Slave.

Переключатель. Элемент «Переключатель» используется для изменения состояния выбранного бита (ВКЛ/ВЫКЛ). Уникальным параметром элемента «Переключатель» является вкладка «Действие» (рис. 11.2).

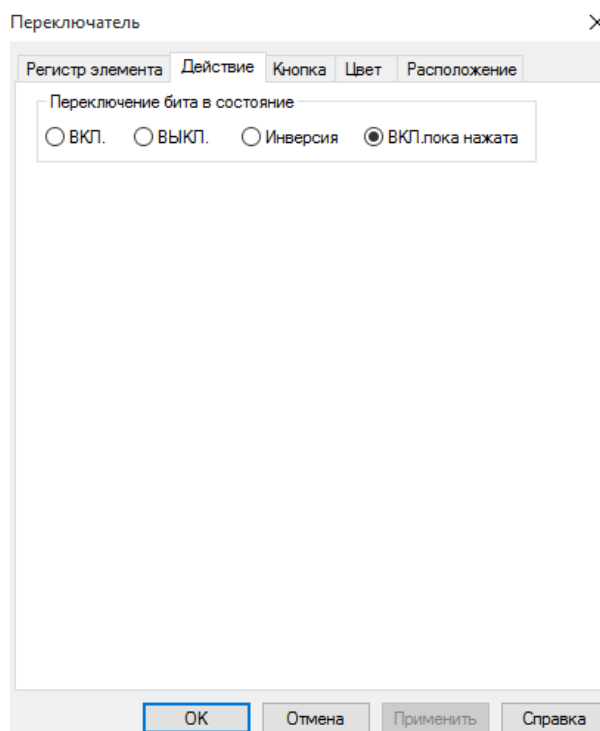


Рисунок 11.2 – Параметры элемента «Переключатель» во вкладке «Действие»

Вкладка «Действие» – действие, выполняемое при нажатии на элемент:

- *ВКЛ* – переключение бита в состояние ВКЛ;
- *ВЫКЛ* – переключение бита в состояние ВЫКЛ;
- *Инверсия* – переключение элемента в состояние, противоположное текущему;
- *ВКЛ, пока нажата* – переключение бита в состояние ВКЛ на время зажатия элемента, после отпускания – переключение в состояние ВЫКЛ.

Переключатель с индикацией. Элемент «Переключатель с индикацией» используется для отображения и переключения состояний привязанных бит (ВКЛ/ВЫКЛ). К элементу может быть привязан как один бит (отображение бита 1, переключение бита 1), так и два (отображение бита 1, переключение бита 2).

Цифровой дисплей. Элемент «Цифровой дисплей» используется для отображения численного значения привязанного регистра (рис. 11.3).

