

АО «Айсорс»



айсорс®

Решения
в промышленном
масштабе

VC SYSTEM

РУКОВОДСТВО РАЗРАБОТЧИКА VCSTUDIO

VC024SA

РЕВИЗИЯ В

2025

Содержание

| | |
|--|-----------|
| 1. ИСТОРИЯ ИЗМЕНЕНИЯ ДОКУМЕНТА | 4 |
| 2. ВВЕДЕНИЕ | 5 |
| 3. ИНСТАЛЛЯЦИЯ ПРОГРАММЫ | 6 |
| 3.1. Требования к программному и аппаратному обеспечению для VCStudio | 6 |
| 3.2. Требования к программному и аппаратному обеспечению для VCont в режиме эмулятора | 6 |
| 3.3. Требования к программному и аппаратному обеспечению для VCont в операционном режиме | 6 |
| 3.4. Способы получения дистрибутивов программ | 7 |
| 3.5. Установка программ на ОС Linux | 7 |
| 3.6. Установка программ на ОС Windows | 7 |
| 4. ЗАПУСК ПРОГРАММЫ VCSTUDIO | 8 |
| 4.1. Запуск программы на ОС Linux | 8 |
| 4.2. Запуск программы на ОС Windows | 8 |
| 5. РАБОЧАЯ ОБЛАСТЬ И ПРОЕКТ В VCSTUDIO | 9 |
| 5.1. Рабочая область и проект | 9 |
| 5.2. Создание новой рабочей области | 9 |
| 5.3. Открытие существующей рабочей области | 10 |
| 5.4. Архивные файлы рабочей области | 10 |
| 5.4.1. Действия пользователя | 10 |
| 5.4.2. Системные сообщения о работе VCStudio | 10 |
| 5.4.3. Резервные копии проекта | 10 |
| 5.5. Авторизация | 10 |
| 5.6. Создание проекта | 11 |
| 5.7. Области интерфейса программы VCStudio | 12 |
| 5.7.1. Сброс расположения окон | 12 |
| 5.7.2. Панель инструментов | 13 |
| 5.7.3. Объекты в окне навигации | 14 |
| 5.8. Настройка интерфейса VCStudio | 15 |
| 5.8.1. Периодичность сохранения рабочей области | 15 |
| 5.8.2. Настройка «горячих» клавиш | 15 |
| 5.8.3. Локальная история | 17 |
| 5.8.4. VCStudio | 17 |
| 6. РАЗРАБОТКА ПРОЕКТА | 19 |
| 6.1. Создание Устройства | 19 |
| 6.2. Создание Ресурса | 20 |
| 6.3. Создание Приложения | 22 |
| 6.4. Создание Контура управления | 23 |
| 6.5. Создание Задач | 24 |
| 6.6. Создание алгоритмов в контуре периодической задачи | 26 |
| 6.6.1. Добавление функциональных блоков | 27 |
| 6.6.2. Внешний вид функционального блока | 27 |
| 6.6.3. Свойства параметров | 28 |
| 6.6.4. Создание соединений внутри одного контура | 28 |
| 6.6.5. Установка значений по умолчанию | 28 |
| 6.6.6. Инициализация типов для ANY переменных | 29 |
| 6.6.7. Блоки с изменяемым количеством входов/выходов | 32 |
| 6.6.8. Порядок выполнения функциональных блоков | 35 |
| 6.6.9. Замена типа блока на другой в существующем контуре | 36 |
| 6.6.1. Команда «Обновить тип блока» | 37 |
| 6.7. Создание алгоритмов в контуре событийной задачи | 38 |
| 6.7.1. Добавление событийных связей | 38 |
| 6.7.2. Добавление генератора событий | 38 |
| 6.7.3. Разветвление и замыкание событий | 39 |
| 6.8. Создание соединений между контурами | 40 |
| 7. ОТЛАДКА ПРОЕКТА | 43 |

| | | |
|------------|--|-----------|
| 7.1. | Запуск среды исполнения (vcont)..... | 43 |
| 7.2. | Он-лайн команды | 43 |
| 7.2.1. | Загрузка проекта | 43 |
| 7.2.2. | Сохранение базы загрузки | 44 |
| 7.2.3. | Инициализация базы загрузки | 45 |
| 7.2.4. | Сброс ресурса..... | 46 |
| 7.2.5. | Перезагрузка | 47 |
| 7.3. | Режим мониторинга | 48 |
| 7.3.1. | Включение мониторинга..... | 48 |
| 7.3.2. | Добавление параметров в мониторинг..... | 49 |
| 7.3.3. | Изменение входных параметров в он-лайн режиме..... | 50 |
| 7.3.4. | Форсирование переменных..... | 50 |
| 7.3.5. | Принудительная генерация событий..... | 52 |
| 8. | НАСТРОЙКА КОНТРОЛЛЕРА | 54 |
| 8.1. | Загрузочный файл | 54 |
| 9. | ОБСЛУЖИВАНИЕ..... | 56 |
| 9.1. | Восстановление проекта из резервной копии | 56 |
| 10. | КОММУНИКАЦИОННЫЕ ВОЗМОЖНОСТИ..... | 58 |
| 10.1. | Modbus TCP Server..... | 58 |
| 10.1.1. | Конфигурирование Modbus Server | 58 |
| 10.1.2. | Привязка параметров к регистрам Modbus TCP сервера..... | 59 |
| 10.1.1. | Правила чтения/записи регистров входных параметров | 60 |
| 10.2. | Modbus TCP Client..... | 60 |
| 10.2.1. | Видимость параметров данных | 61 |
| 10.2.2. | Инициализация типа ANY параметров | 61 |
| 10.2.3. | Блок MBWRITE | 62 |
| 10.2.4. | Блок MBREAD..... | 64 |
| 10.3. | OPC UA Client | 65 |
| 10.3.1. | Блок CLIENT..... | 65 |
| 10.3.2. | Блок SUBSCRIBE | 67 |
| 10.4. | OPC UA Server..... | 69 |
| 10.4.1. | Блок PUBLISH | 69 |
| 10.4.2. | Блок SUBSCRIBE | 70 |

1. История изменения документа

| Ревизия | Дата | Описание изменений |
|---------|------------|-------------------------|
| A | 20.10.2024 | Первая ревизия |
| B | 25.03.2025 | Описание новой иерархии |

2. Введение

«Айсорс – виртуальный контроллер» (VCSysSystem) представляет собой программный комплекс, предназначенный для создания автоматизированных систем управления. Комплекс состоит из исполняемой среды VCont и среды разработки VCStudio.

Исполняемая среда VCont – это программно-реализованный контроллер (Soft PLC), который выполняет функции автоматизации различных процессов. Программно-реализованный контроллер или Виртуальный контроллер работает на базе операционной системы реального времени (OSPV) семейства Linux, что обеспечивает высокую производительность, детерминированность и надежность выполнения задач управления. Виртуальные контроллеры могут быть установлены на различные вычислительные средства: промышленные компьютеры, серверы и т.д. Виртуальные контроллеры опционально могут быть настроены для работы в резервированном режиме, чтобы обеспечить отказоустойчивое управление.

Для конфигурирования системы VCSysSystem используется среда разработки VCStudio. Среда разработки позволяет сконфигурировать системные настройки контроллера, в том числе, коммуникационные возможности. А также создать логические схемы для управления.

Для выполнения функций автоматизированной системы программный комплекс работает совместно с модулями ввода/вывода и другими системами сторонних производителей: платформы для сбора, обработки и оперативного управления.

Данное руководство предназначено для инженеров, разработчиков и специалистов в области автоматизации, которые хотят освоить принципы работы, настройки и программирования Виртуального контроллера. В нем рассмотрены основные концепции, архитектура и ограничения Виртуального контроллера.

3. Установка программы

3.1. Требования к программному и аппаратному обеспечению для VCStudio

Требование к аппаратному обеспечению:

- от 8Гб
- от HDD 500 Gb
- X86 64bit Core i5 и выше

Требование к программному обеспечению:

- Astra Linux 1.7 и выше
- Ubuntu 20.04 и выше
- Debian 12 и выше
- Windows 10 и выше

3.2. Требования к программному и аппаратному обеспечению для VCont в режиме эмулятора

Требование к аппаратному обеспечению:

- от 2Гб
- от HDD 500 Gb
- X86 64bit Core i5 и выше

Требование к программному обеспечению:

- Astra Linux 1.7 и выше
- Ubuntu 20.04 и выше
- Debian 12 и выше
- Windows 10 и выше

3.3. Требования к программному и аппаратному обеспечению для VCont в операционном режиме

- Операционная система на базе ОС Linux с настройками для работы в детерминированном режиме.

Отладку и настройку операционной системы для работы vcont в режиме реального времени осуществляют только сервисные инженеры компании Айсорс

3.4. Способы получения дистрибутивов программ

Получить демо-версию программ можно через запрос на почту pasupport@isource.ru. Дистрибутивы программ поставляются в виде архива или образа виртуальной машины.

3.5. Инсталляция программ на ОС Linux

Выполните следующие шаги, чтобы установить VCStudio:

1. Создайте папку /opt/vcstudio/
2. Разархивируйте в эту папку архив с программой

Выполните следующие шаги, чтобы установить VCont:

1. Создайте папку /opt/vcont/
2. Разархивируйте в эту папку архив с программой

3.6. Инсталляция программ на ОС Windows

Выполните следующие шаги, чтобы установить VCStudio:

1. Создайте папку в корне диска, назовите её “vcstudio”

Обратите внимание, что папка с программой должна обязательно находиться в корне диска. Иначе могут возникнуть ошибки при распаковке файлов в связи с длиной пути.

2. Разархивируйте в эту папку архив с программой

Выполните следующие шаги, чтобы установить VCont:

1. Создайте папку в корне диска, назовите её “vcont”
2. Разархивируйте в эту папку архив с программой

4. Запуск программы VCStudio

4.1. Запуск программы на ОС Linux

Для запуска программы на ОС Linux выполните следующие шаги:

1. Откройте директорию с программой
2. Запустите файл vcstudio

4.2. Запуск программы на ОС Windows

Для запуска программы на ОС Windows выполните следующие шаги:

3. Откройте директорию с программой
4. Запустите файл vcstudio.exe

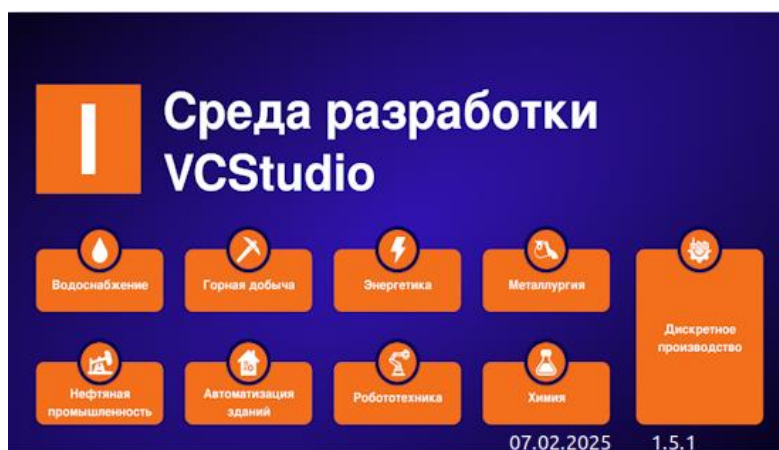


Рисунок 1 – Запуск программы

5. Рабочая область и проект в VCStudio

5.1. Рабочая область и проект

Рабочая область – это данные, которые содержат настройки окружения среды разработки, историю изменений проекта, логи действий пользователя в среде разработки и логи работы программы.

Проект – это совокупность устройств, ресурсов и логических схем управления, имеющих общее адресное пространство.

Директория рабочей области содержит директорию проекта.

На этапе открытия программы выбирается директория рабочей области.

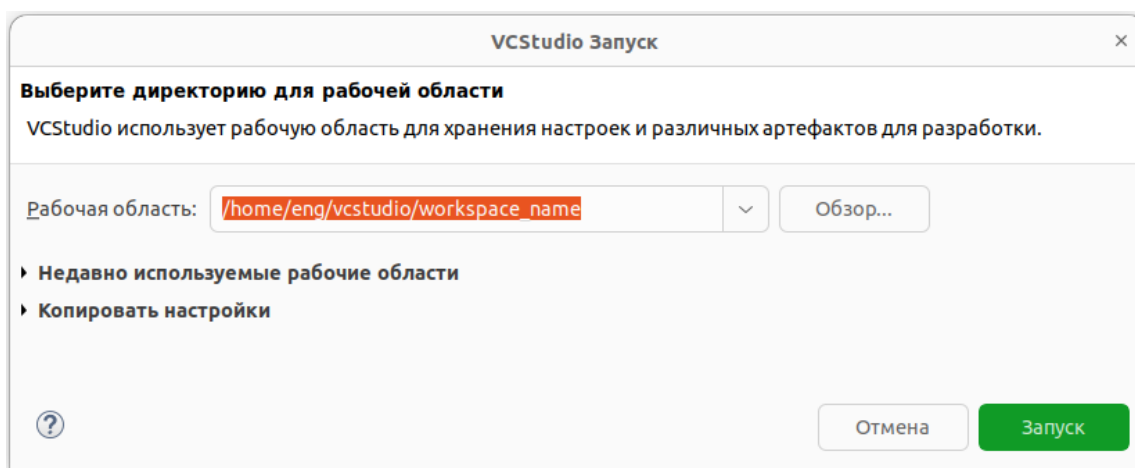


Рисунок 2 – Выбор рабочей области

5.2. Создание новой рабочей области

Для создания новой рабочей области выполните следующие шаги:

1. Создайте директорию с именем проекта.

Имя может содержать латиницу, цифры, нижнее подчеркивание и цифры.
После создания рабочей области имя директории рабочей области и имя проекта изменять нельзя

2. Запустите VCStudio
3. В диалоговом окне **VCStudio Запуск** через кнопку обзор нужно выбрать созданную директорию.

Директорию можно не создавать заранее, а прописать в диалоговом окне вручную название новой директории

4. Нажмите **Запуск** для открытия программы VCStudio. В момент открытия VCStudio создаёт структуру рабочей области и копирует библиотеку функциональных блоков из системной директории.

5.3. Открытие существующей рабочей области

Для открытия существующей рабочей области выполните следующие шаги:

1. Запустите VCStudio
2. В диалоговом окне **VCStudio Запуск** через кнопку **Обзор** выберите директорию рабочей области.

Если выбрать неверную директорию, то VCStudio создаст в новую рабочую область. Рекомендуется называть рабочие области таким образом, чтобы можно было распознать эту корневую папку.

3. Нажмите **Запуск** для открытия программы VCStudio

5.4. Архивные файлы рабочей области

Следующие архивные файлы не очищаются автоматически, требуется периодическое обслуживание через их удаление

5.4.1. Действия пользователя

Действия пользователя записываются в файл `/workspace_name/log.txt`

5.4.2. Системные сообщения о работе VCStudio

Системные сообщения о работе VCStudio записываются в файл `/workspace_name/.metadata/.log`

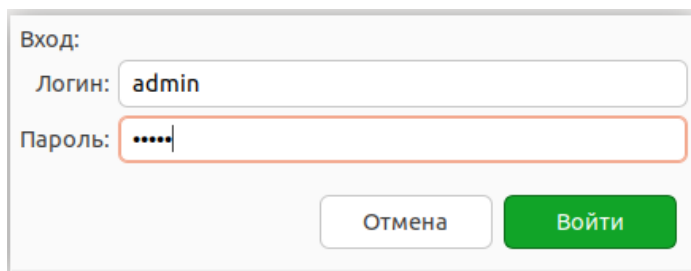
5.4.3. Резервные копии проекта

Резервные копии проекта сохраняются в директории `/workspace_name/.metadata/.plugins/org.eclipse.core.resources/.history/`

5.5. Авторизация

После выбора рабочей области программа откроет окно авторизации.

Логин и пароль по умолчанию: «admin», «admin».



Вход:

Логин: admin

Пароль:

Отмена Войти

5.6. Создание проекта

Для создания проекта выполните следующие шаги:

1. Запустите VCStudio
2. Создайте новое рабочее пространство или откройте существующее рабочее пространство без проекта
3. Выполните авторизацию (по умолчанию, логин и пароль по умолчанию: «admin», «admin»)
4. В строке меню программы выберите Файл – Новый проект

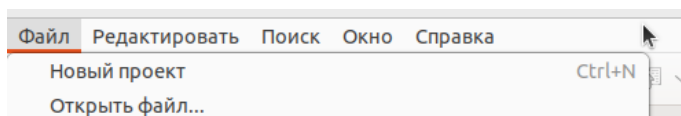
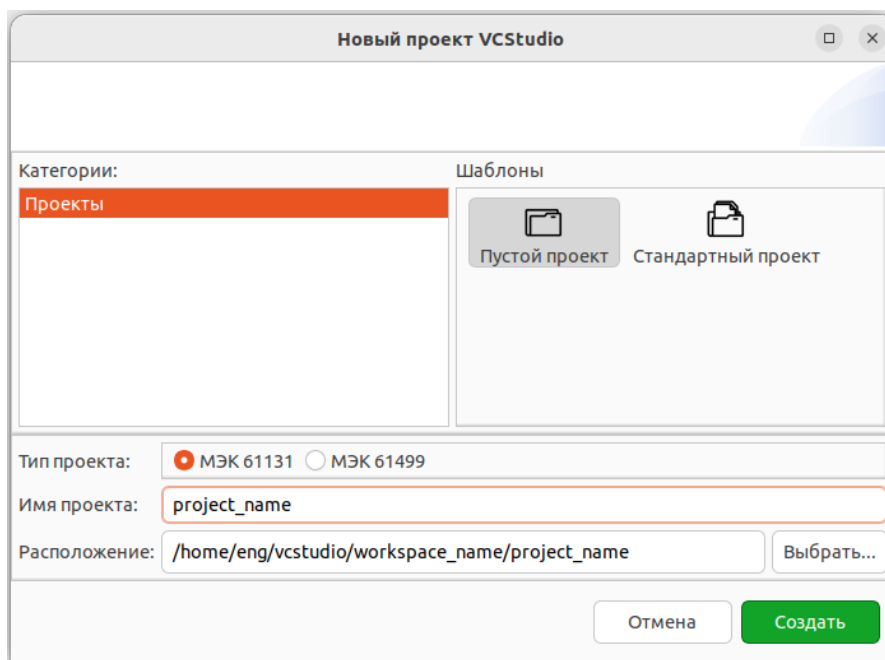


Рисунок 3 – Создание нового проекта

5. В открывшемся диалоговом окне задайте тип проекта, имя проекта и шаблон проекта.



Шаблоны – это заранее созданная структура проекта, которая помогает быстрее и качественнее начать разрабатывать проект.

Тип проекта – тип проекта определяет структуру объектов и тип задач, которые будут использоваться:

- МЭК 61131 – в проекте используются событийные и периодические задачи, структура проекта состоит из Приложений и контуров.
- МЭК 61499 – устаревший тип, используются только событийные задачи, структура проекта состоит из Подприложений (SubApplication).

6. Нажмите кнопку **Создать** для создания проекта в рабочей области.

5.7. Области интерфейса программы VCStudio

Пространство интерфейса VCStudio разделено на 4 области:

1. **Область навигации по проекту** – состоит из двух вкладок:

Структура системы – используется для создания и управления устройствами, ресурсами, приложениями и контурами.

Библиотека – библиотека функциональных блоков, содержит список доступных функциональных блоков.

2. **Редактор** – в этой области открываются редакторы объектов.

3. **Навигация по контуру** – графическая навигация по хосту контура.

4. **Свойства и сообщения** – отображение свойств объектов, вывод сообщений при работе с программой.

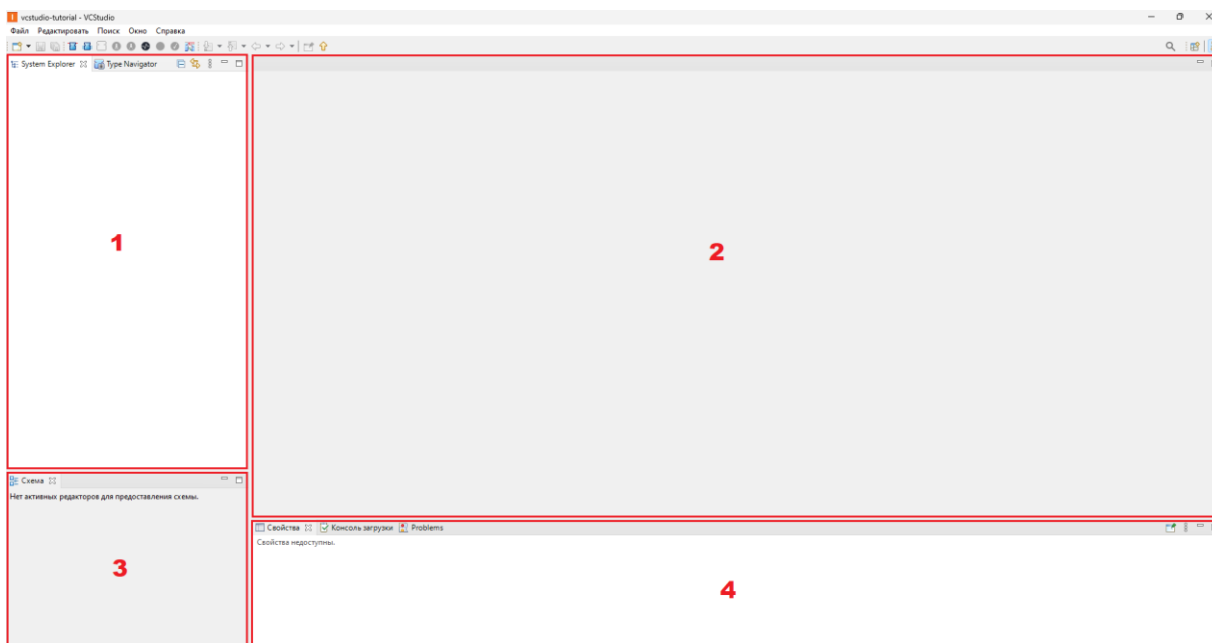


Рисунок 4 – Интерфейс программы

5.7.1. Сброс расположения окон

Для того, чтобы вернуть расположение окон в их положение по умолчанию, нужно выполнить команду на панели быстрого доступа – нажать на кнопку «System» и выполнить команду «Сброс» (см. [Рисунок 5](#), [Рисунок 6](#)). Подтвердить выполнение – нажать «Сбросить перспективу».

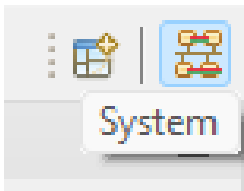


Рисунок 5 – Кнопка «System»

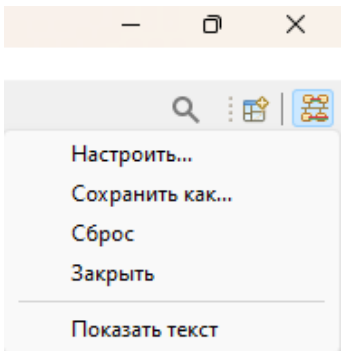


Рисунок 6 – Команда «Сброс»

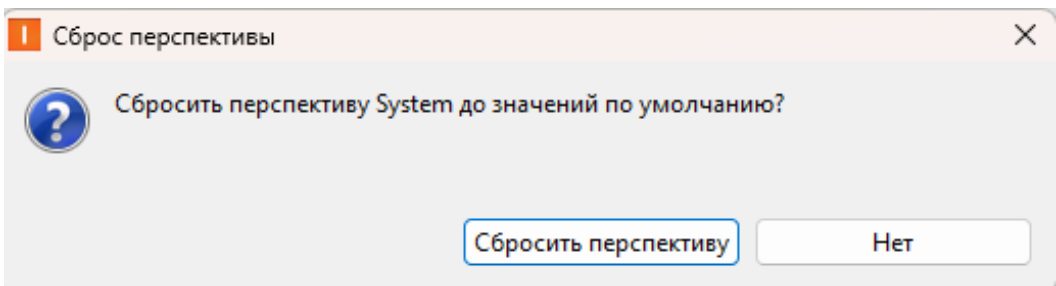


Рисунок 7 – «Сбросить перспективу»

5.7.2. Панель инструментов



Панель инструментов содержит основные часто используемые команды (см. [Рисунок 8](#)).








Рисунок 8 – Панель быстрого доступа

Функционал кнопок указан в следующей таблице (см. [Таблица 1](#)):

Таблица 1 – Элементы панели инструментов






| | |
|---|--|
|  | Скрыть связи событий |
|  | Скрыть подключения для передачи данных |

| | |
|---|---|
|  | Печать приложений и оборудования автоматизации |
|  | Позволяет сохранять изменения из панели инструментов или файла (используется «Сохранить» или «Сохранить все»), альтернативный доступ по нажатию сочетания клавиш Ctrl+S |
|  | обеспечивает отмену и/или повтор последних изменений |
|  | Обеспечивает функцию масштабирования на панели инструментов или пунктах меню в контекстном меню редактора, а также при нажатии Ctrl при прокрутке скроллера «мыши» |
|  | Меню настройки интерфейса, сброс расположения окон |

5.7.3. Объекты в окне навигации

Окно навигации позволяет быстро сориентироваться в проекте, найти, отредактировать и переименовать объект. В следующей таблице представлен список объектов и их описание (см. [Таблица 2](#)).