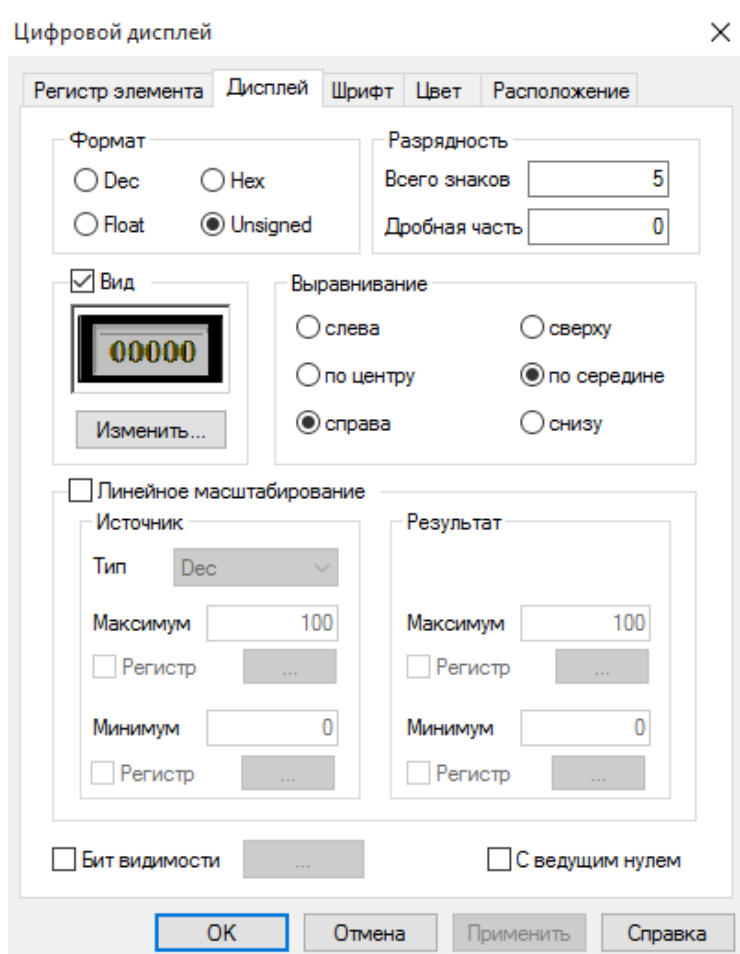


Рисунок 11.3 – Параметры элемента Цифровой дисплей, вкладка Дисплей

Рассмотрим уникальные параметры элемента «Цифровой дисплей». Вкладка «Дисплей» содержит следующие разделы:

1. **Формат** – формат данных отображаемых элементом:
 - *Dec* – целочисленный знаковый;
 - *Float* – с плавающей точкой;
 - *Hex* – шестнадцатеричный;
 - *Unsigned* – целочисленный беззнаковый (только положительные значения).
2. **Разрядность** – полное количество знаков, отображаемых элементом, и количество знаков, отображаемых после запятой.
3. **Вид** – при отсутствии галочки, элемент не имеет фона. При наличии – можно выбрать фоновое изображение из шаблонов элемента с помощью кнопки Изменить.
4. **Выравнивание** – горизонтальное и вертикальное выравнивание области вывода элемента.

5. **Линейное масштабирование** – для исходных данных (значений регистра, привязанного к элементу), указывается тип и верхний/нижний предел. Для результата (отображаемого значения) указывается верхний/нижний предел. Вместо фиксированных значений можно указать соответствующие регистры.
6. **Бит видимости** – при наличии галочки можно выбрать бит, значение которого будет определять видимость элемента (ВЫКЛ – элемент невидим, ВКЛ – элемент виден).
7. **С ведущим нулем** – При наличии галочки, перед значением регистра отображаются ведущие нули.

Настройка ПЛК. Проект для контроллера создан в среде CODESYS V2.3.

После этого в дереве проекта во вкладке «Ресурсы» следует перейти в конфигурацию ПЛК. В соответствующем окне необходимо нажать на соответствующую конфигурацию правой клавишей мыши по активному элементу (PLC110_30), далее нажимаем «Добавить Подэлемент», выбираем необходимый нам элемент, а именно ModBus (slave) (рис. 11.4).

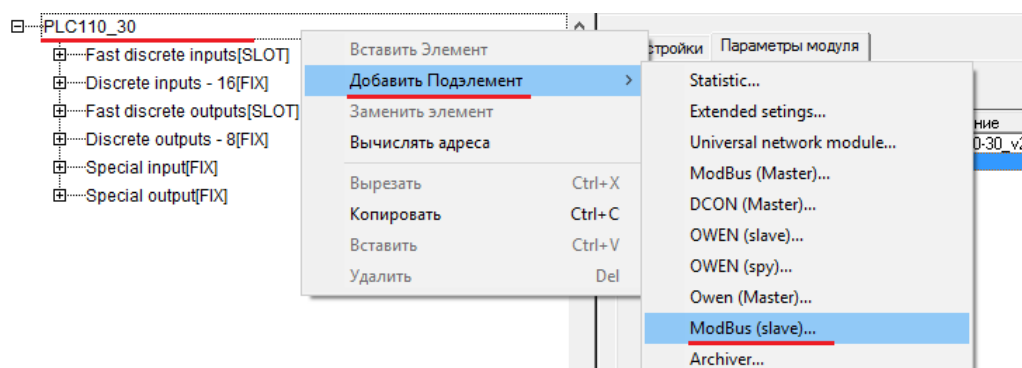


Рисунок 11.4 – Добавление подэлемента ModBus (slave)

При добавлении подэлемента он отобразится в дереве конфигурации ПЛК (рис. 11.5). Перейдём к данному подэлементу, который имеет базовые параметры: идентификация модуля, идентификация узла, адрес входов, адрес выходов, адрес диагностики и комментарий (рис. 11.5). Все параметры, за исключением комментария, являются постоянными и неизменными.