Таблица 2 – Графические элементы программы

| | | |
|---|--------------------------|--|
|  | Проект | Проект – это пространство, в котором содержится вся конфигурация системы. Проект предполагает единое адресное пространство, в котором соблюдено правило уникальности имен объектов. В одном экземпляре VCStudio может быть открыт только один проект. |
|  | Устройство | Устройство – это сервер, промышленный компьютер, который является хостом для среды исполнения (Ресурса) |
|  | Ресурс | Ресурс, среда исполнения – это приложение, которое выполняет периодические или событийные задачи. Периодические выполняются в режиме жесткого реального времени. |
|  | Приложение | Объект Приложение используется: - для структурирования алгоритмов, например, по технологическим блокам. - для определения порядка выполнения контуров управления |
|  | Контур управления | Контур управления – связный алгоритм для управления одним объектом управления или, если нет управления, группа сигналов для одного объекта |

5.8. Настройка интерфейса VCStudio

Для настройки интерфейса VCStudio необходимо открыть меню Окно -> Параметры (см. [Рисунок 9](#)).

Перейти в основное окно настройки программы – «Параметры. Открыть раздел «Общие» (см. [Рисунок 10](#)).

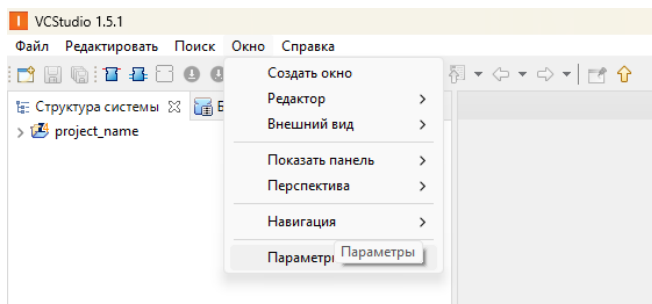


Рисунок 9 – Открытие окна «Параметры»

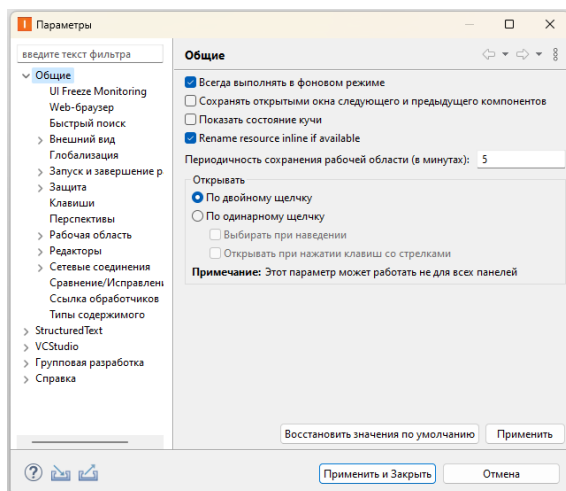


Рисунок 10 – Раздел параметров – «Общие»

5.8.1. Периодичность сохранения рабочей области

Для настройки данного параметра, необходимо перейти в раздел «Общие».

В данной настройке необходимо указать параметр – периодичность сохранения рабочей области (в минутах). Параметр указан – 5 минут (см. [Рисунок 10](#)).

5.8.2. Настройка «горячих» клавиш

В программе есть возможность настройки «горячих» клавиш. Необходимо перейти в настройках по пути «Общие»-«Клавиши» к окну настроек (см. [Рисунок 11](#)). Далее, необходимо в окне «введите текст фильтра»-ввести название программы VCStudio и перейти непосредственно в окно с настройкой «горячих» клавиш (см. [Рисунок 12](#)).

5. Рабочая область и проект в VCStudio

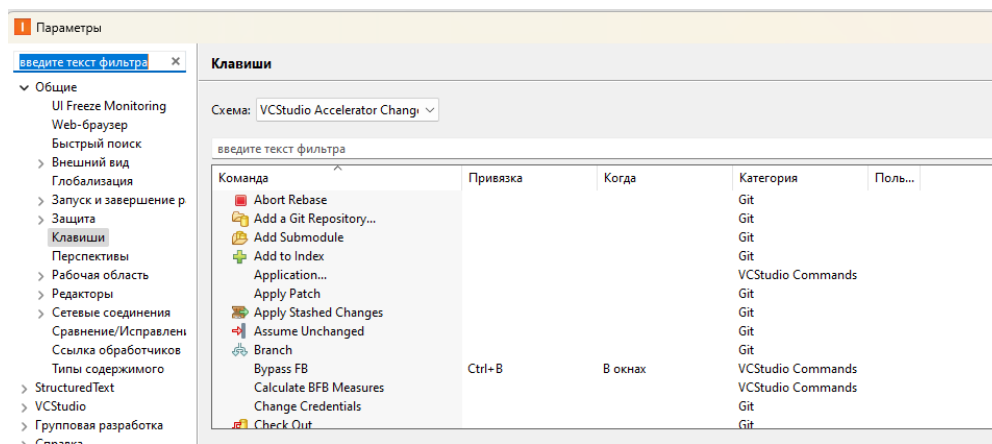


Рисунок 11 – Окно настроек «горячих» клавиш

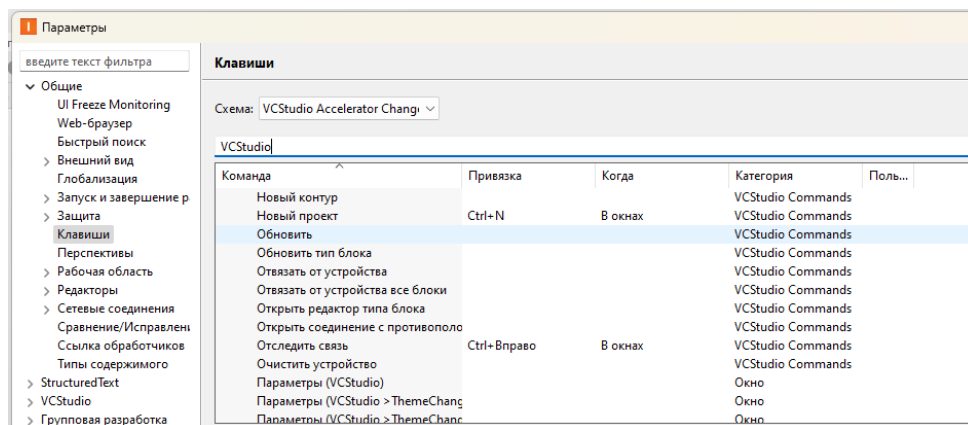


Рисунок 12 – Команды «горячих» клавиш

Для настройки любой команды, например – «Мониторинг», необходимо в области «Команда» нажать ЛКМ (левая кнопка манипулятора типа «мышь») на неё. В окне «Привязка» нажать нужные клавиши, например – «Ctrl+M». Теперь, данные клавиши будут привязаны к команде «Мониторинг» (см. [Рисунок 13](#)). И далее, нажать «Применить и Заккрыть».

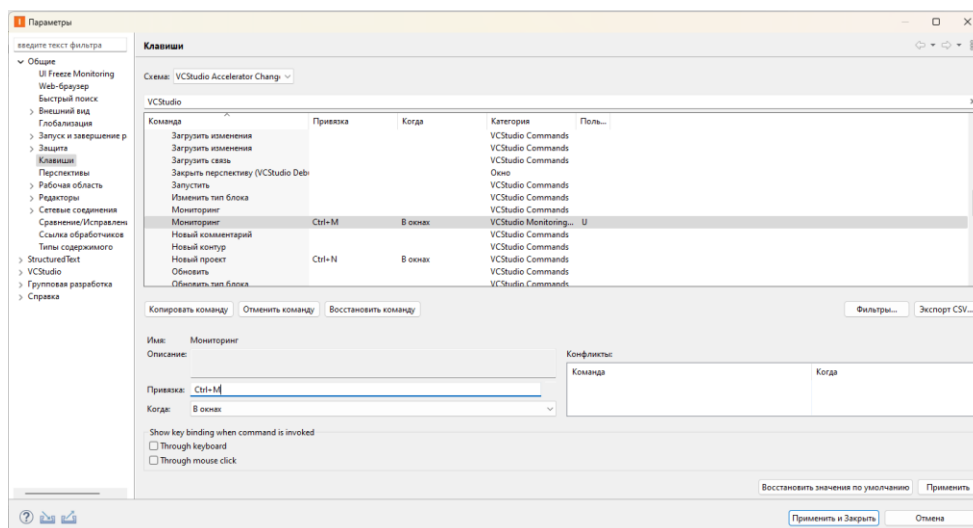


Рисунок 13 – «Горячие» клавиши для команды «Мониторинг»

Таким способом можно настроить «горячие» клавиши для других команд из списка.

5.8.3. Локальная история

Необходимо перейти в настройках по пути «Общие»-«Рабочая область»-«Локальная история» к окну настроек (см. [Рисунок 14](#)).

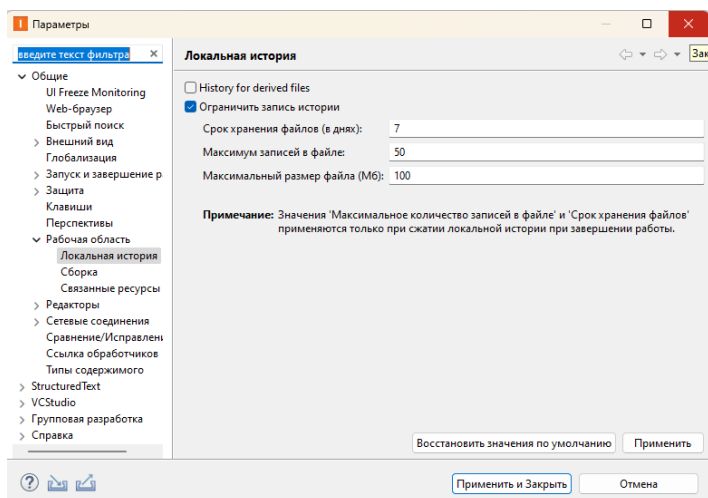


Рисунок 14 – Окно «Локальная история»

Настраивается параметр – «Ограничить запись истории». Каждое сохранение – сохраняет предыдущее состояние. В настройках указывается правило сохранения истории. Чтобы архив не переполнялся, вводятся указанные ограничения (см. [Рисунок 14](#)). После настройки, необходимо нажать «Применить и Закрывать».

5.8.4. VCStudio

Необходимо перейти в настройках по пути «Общие»-«VCStudio» к окну настроек (см. [Рисунок 15](#)). Можно настроить цвета объектов.

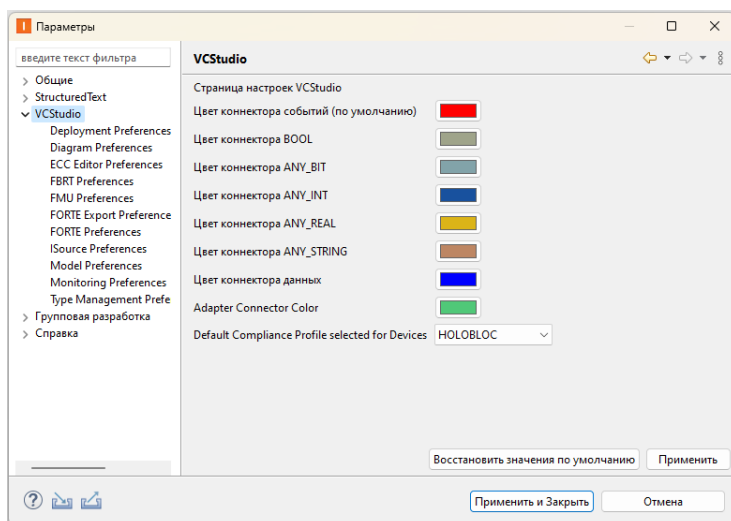


Рисунок 15 – Окно настроек VCStudio

5. Рабочая область и проект в VCStudio

Для Deployment Preferences (Настройки развертывания) увеличиваем время подключения до 10000 мс (см. [Рисунок 16](#)). И нажимаем «Применить и Заккрыть».

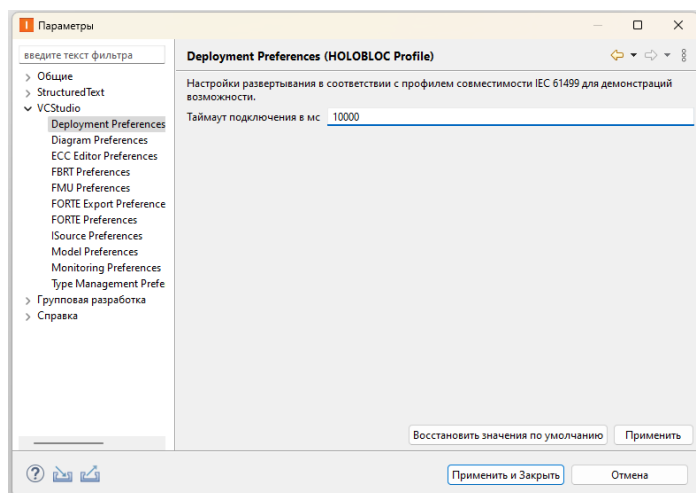


Рисунок 16 – Настройка развертывания

Для Monitoring Preferences (Настройки мониторинга) – опрос при мониторинге (Polling interval in ms) устанавливаем 300 мс (см. [Рисунок 17](#)). По окончании настроек нажимаем «Применить и Заккрыть».

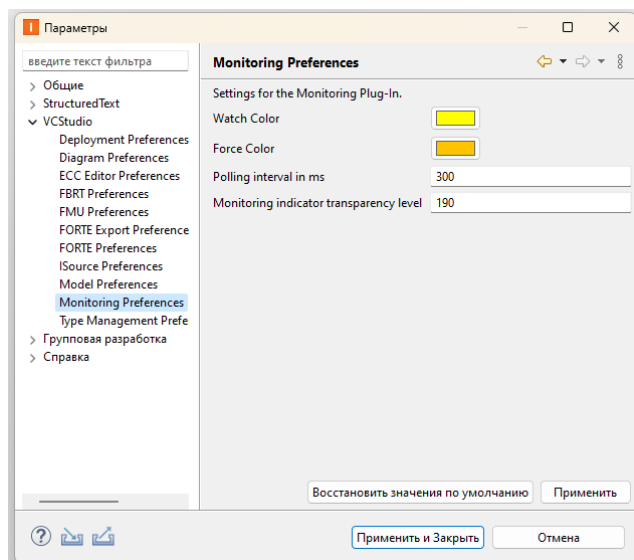


Рисунок 17 – Настройка мониторинга

6. Разработка проекта

6.1. Создание Устройства

Для создания устройства необходимо открыть в левой части окна рабочую область – «Структура системы» (см. [Рисунок 18](#)).

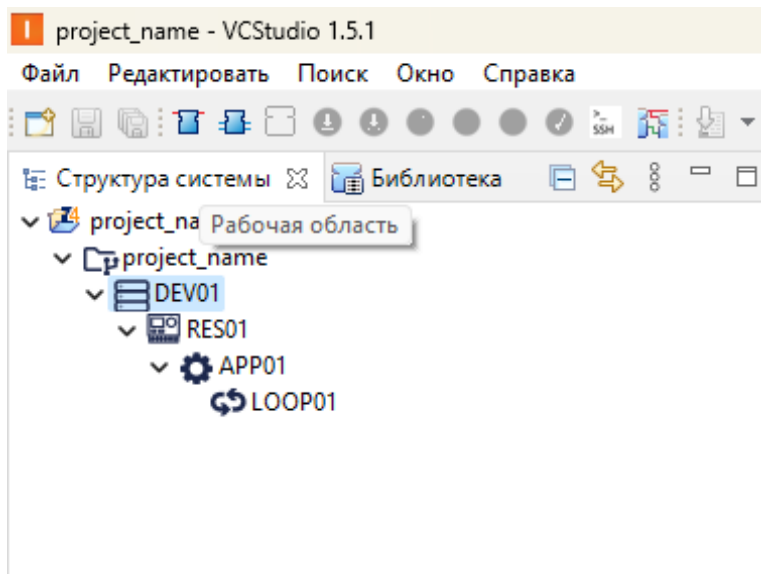


Рисунок 18 – Рабочая область – «Структура системы»

Выбираем команду «Добавить устройство» контекстного меню проекта (см. [Рисунок 19](#)).

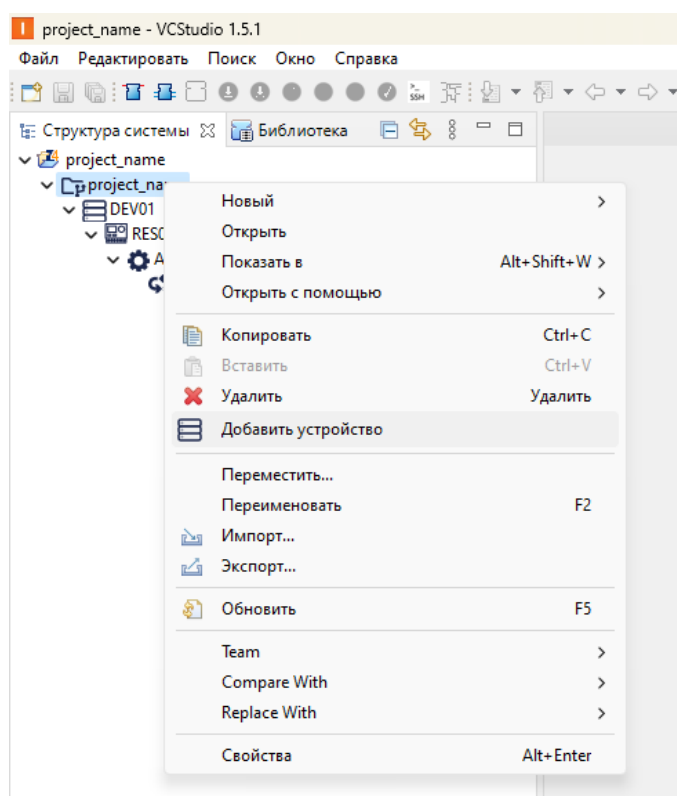


Рисунок 19 – Выбираем команду – «Добавить устройство»

В списке устройств появилось новое устройство – DEV02 (см. [Рисунок 20](#)).

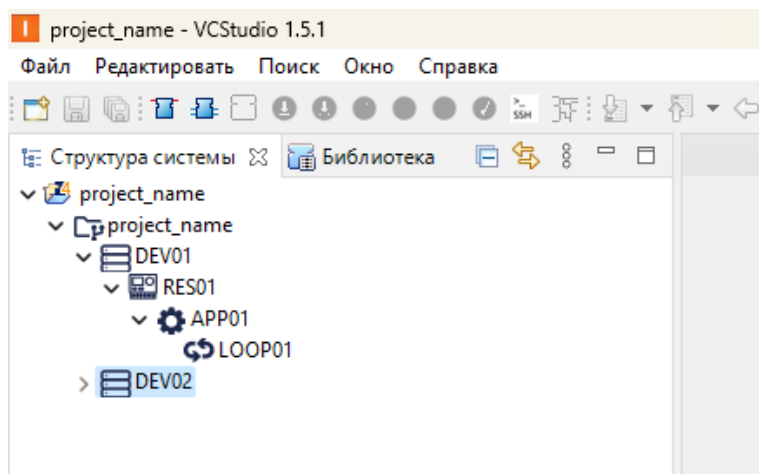


Рисунок 20 – Новое устройство – DEV02

Затем, если необходимо, можно переименовать созданное устройство в нижней части окна в меню «Свойства». Например – DEVICE02 (см. [Рисунок 21](#)).

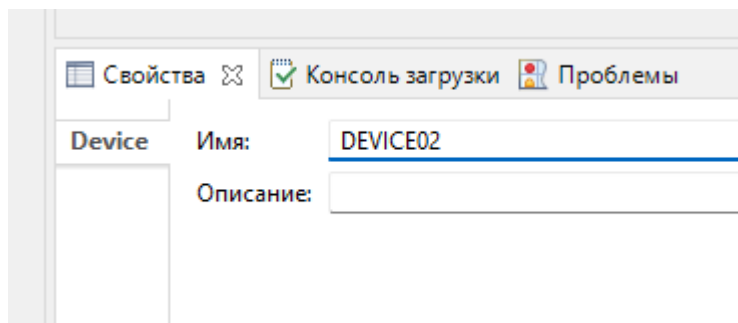


Рисунок 21 – Переименование созданного устройства

6.2. Создание Ресурса

Чтобы создать ресурс, необходимо выбрать команду «Добавить ресурс» в контекстном меню устройства (см. [Рисунок 22](#)).

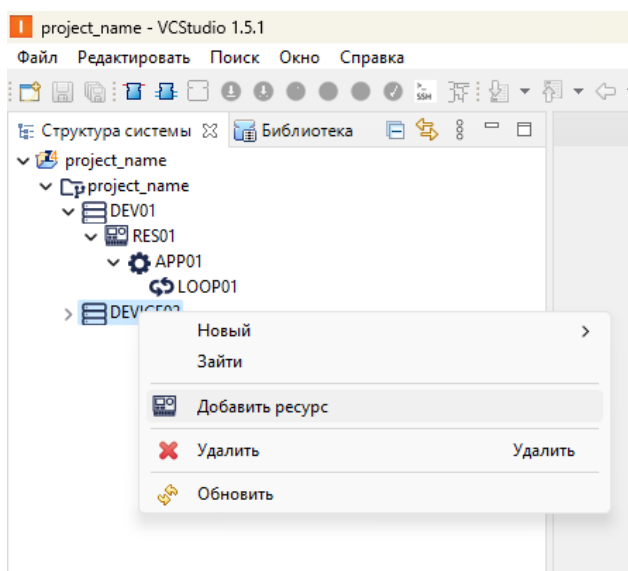


Рисунок 22 – Выбираем команду – «Добавить ресурс»

Под устройством появится ресурс RES01 (см. [Рисунок 23](#)).

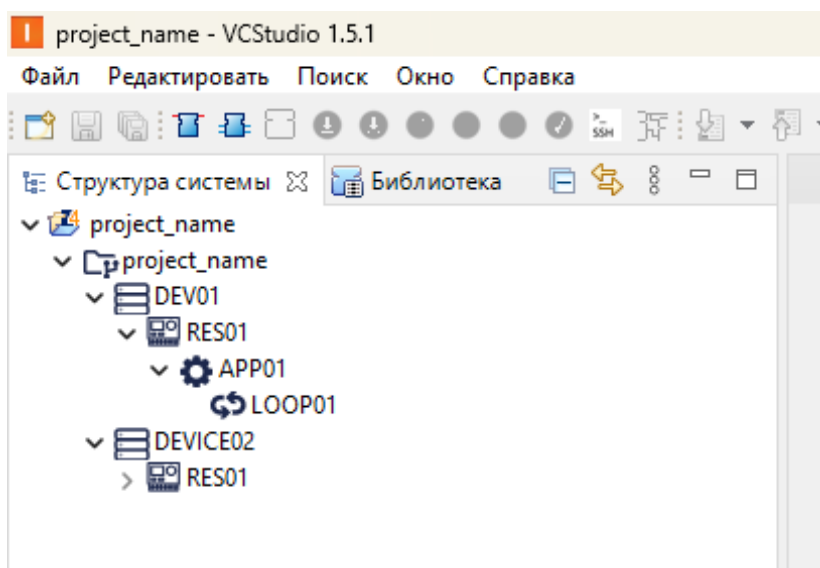


Рисунок 23-Добавление ресурса

Название ресурса можно также переименовать через меню «Свойства» в нижней части окна (см. [Рисунок 24](#)).

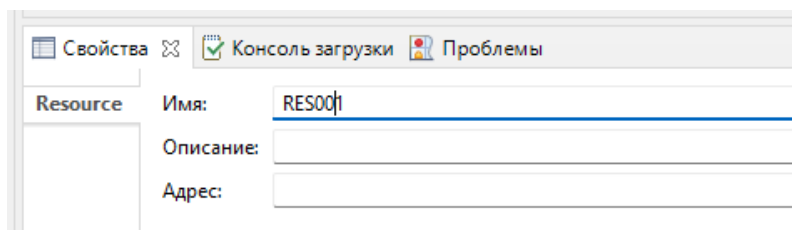


Рисунок 24 – Изменение имени ресурса

Для связи с устройством необходимо добавить IP-адрес устройства и номер порта в окне «Адрес» (см. [Рисунок 25](#)).

6. Разработка проекта

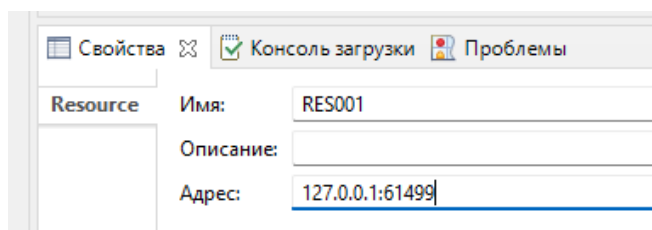


Рисунок 25 – Добавление IP-адреса и номера порта

6.3. Создание Приложения

Для создания приложения ресурса нужно выполнить команду «Добавить приложение» из контекстного меню ресурса (см. [Рисунок 26](#)).