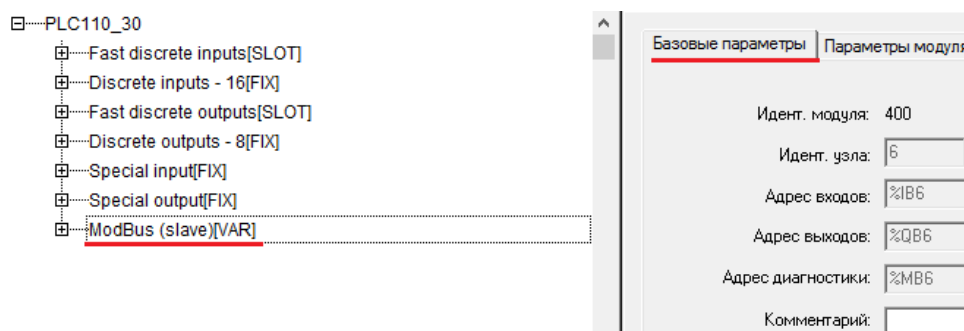


Рисунок 11.5 – Базовый параметры ModBus (slave)

Перейдём во вкладку «Параметры модуля» (рис. 11.6). Данная вкладка имеет два индекса: имя (Name) и адрес (Address). Оставляем параметры без изменения (рис. 11.6).



Рисунок 11.6 – Параметры модуля ModBus (slave)

Рассмотрим структуру данного подэлемента. Для этого либо левой клавишей мыши нажмем на него, либо нажмем на знак «+», расположенный слева от подэлемента. При открытии видим стандартный элемент – Modbus[FIX]. Данный элемент служит для настройки связи между ПЛК и устройством (рис. 11.7). В зависимости от типа устройства пользователь может выбрать конкретный тип интерфейса, выбрав его из соответствующего ряда интерфейсов (рис. 11.7):

1) RS-232 (Recommended Standard 232) – стандарт физического уровня для асинхронного интерфейса (UART), поддерживающий последовательный порт персональных компьютеров. RS-232 обеспечивает передачу данных и некоторых специальных сигналов между терминалом (Data Terminal Equipment) и коммуникационным устройством (Data Communications Equipment) на расстояние до 15 метров на максимальной скорости (115200).

2) Debug RS-232 – аналог RS-232 с возможностью конфигурирования контроллера, записи и выгрузки пакетов данных и информации.

3) RS-485 (Recommended Standard 485) – стандарт физического уровня для асинхронного интерфейса. Регламентирует электрические параметры полудуплексной многоточечной дифференциальной линии связи типа «общая шина», в котором для передачи и приёма данных используется одна витая пара проводов, иногда сопровождаемая экранирующей оплеткой или общим проводом, где передача данных осуществляется с помощью дифференциальных сигналов. Разница напряжений между проводниками одной полярности означает логическую единицу, разница другой полярности – ноль.

4) Modem – подмодуль, принцип работы которого схож с работой обычного модема.

5) FTDI – аппаратный мост для интерфейса USB, который производит и осуществляет поддержку устройств и соответствующих программных драйверов для преобразования последовательной передачи данных по RS-232 или уровней TTL в сигналы шины USB, для того чтобы дать возможность совме-

менным компьютерам использовать простой классический RS-232 интерфейс, а разработчикам не приходилось вникать в сложности USB интерфейса.

6) TCP (Transmission Control Protocol) – сетевая модель передачи данных, представленных в цифровом виде. Модель описывает способ передачи данных от источника информации к получателю. В модели предполагается прохождение информации через четыре уровня, каждый из которых описывается правилом (протоколом передачи). Наборы правил, решающих задачу по передаче данных, составляют стек протоколов передачи данных, на которых базируется Интернет.

7) Cloud – это совокупность нескольких IT-ресурсов. В состав облака могут входить сервера, каналы связи, сетевое оборудование, дисковое пространство и различное ПО. Главная цель создания облачных систем – передача доступа к хранящимся в облаке данным определённому кругу пользователей.