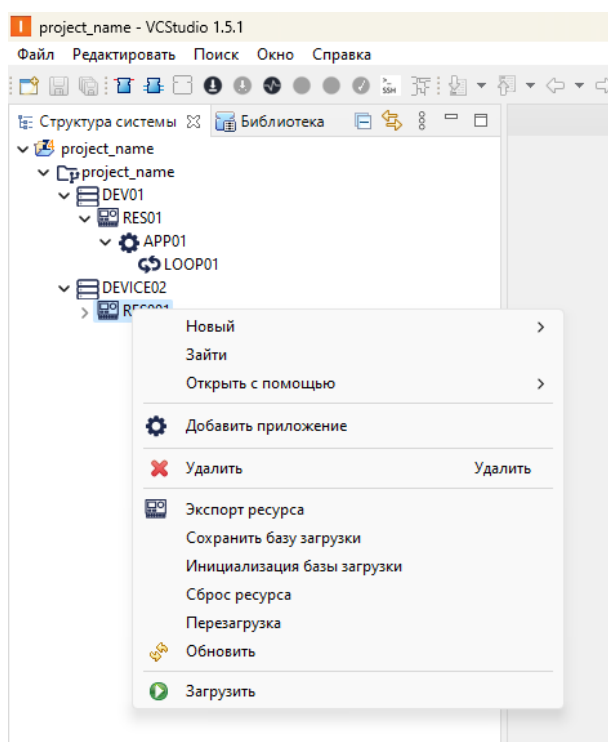


Рисунок 26 – Добавление приложения

Под ресурсом появляется приложение APP01 (см. [Рисунок 27](#)).

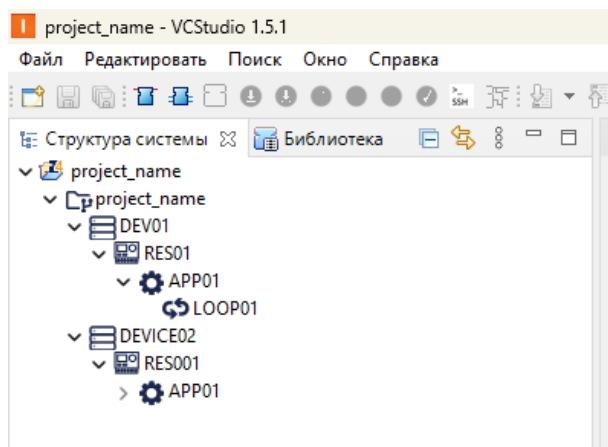


Рисунок 27 – Приложение ресурса

Приложение можно переименовать через меню «Свойства» (см. [Рисунок 28](#)).

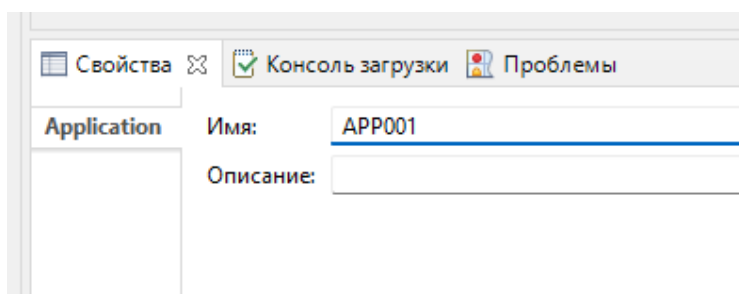


Рисунок 28 – Переименование приложения

6.4. Создание Контура управления

Чтобы добавить контур управления в Приложение, нужно выполнить команду «Добавить контур управления» из контекстного меню приложения (см. [Рисунок 29](#)).

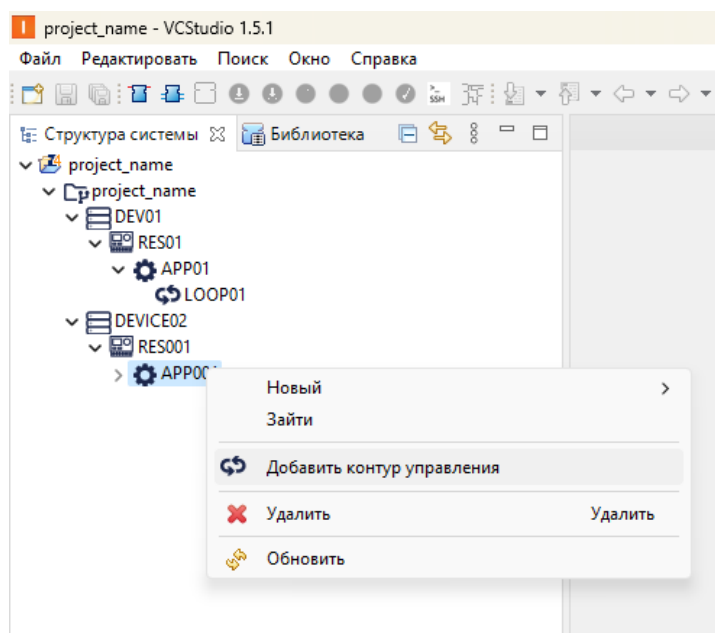


Рисунок 29 – Добавление контура управления

У приложения появляется контур управления LOOP01 (см. [Рисунок 30](#)).

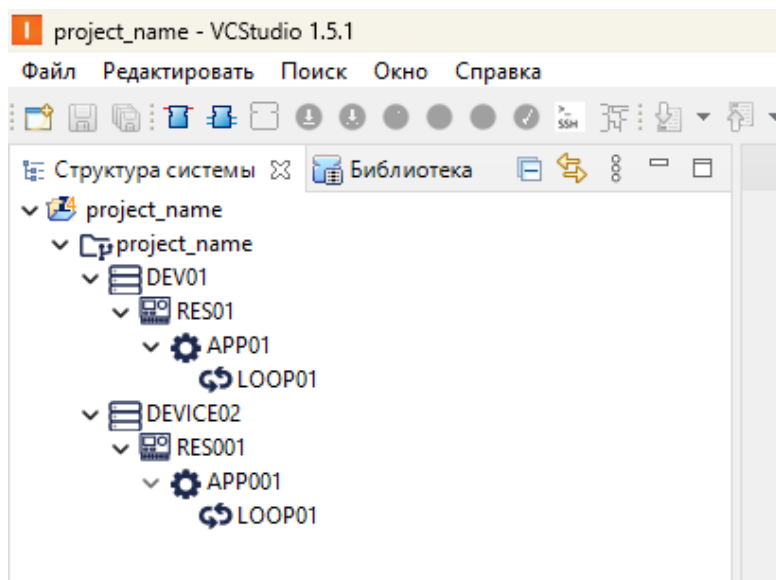


Рисунок 30 – Контур управления приложения

Чтобы переименовать контур управления, нужно выделить контур управления и в его свойствах изменить имя (см. [Рисунок 31](#)).

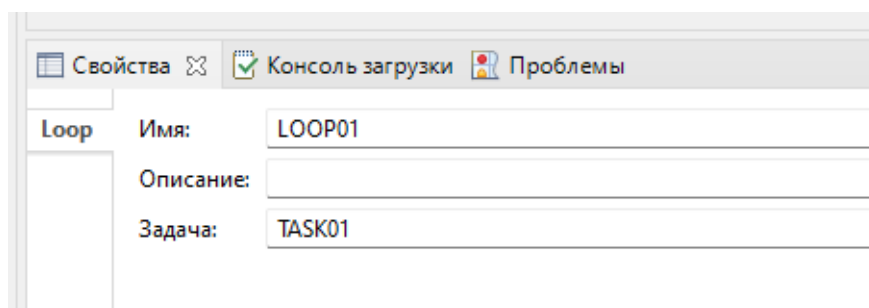


Рисунок 31 – Свойства контура управления

6.5. Создание Задач

Контур управления состоит из функциональных блоков, которые определяют алгоритм для обработки данных.

Функциональный блок может обрабатываться двумя способами:

- в событийной задаче через событие, которое подаётся на событийный вход
- в периодической задаче через его запуск в порядке циклической очереди. Каждый контур должен быть назначен задаче.

Таким образом, существуют следующие типы задач:

- Периодические (функциональные блоки выполняются один раз в заданный период)
- Событийные (блоки выполняются от триггеров событий)

Для создания задачи необходимо ЛКМ два раза нажать на ресурс (см. [Рисунок 32](#)). Откроется окно редактирования ресурса. Необходимо выбрать вкладку

«Задачи». Во вкладке можно редактировать существующие задачи или создавать новые.

Новая задача создается при нажатии ЛКМ на значок  (см. [Рисунок 32](#)).

Возможна смена имени задачи. Для этого, необходимо нажать ЛКМ два раза на задачу. При этом, она выделяется синим цветом (см. [Рисунок 33](#)).

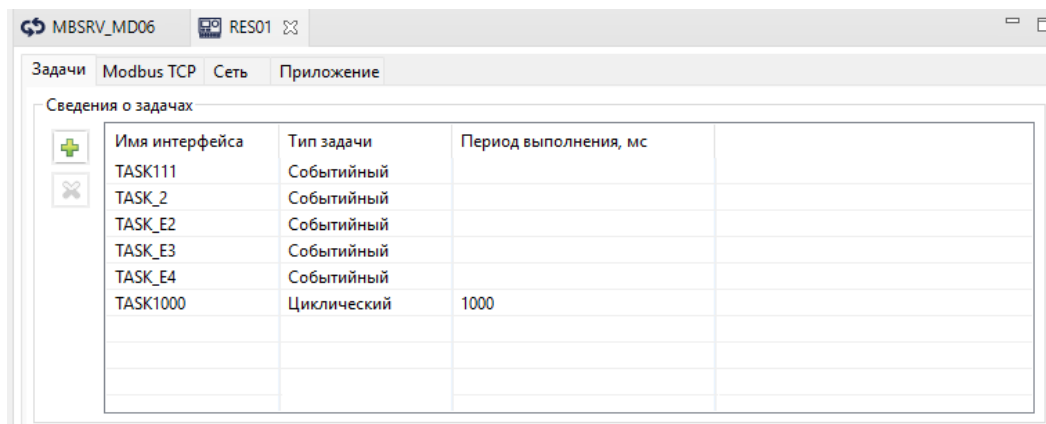


Рисунок 32 – Окно задач

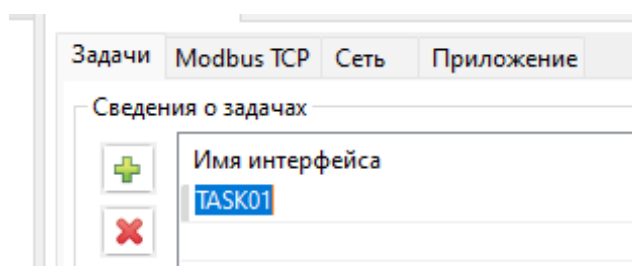


Рисунок 33 – Смена имени задачи

Для циклических задач, необходимо устанавливать период выполнения, который задается также двойным нажатием ЛКМ на цифру периода (см. [Рисунок 34](#)). Произойдет выделение синим цветом значения периода.

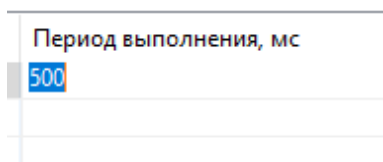


Рисунок 34 – Изменение периода выполнения

Внимание: нельзя назначать одинаковый период выполнения для разных задач!

При удалении периода выполнения в циклической задаче, задача становится – событийной.

Каждому контуру назначается определенная задача. Для этого, необходимо выделить контур и в свойствах указать нужную задачу (см. [Рисунок 35](#)).

6. Разработка проекта

Задача может быть событийной или циклической в зависимости от определения типа в окне «Задачи».