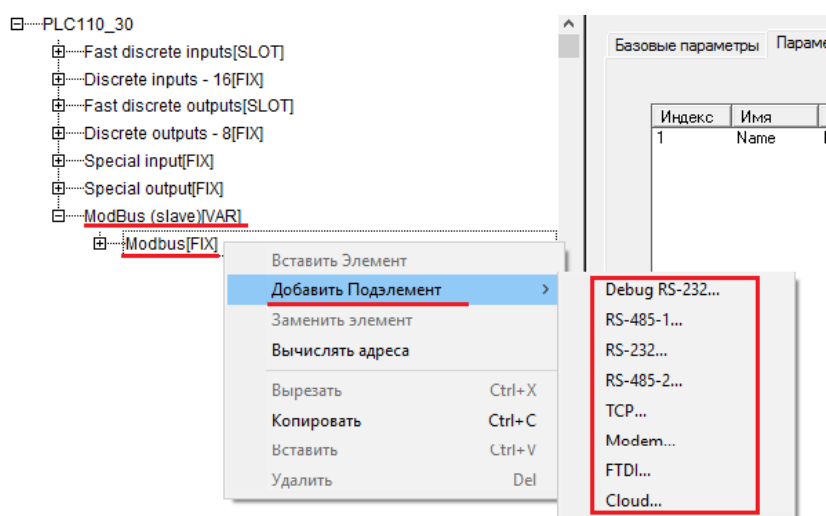


Рисунок 11.7 – Выбор интерфейса подключения

Настройка СПЗхх. Для того чтобы использовать панель СПЗхх в режиме «Modbus RTU Master» необходимо в настройках проекта на вкладке «Устройство» для нужного СОМ-порта выбрать режим «Modbus RTU Master» и указать его настройки. В соответствии с примером используется PLC-порт с настройками 115200-8-N-1. Настройка панели завершена.

Практическая часть

Ход работы. «Реализация совместной работы СПЗхх и ПЛК». Подключите контроллер ПЛК110 к панели оператора СП307. В контроллер загрузите соответствующую программу.

Создайте новый пустой пользовательский проект. Выполните настройки проекта: *СОМ устройство – Modbus RTU Master, скорость – 115200, бит данных 8 бит, контроль четности – отсутствует, изменить порядок регистров – установить галочку.*

Настройка Экран 1. Разместить 4 элемента Индикатор  для отображения состояния реальных входов контроллера ПЛК110:

1й индикатор: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 00;*

2й индикатор: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 01;*

3й индикатор: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 02;*

4й индикатор: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 03.*

Разместить 4 элемента Переключатель с индикацией  для управления состоянием реальных выходов контроллера ПЛК110:

1й переключатель с индикацией работающий в режиме тумблера: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 08;*

2й переключатель с индикацией работающий в режиме тумблера: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 09;*

3й переключатель с индикацией работающий в режиме тумблера: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 10;*

4й переключатель с индикацией работающий в режиме тумблера: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 0, Бит 11.*

Разместить 2 элемента Цифровой дисплей  для отображения численных значений, переданных из контроллера ПЛК110:

1й цифровой дисплей: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 2, Тип – DWord, Формат – Float, Всего знаков – 4, Дробная часть – 1 ;*

1й цифровой дисплей: *Порт – PLC порт, Адрес – 1, Функция – 4х, Регистр – 4, Тип – DWord, Формат – Float, Всего знаков – 4, Дробная часть – 1 .*

Пример расположения элементов изображен на рисунке 11.8.



Рисунок 11.8 – Расположение элементов на Экран1

Запустите online эмуляцию («Файл → Эмуляция OnLine») и исследуйте принцип работы элементов (рис. 11.9). По окончании работы, закройте окно эмуляции.

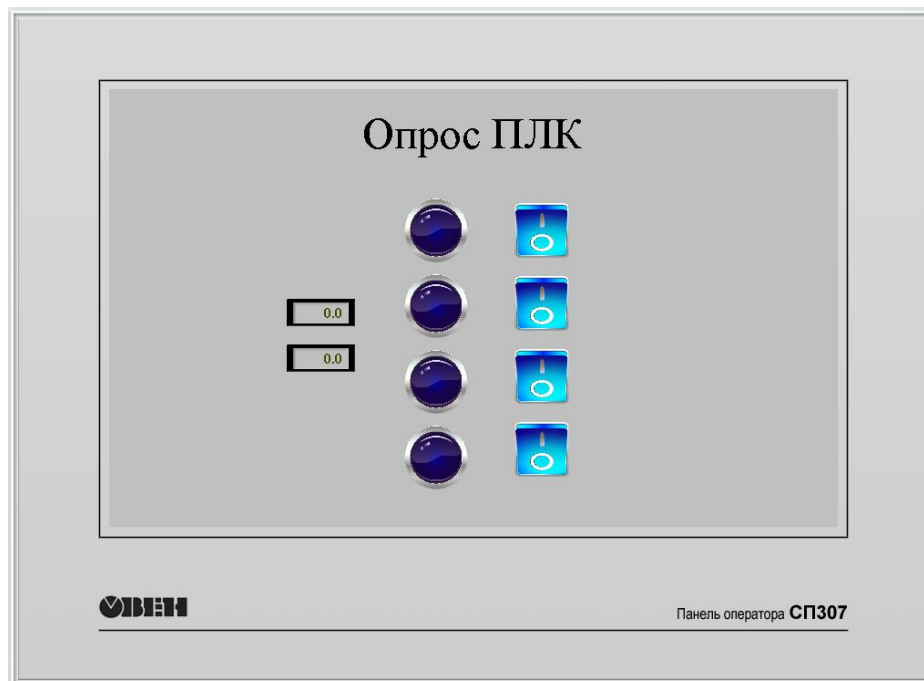


Рисунок 11.9 – Эмуляция работы программы

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Настройка ПЛК.
4. Настройка СПЗхх.
5. Реализация совместной работы СПЗхх и ПЛК.

Контрольные вопросы:

- 1) Для чего предназначена панель оператора СПЗхх?
- 2) Какое программное обеспечение применяется для конфигурирования панели СПЗхх?
- 3) Какие параметры элемента «Переключатель» содержатся во вкладке «Действия»?
- 4) Какие параметры элемента «Индикатор» содержатся во вкладке «Индикатор»?
- 5) Какие форматы данных отображаемых элементом содержит поле «Формат»?

Лабораторная работа 12 «Настройка OPC-сервера»

Цель работы: изучить методику создания и настройки OPC-сервера.

Теоретическая часть

Owen OPC Server предназначен для осуществления обмена данными между приборами с жестко заданной логикой, свободно программируемыми устройствами ОВЕН и любыми SCADA-системами (рис. 12.1). Теперь для настройки обмена по протоколам ОВЕН и Modbus используется один Owen OPC Server.

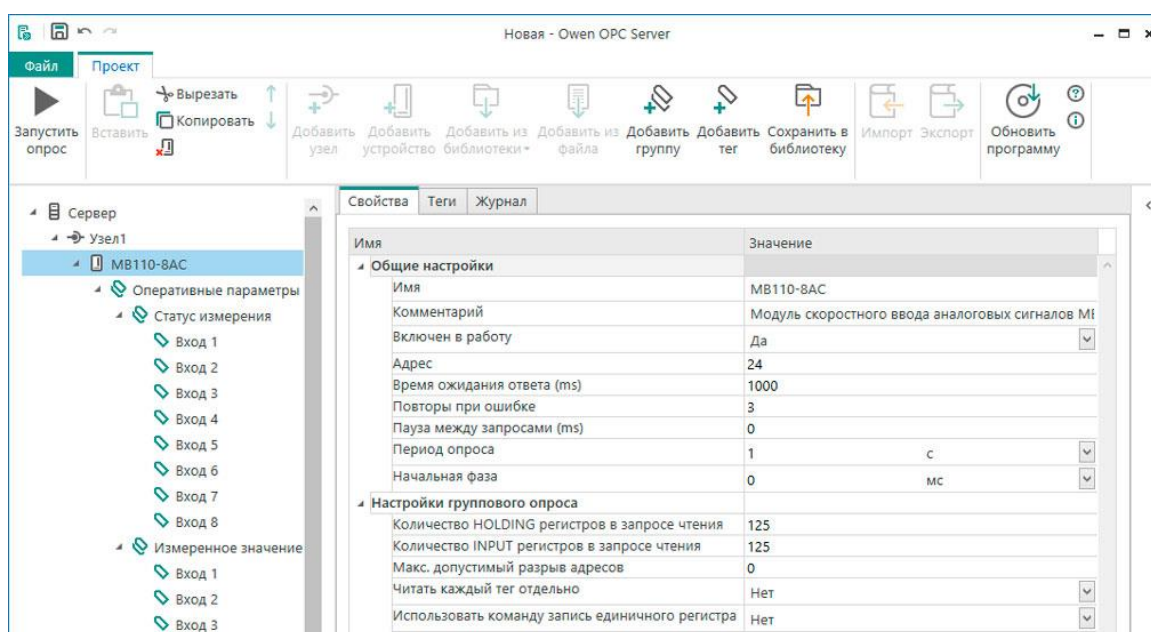


Рисунок 12.1 – Вид программы Owen OPC Server

Основные характеристики Owen OPC Server:

1. Работа по интерфейсу RS-485 (тестировалась с использованием преобразователей интерфейсов ОВЕН АС4 и АС3-М).
2. Поддержка протоколов ОВЕН и функций с 1 по 6,15(0x0F),16(0x10) Modbus RTU/ASCII.
3. Стабильная работа с большим количеством переменных – до 10 000 тегов.
4. Контроль качества связи и удобный лог диагностических сообщений.
5. Возможность просмотра значений переменных в реальном времени.
6. Возможность добавления в OPC-сервер устройств, работающих по протоколам ОВЕН, Modbus RTU/ASCII и Modbus TCP.
7. Возможность создания и сохранения своих шаблонов.