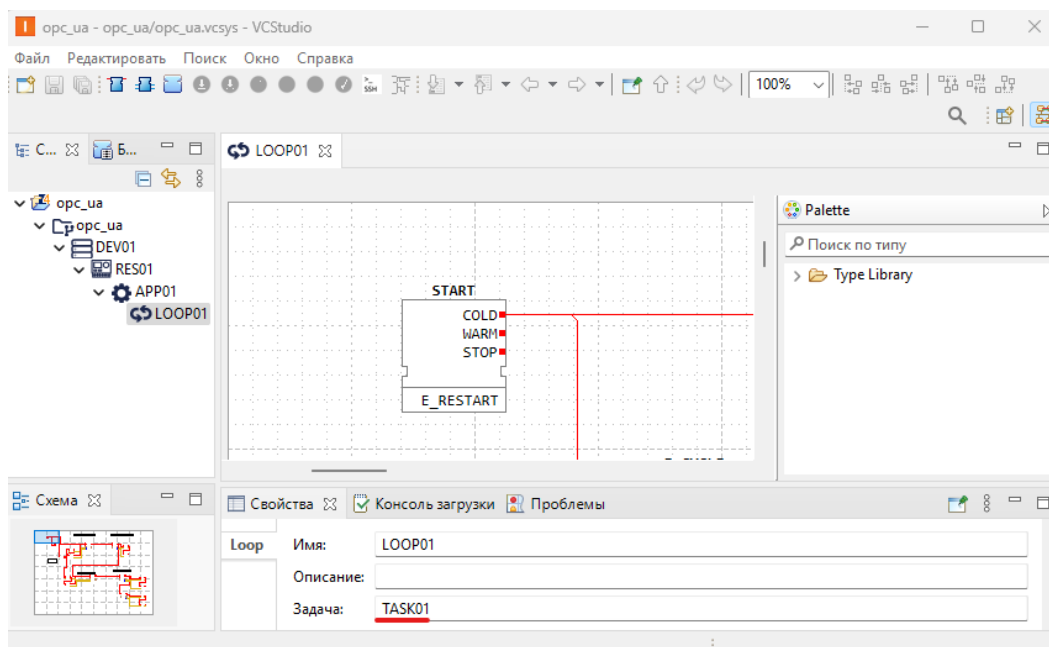


Рисунок 35 – Назначение задачи контуру LOOP01

6.6. Создание алгоритмов в контуре периодической задачи

Для создания алгоритма необходимо открыть редактор контура, нажав два раза на контур ЛКМ. Справа от редактора откроется палитра (Palette) библиотеки функциональных блоков (см. [Рисунок 36](#)).

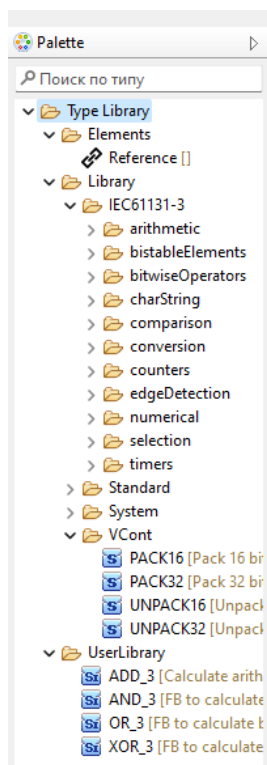


Рисунок 36 – Библиотека ФБ

6.6.1. Добавление функциональных блоков

Библиотека функциональных блоков содержит различные ФБ как стандарта IEC61131-3, так и ФБ программы VCont.

Для создания экземпляра ФБ нужно перенести ФБ из палитры в окно редактора контура.

Для удобного поиска типа блока в палитре в строке поиска можно написать название необходимого ФБ. Например, BOOL2BOOL (см. [Рисунок 37](#)).

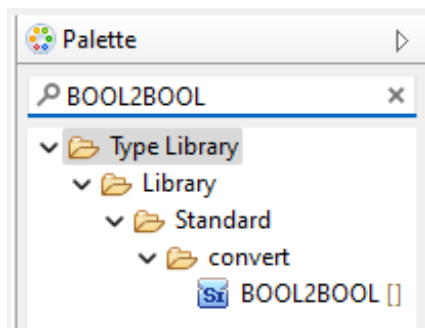


Рисунок 37 – ФБ BOOL2BOOL

После переноса функциональных блоков в окно редактора блоки появляются в редакторе со сгенерированными по умолчанию именами (см. [Рисунок 38](#)).

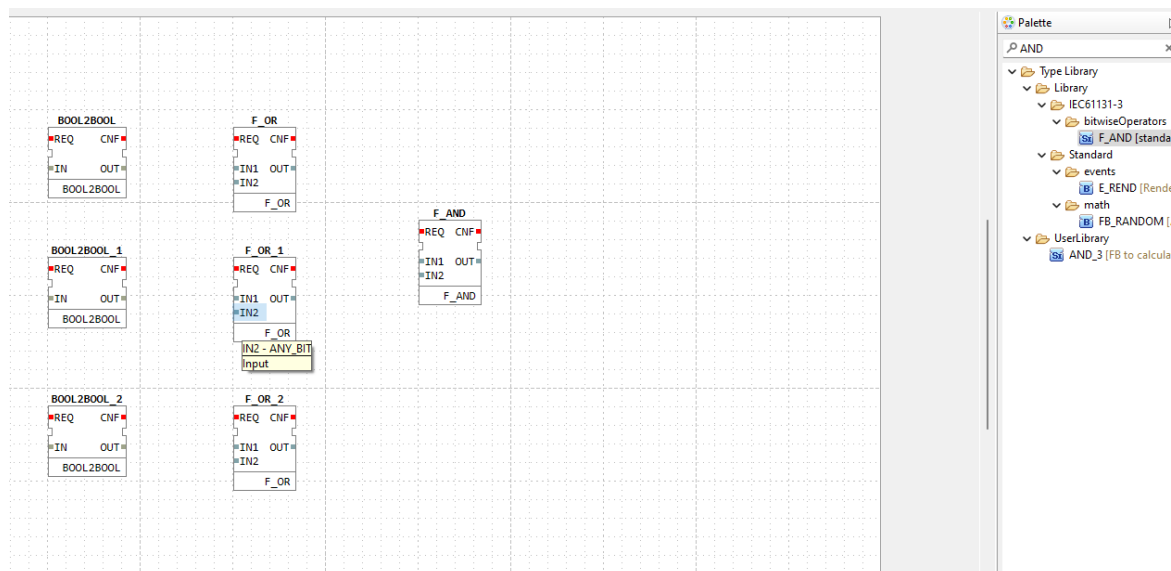


Рисунок 38 – Добавление функциональных блоков

6.6.2. Внешний вид функционального блока

Название экземпляра функционального блока находится первой строкой. Порты ввода находятся слева от блока, порты выхода – справа. Внизу блока указан тип блока. Красным цветом обозначены порты, которые работают только с событийными задачами. При наведении курсора на любой вход или выход, появится его описание.

6.6.3. Свойства параметров

В свойствах параметров можно увидеть тип данных параметра и задать начальное значение (см. [Рисунок 39](#)).

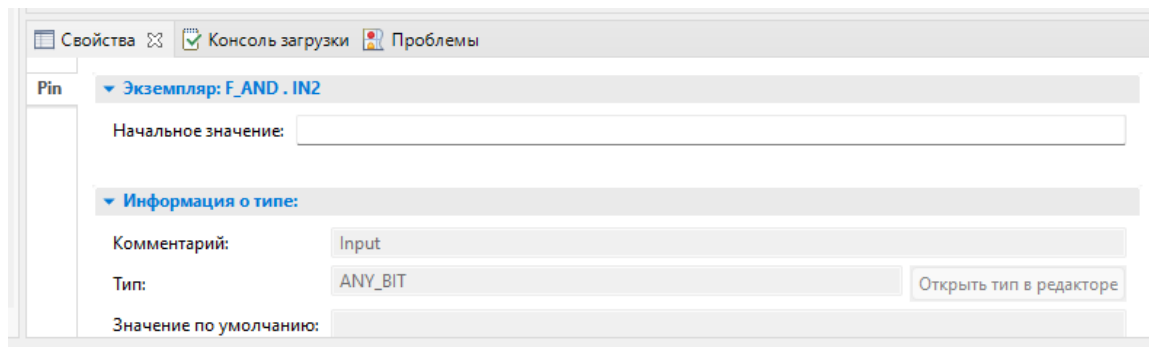


Рисунок 39 – Тип параметра функционального блока

6.6.4. Создание соединений внутри одного контура

Для передачи значений между параметрами блоков используются соединения (связи). Соединения могут быть двух типов:

- Событийные связи (используются только в контурах событийных задач)
- Связи передачи данных

Для добавления связи нужно ЛКМ нажать на источник и указать приёмник, от выхода одного ФБ ко входу другого (см. [Рисунок 40](#)).

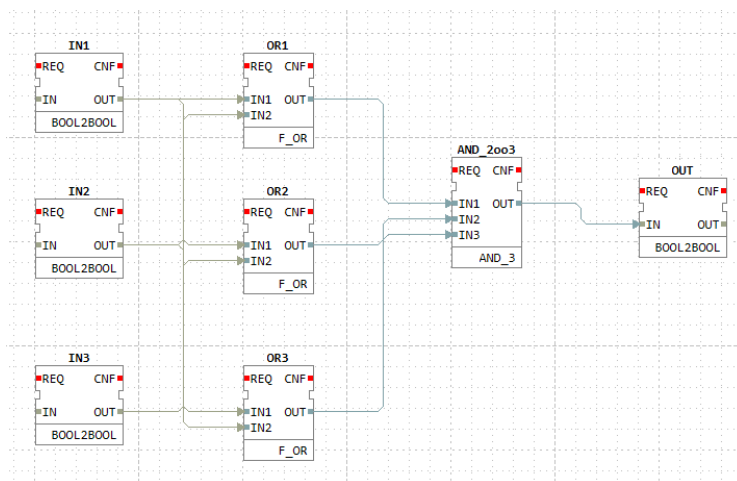


Рисунок 40 – Добавление связей

Из одного выходного параметра может выходить несколько соединений.

6.6.5. Установка значений по умолчанию

Значения параметров блоков по умолчанию не определены, после загрузки в контроллер и запуска задачи входные параметры, которые не подключены, примут значения в соответствии с внутренним алгоритмом. В большинстве случаев: false – для булевых параметров, 0 – для аналоговых параметров.

Установить своё значение по умолчанию можно двумя способами:

- в свойствах блока (см. рисунки [Рисунок 41](#), [Рисунок 42](#)) в поле «Начальное значение»
- через графическое поле слева графического изображения параметра в контуре (см. [Рисунок 43](#))

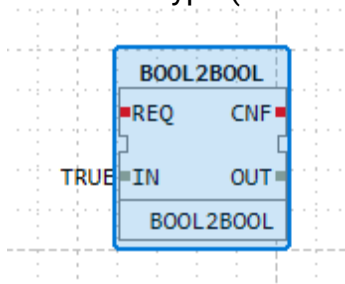


Рисунок 41 – Значение TRUE на входе

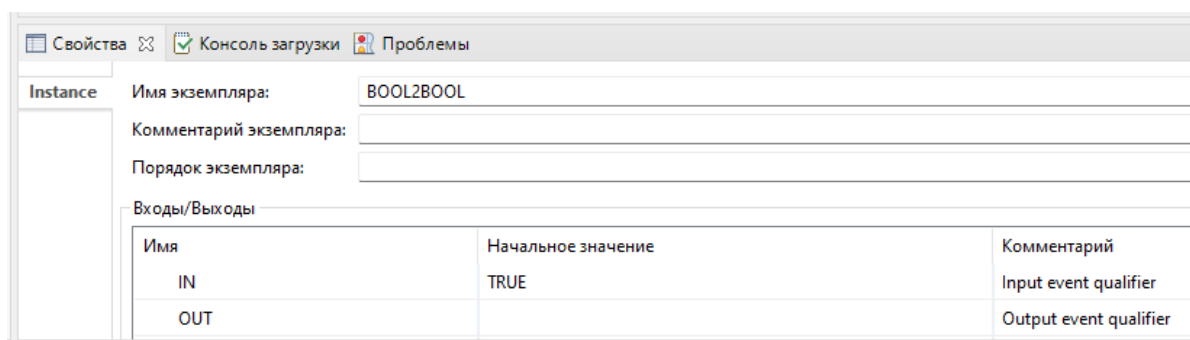


Рисунок 42 – Установка значения TRUE в свойствах ФБ

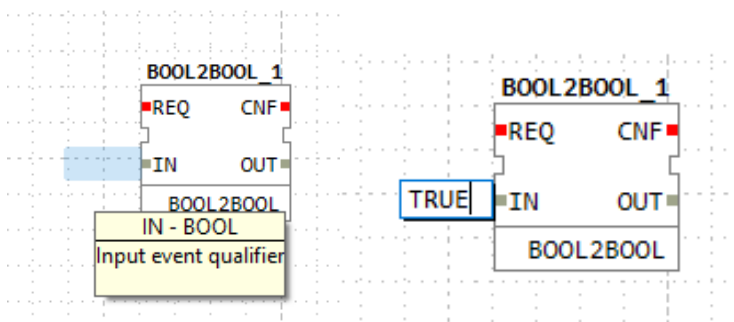


Рисунок 43 – Ввод значения по умолчанию

6.6.6. Инициализация типов для ANY переменных

Параметры типа ANY должны быть обязательно инициализированы. Если параметры не инициализированы, то функциональный блок не будет обрабатывать эти данные (вход или выход).