Для удобства работы с приборами с жесткой логикой и свободно программируемыми устройствами Owen OPC Server содержит:

1. Готовые шаблоны – списки параметров для приборов ОВЕН, работающих по протоколам ОВЕН, Modbus RTU/ASCII и Modbus TCP.
 2. Экспорт таблицы сетевых переменных из OwenLogic с использованием плагина для OwenLogic.
 3. Экспорт сетевых переменных конфигурации ПЛК из CODESYS V2.3.
 4. Интеграция с OwenCloud.
- Возможности Owen OPC Server:
1. Связь с приборами по протоколу Modbus RTU/ASCII.
 2. Работа с любым Modbus-устройством.
 3. Поддержка групповых запросов протокола Modbus.

Практическая часть

Ход работы. Настройка OPC-сервера. Для настройки OPC-сервера следует:

1. Открыть программу Owen OPC server.
2. Создать новый проект.
3. Произвести настройку сервера.
4. Добавить узел и сконфигурировать его.
5. Добавить устройство и произвести его настройку
6. Заполнить устройство тегами или группами тегов, исходя из перечня.
7. Произвести настройку тегов: тип доступа – только чтение.
8. Сохранить проект.
9. Включить лабораторный стенд.
10. Открыть ранее созданный проект в среде программирования CoDeSys.
11. Загрузить данный проект в ПЛК.
12. Включить устройства.
13. Произвести опрос OPC-сервера.
14. Снять полученные данные.
15. Остановить опрос сервера.
16. Отключить лабораторные стенды.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Скриншот заполненной структуры OPC-сервера.
4. Скриншот результата работы.
5. Вывод о проделанной работе.

Контрольные вопросы:

1. Owen OPC server. Характеристика, назначение, особенности.
2. Методики формирования OPC-сервера.
3. Методика формирования через опрос устройства.
4. Методика формирования через получение данных из Owen Cloud.
5. Методика формирования через добавление данных программируемого реле.
6. Методика формирования через добавление в OPC-сервер данных ПЛК.

Лабораторная работа 13 «Настройка обмена данных SCADA-системы»

Цель работы: изучение и формирование принципов создания базы данных для хранения полученных результатов.

Теоретическая часть

SCADA (supervisory control and data acquisition, диспетчерское управление и сбор данных) – программный пакет, предназначенный для разработки или обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления. SCADA может являться частью АСУ ТП, АСКУЭ, системы экологического мониторинга, научного эксперимента, автоматизации здания и т. д. SCADA-системы устанавливаются на компьютеры и, для связи с объектом, использует драйверы ввода-вывода или OPC / DDE-серверы (рис. 13.1).

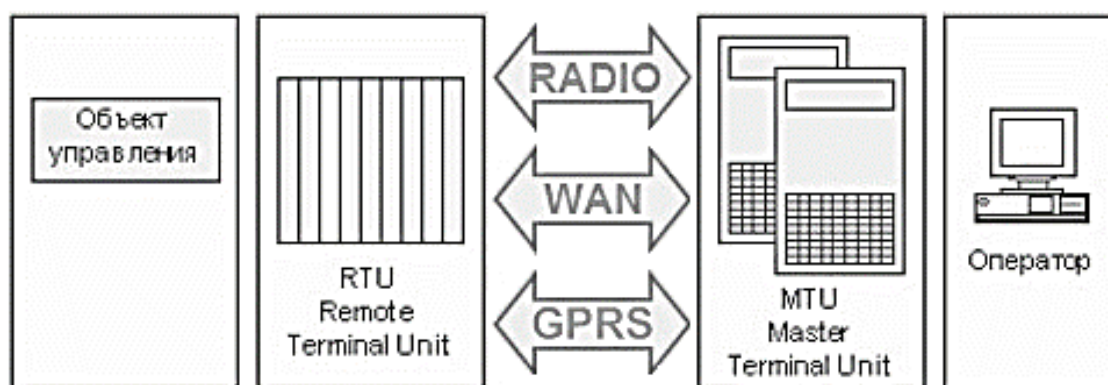


Рисунок 13.1 – Структура SCADA-система

Главная задача SCADA-систем – это сбор информации о множестве удаленных объектов, поступающей с пунктов контроля, и отображение этой информации в едином диспетчерском центре. Также, SCADA-система должна обеспечивать долгосрочное архивирование полученных данных. Диспетчер зачастую обладает возможностью не только пассивно наблюдать за объектом, но и им управлять им, реагируя на различные ситуации.

Задачи SCADA-систем:

- обмен данными с УСО (устройства связи с объектом, то есть с промышленными контроллерами и платами ввода/вывода) в реальном времени через драйверы;
- обработка информации в реальном времени;
- отображение информации на экране монитора в понятной для человека форме;

- ведение базы данных реального времени с технологической информацией;
- аварийная сигнализация и управление тревожными сообщениями;
- подготовка и генерирование отчетов о ходе технологического процесса;
- обеспечение связи с внешними приложениями (СУБД, электронные таблицы, текстовые процессоры и т. д.).

Практическая часть

Ход работы. Настройка обмена данных SCADA-системы. Для настройки обмена данных необходимо:

1. На рабочем столе открыть программу ТЕЛЕМЕХАНИКА ЛАЙТ.
2. Создать новый проект.
3. Перейти во вкладку «Контроллер».
4. Сконфигурировать контроллер по принципу.
5. Сохранить полученную конфигурацию.
6. Перейти во вкладку «История».
7. В дереве проекта добавить новую базу данных.
8. Произвести ее конфигурирование.
9. Заполнить созданную базу данных элементами.
10. Сохранить проект.
11. Подключить лабораторный стенд и устройства.
12. На главной станции программы запустить сервер.
13. Проверить получение данных с устройств.
14. Отключить сервер.
15. Отключить лабораторные стенды.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Скриншот окна «Контроллер».
4. Скриншот окна «История».
5. Скриншот результата опроса переменных.
6. Вывод о проделанной работе.

Контрольные вопросы:

1. Назначение и описание SCADA-систем.
2. Задачи SCADA-систем.
3. Структура SCADA-систем.
4. Особенности создания баз данных.
5. Принцип получения данных с сервера.

Лабораторная работа 14

«Настройка интерфейса SCADA-системы»

Цель работы: изучение методики формирования и настройки интерфейса SCADA-системы.

Теоретическая часть

SCADA (supervisory control and data acquisition, диспетчерское управление и сбор данных) - программный пакет, предназначенный для разработки или обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления (рисунок 14.1).

SCADA-система обычно содержит следующие подсистемы:

1. Драйверы или серверы ввода-вывода.
2. Система реального времени.
3. Человеко-машинный интерфейс (HMI, англ. Human Machine Interface).
4. Система логического управления.
5. База данных реального времени.
6. Система управления тревогами.
7. Генератор отчетов.
8. Внешние интерфейсы.

Особенности процесса управления в SCADA-системах:

1. В системах SCADA обязательно наличие человека (оператора, диспетчера).
2. Любое неправильное воздействие может привести к отказу объекта управления или даже катастрофическим последствиям.
3. Диспетчер несет, как правило, общую ответственность за управление системой, которая, при нормальных условиях, только изредка требует подстройки параметров для достижения оптимального функционирования.
4. Большую часть времени диспетчер пассивно наблюдает за отображаемой информацией, а активное участие диспетчера в процессе управления происходит нечасто, обычно в случае наступления критических событий.
5. Действия оператора в критических ситуациях могут быть жестко ограничены по времени (несколькими минутами или даже секундами).