Существует три варианта инициализации типов данных, которые поддерживает система:

1. Создание соединений параметров типа ANY с параметром определенного типа. Например, следующие соединения определяют параметры блока F_AND как BOOL(см. рисунки [Рисунок 44](#), [Рисунок 45](#)):

6. Разработка проекта

- BOOL2BOOL.OUT (BOOL) -> F_AND.IN (ANY_BIT)
- F_AND.OUT (ANY_BIT) -> BOOL2BOOL.IN (BOOL)

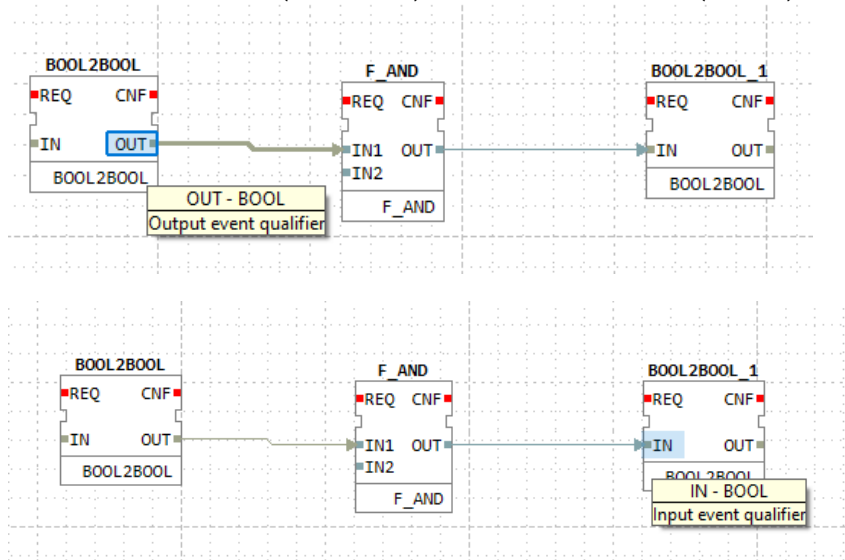


Рисунок 44 – Тип данных BOOL

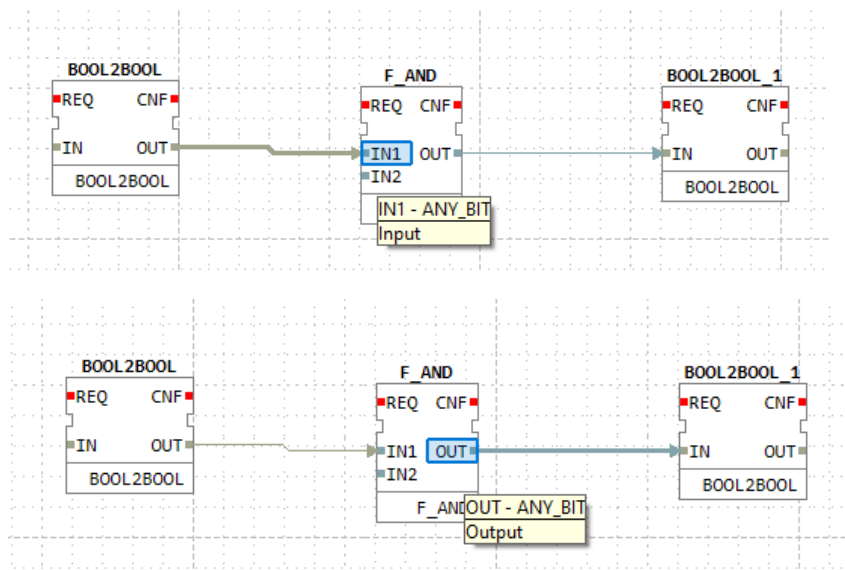


Рисунок 45 – Параметры соединения (типы данных ANY_BIT)

2. Для входных параметров возможна установка начальных значений в функциональном блоке с принудительным указанием типа данных.

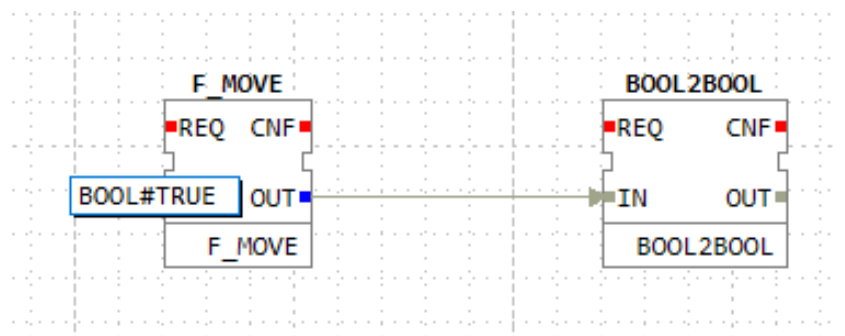


Рисунок 46 – Задание начального значения

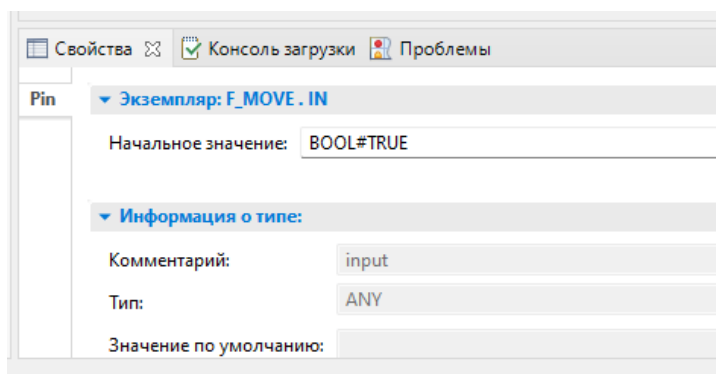


Рисунок 47 – Начальное значение BOOL#TRUE

Например, если задать стартовое значение BOOL#TRUE, то параметр будет определен как тип BOOL (см. рисунки [Рисунок 46](#), [Рисунок 47](#)).

Выходной параметр блока F_MOVE.OUT проинициализировать таким способом нельзя. Его можно проинициализировать только через соединение к блоку с параметром определенного типа.

3. Третий способ инициализации типа данных – это публикация данных на внутренний Modbus Server (см. раздел [10.1 Modbus TCP Server](#)).

На данный момент инициализация возможна для функциональных блоков MBREAD и MBWRITE (см. [Рисунок 48](#)). Для этого нужно в свойствах параметров в поле Modbus прописать ссылку для публикации данных (см. рисунки [Рисунок 49](#), [Рисунок 50](#)). Например,

- MBSRV01.BOOL#QX01 инициализация входного или выходного параметра в тип BOOL
- MBSRV01.INT#QX01 инициализация входного или выходного параметра в тип INT

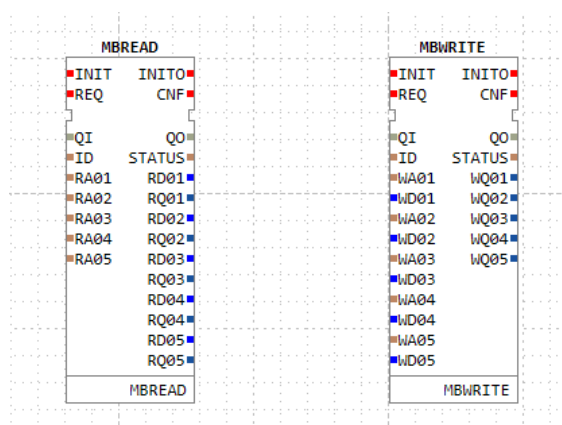


Рисунок 48 – Функциональные блоки MBREAD и MBWRITE

6. Разработка проекта

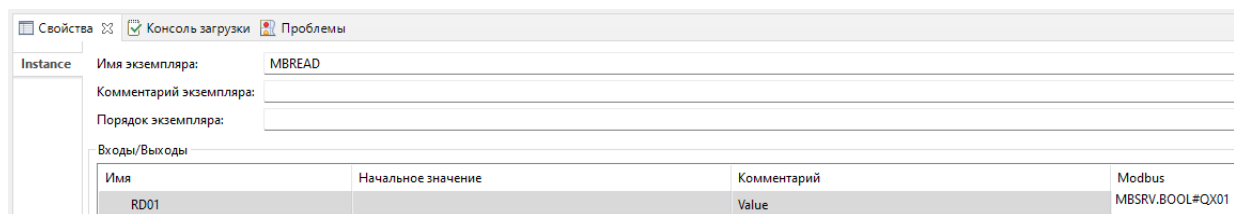


Рисунок 49 – Ссылка для публикации данных RD01

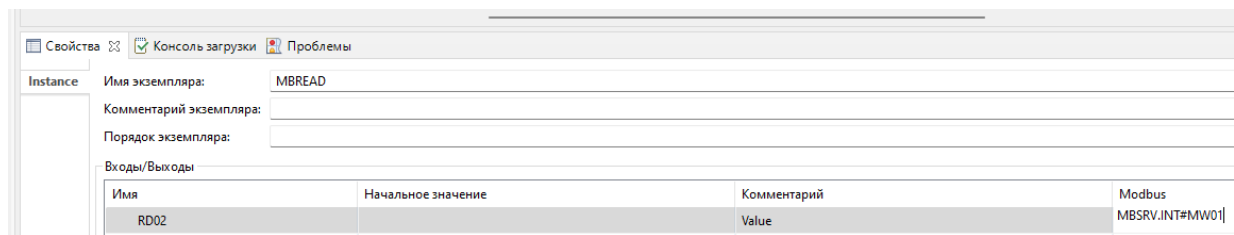


Рисунок 50 – Ссылка для публикации данных RD02

Ссылка для RD01 является логическим типом данных, а для RD02 будет являться целочисленным значением данных.

6.6.7. Блоки с изменяемым количеством входов/выходов

Существуют функциональные блоки, у которых возможно изменение количества входов/выходов. Это блоки типа:

- MUX
- APPEND_STRING
- CSV_WRITER
- ADD
- XOR
- AND
- OR
- SERVER
- CLIENT
- PUBLISH
- SUBSCRIBE

Для того, чтобы изменить количество входных данных (переменных), необходимо создать новый тип функционального блока через копирование существующего:

1. Скопировать блок через контекстное меню в библиотеке проекта (в левой части окна).
2. Перейти в папку User Library и вставить этот блок (см. [Рисунок 51](#))

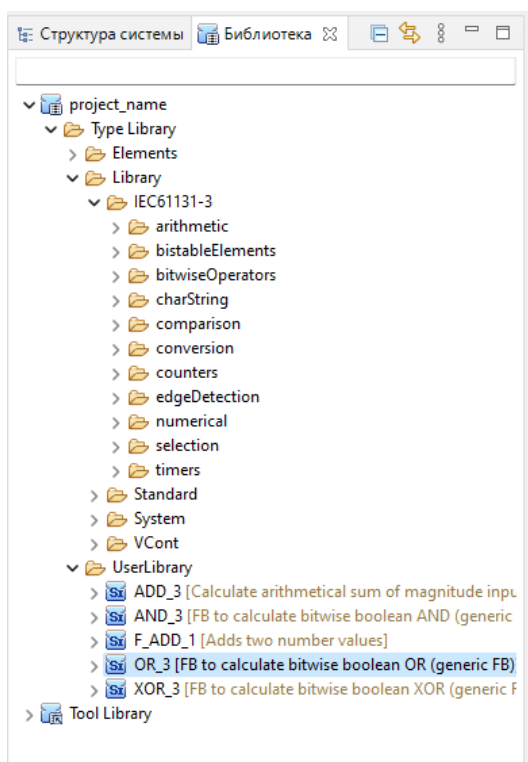


Рисунок 51 – Копирование блока F_AND

3. Переименовать этот блок в соответствии с количеством входов/выходов (команда «Переименовать» в контекстном меню блока).

Правило наименования: <имя блока>_N_K N – количество входов, K – количество выходов.

Например:

- ADD_5, блок FB_ADD с 5-ю входами (см. [Рисунок 52](#))
- CLIENT_0_2, блок CLIENT с 2-я выходами

6. Разработка проекта

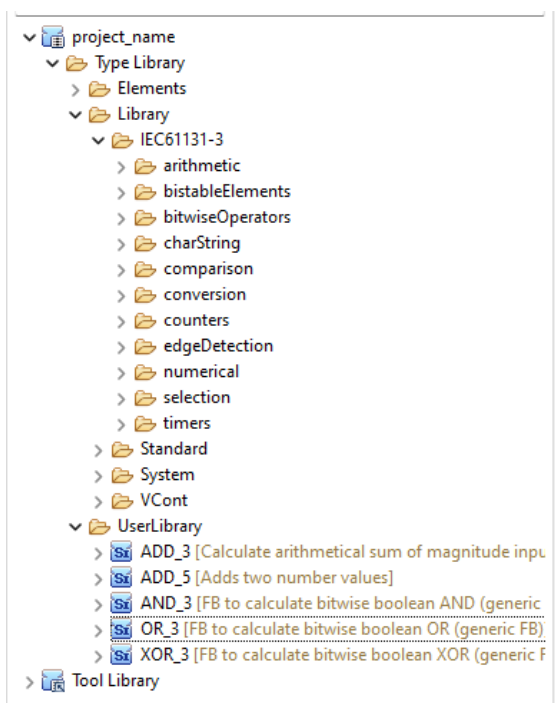


Рисунок 52 – Новый блок ADD_5

4. Модифицировать блок в редакторе для коррекции количества входов/выходов.