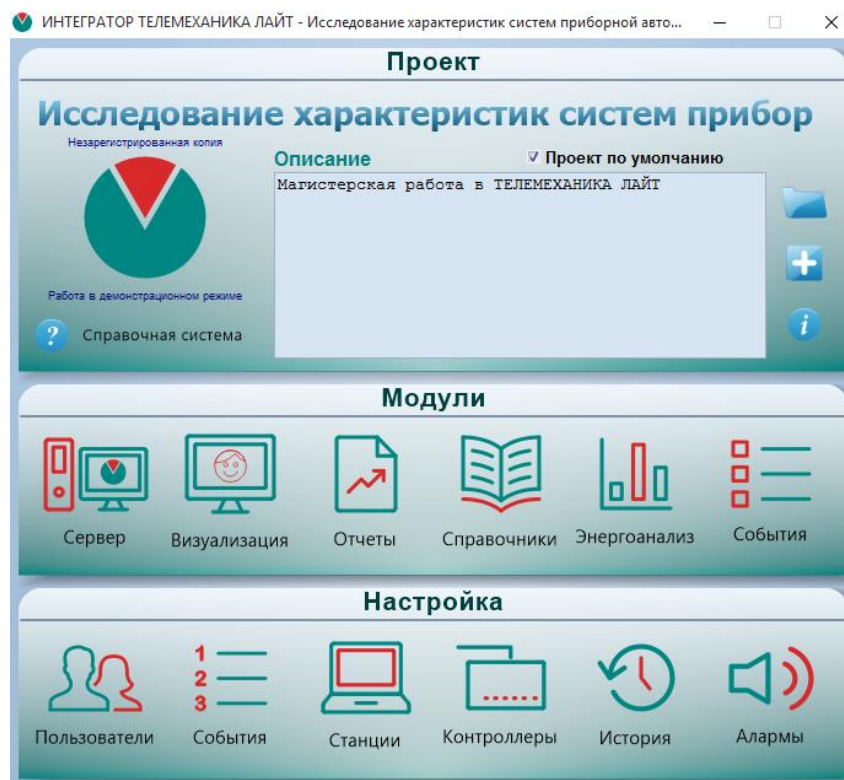


Рисунок 14.1 – SCADA-система ОВЕН «ТЕЛЕМЕХАНИКА ЛАЙТ»

Практическая часть

Ход работы. Настройка интерфейса SCADA-системы. Для настройки интерфейса следует:

1. Запустить программу ТЕЛЕМЕХАНИКА ЛАЙТ.
2. Открыть ранее созданный проект.
3. Перейти во вкладку «Визуализация».
4. Настроить параметры мнемосхемы.
5. Произвести заполнение мнемосхемы элементной базой: кнопки, цифровые табло, тренды, надписи.
6. Настроить кнопку, цифровые табло.
7. Настроить тренды.
8. Сохранить проект.
9. Включить в работу лабораторные стенды.
10. Загрузить в ПЛК ранее созданный проект.
11. Запустить сервер ТЕЛЕМЕХАНИКА ЛАЙТ.
12. Запустить виртуальный контроллер.
13. Запустить визуализацию.
14. Снять температурные зависимости ТРМ202 и ТРМ210.
15. Отключить лабораторные стенды.

Требования к содержанию отчёта:

1. Название лабораторной работы.
2. Цель работы.
3. Скриншот окна «Контроллер».
4. Скриншот окна «Визуализация».
5. Скриншот температурных зависимостей TPM202 и TPM210.
6. Вывод о проделанной работе.

Контрольные вопросы:

- 1) Назначение и описание SCADA-систем.
- 2) Структура SCADA-систем.
- 3) Основные компоненты SCADA-системы.
- 4) Особенности процесса управления в SCADA-системах.
- 5) Web-SCADA.
- 6) Принципы формирования пользовательского интерфейса.

Литература

- 1) Дипломное проектирование : учебно-методическое пособие для студентов учреждений высшего образования по специальности 6-05-0713-04 "Автоматизация технологических процессов и производств" (Компьютерная мехатроника) / сост. : А. А. Кузнецов [и др.]. ; УО "ВГТУ". – Витебск, 2026. – 152 с.
- 2) Петров, И. В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования / И. В. Петров ; под ред. В. П. Дьяконова. – Москва : СОЛОН-Пресс, 2004. – 256 с.
- 3) Промышленные сети передачи данных / Р. К. Нургалиев, Р. Н. Зарипов, Д. Б. Флакс [и др.] // Вестник Казанского технологического университета. – 2013. – Т. 16, № 1. – С. 252–255.
- 4) Пьявченко, Т. А. Проектирование АСУТП в SCADA-системе / Т. А. Пьявченко. – Таганрог : ТТИ ЮФУ, 2007. – 84 с.
- 5) Автоматизация технологических процессов отрасли : методические указания / сост. : К. Н. Ринейский [и др.]. – Витебск : УО «ВГТУ», 2024. – 70 с.

Учебное издание

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ УПРАВЛЕНИЯ В ЭНЕРГЕТИКЕ

Методические указания по выполнению лабораторных работ

Составители:

Кузнецов Андрей Александрович
Ринейский Константин Николаевич
Самусев Артем Михайлович

Редактор *Р.А. Никифорова*
Корректор *А.С. Прокопюк*
Компьютерная верстка *А.М. Самусев*

Подписано к печати 23.06.2026. Усл. печ. листов 4,1.
Уч.-изд. листов 4,9. Заказ № 173.

Учреждение образования «Витебский государственный технологический университет»
210038, г. Витебск, Московский пр., 72.

Отпечатано на ризографе учреждения образования

«Витебский государственный технологический университет».

Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 1/172 от 12 февраля 2014 г.

Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 3/1497 от 30 мая 2017 г.