Открыть редактор через команду контекстного меню блока «Открыть с помощью» – > «FB Type Editor».

В свойствах блока в секции «Edit Data» нужно увеличить количество входных/выходных параметров (см. [Рисунок 53](#)).

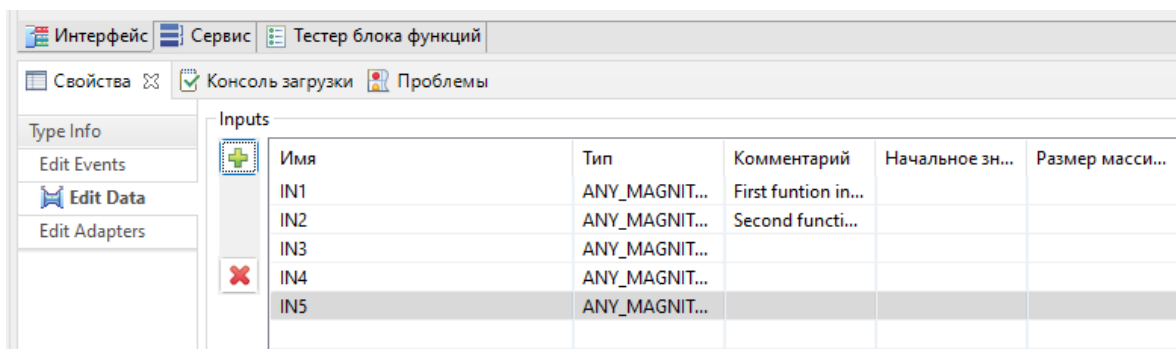


Рисунок 53 – Увеличение количества входов

На графическом представлении блока также произошло увеличение входов (см. [Рисунок 54](#))

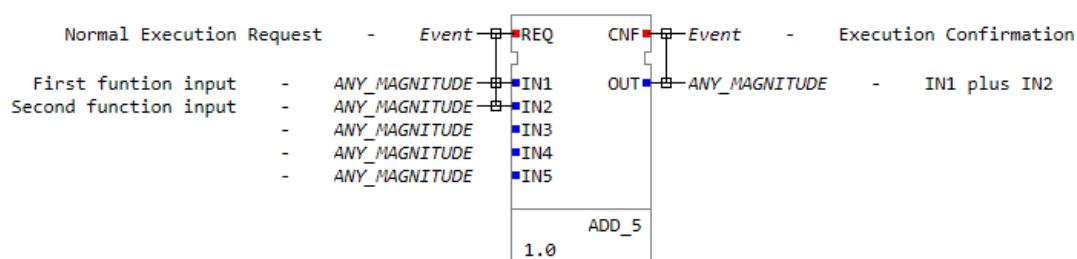


Рисунок 54 – Новый блок ADD_5

5. Далее нужно связать входные значения с входным событием REQ (для выходных значений нужно связывать с соответствующим событием).

Для этого, нажать на событийный параметр REQ и перейти в свойствах на вкладку «Переменная» и отметить все пять входов (см. [Рисунок 55](#)).

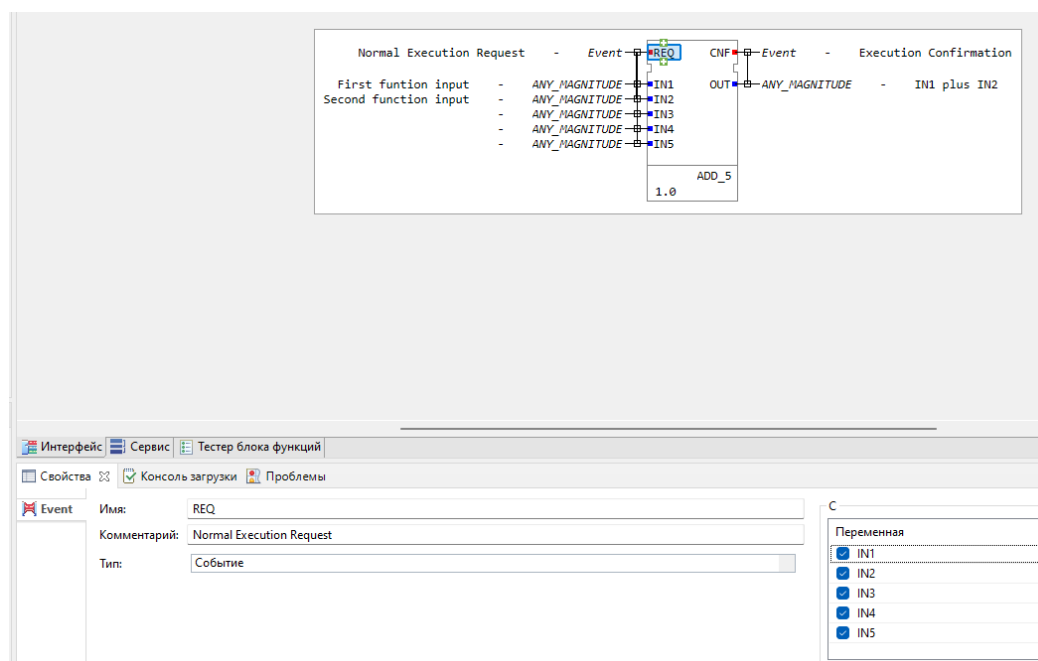


Рисунок 55 – Связывание входных значений с REQ

Сохранить и закрыть блок.

6.6.8. Порядок выполнения функциональных блоков

Порядок выполнения функциональных блоков в контуре определяется явно или неявно:

1. Неявно порядок выставляется по мере добавления блока в контур. Назначить порядок явно можно двумя способами:
2. Явно через установку значения порядка в свойствах каждого блока
3. Явно через команду автоматического расчета «Рассчитать порядок выполнения» (см. [Рисунок 56](#)) (для этого, нужно выделить контур и нажать на быструю клавишу). После выполнения команды автоматический алгоритм

6. Разработка проекта

назначается новый порядок выполнения функциональных блоков (см. [Рисунок 57](#)).

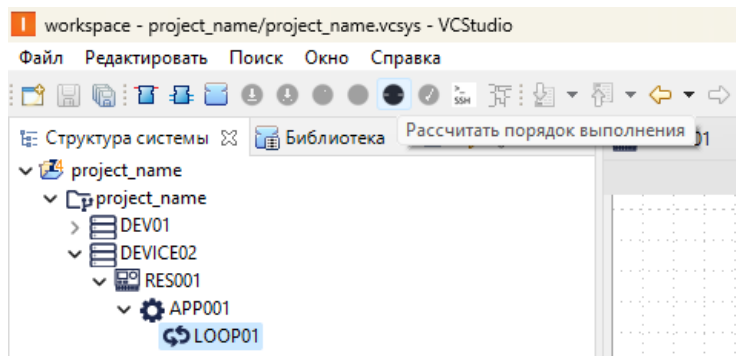


Рисунок 56 – Назначение порядка выполнения ФБ в алгоритме

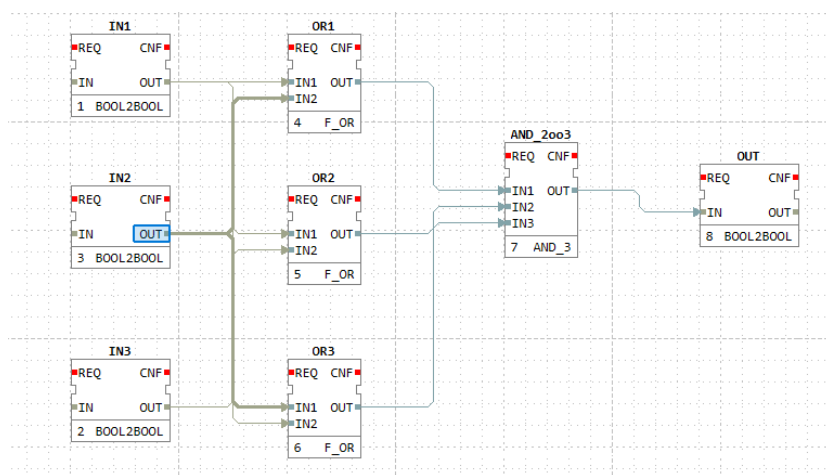


Рисунок 57 – Порядок выполнения ФБ

6.6.9. Замена типа блока на другой в существующем контуре

Команда из контекстного меню блока «Изменить тип блока» заменяет тип существующего блока на другой без разрыва соединений с другими блоками и с сохранением начальных значений блоков (см. [Рисунок 58](#)).

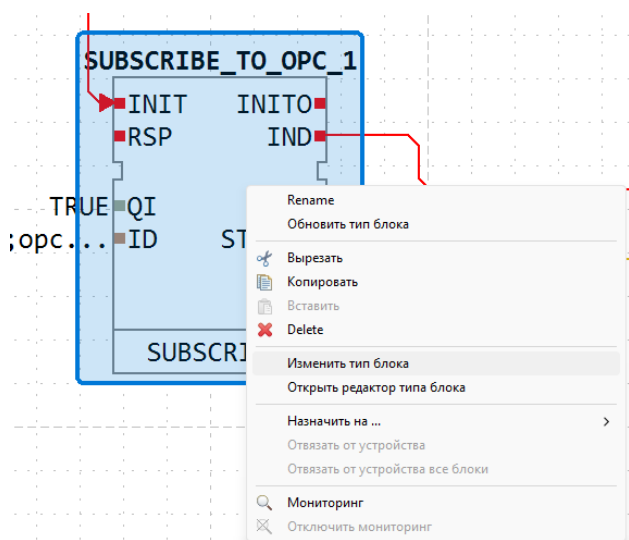


Рисунок 58 – Изменить тип блока

6.6.1. Команда «Обновить тип блока»

Если тип блока был обновлен через редактор блока, то в текущем открытом контуре он может быть обновлен с помощью команды контекстного меню блока «Обновить тип блока». Эта команда обновит количество входов/выходов и их тип.

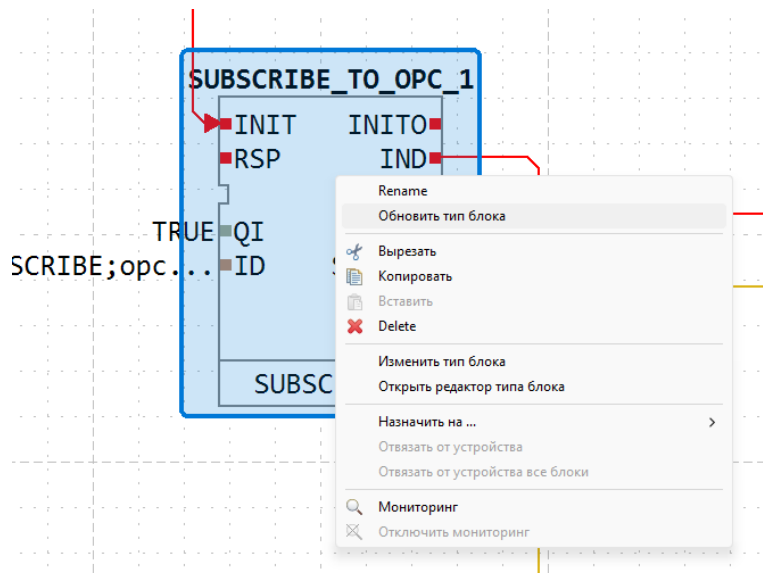


Рисунок 59 – Обновить тип блока

6.7. Создание алгоритмов в контуре событийной задачи

6.7.1. Добавление событийных связей

Если контур принадлежит событийной задаче, то, для обработки блоков, необходимо генерировать событие. Данное событие передаётся по событийной связи.

6.7.2. Добавление генератора событий

Для того, чтобы сгенерировать событие, связанное с включением контроллера (начало его работы), необходимо взять ФБ E_RESTART из библиотеки ФБ. После добавления ФБ, необходимо его переименовать в ФБ START.

После включения контроллера, будет сгенерировано единичное событие на выходе COLD ФБ. Для того, чтобы циклически генерировать события, необходимо добавить ФБ E_CYCLE из библиотеки ФБ. ФБ E_CYCLE генерирует циклические события при поступлении на него события START (см. [Рисунок 60](#)).

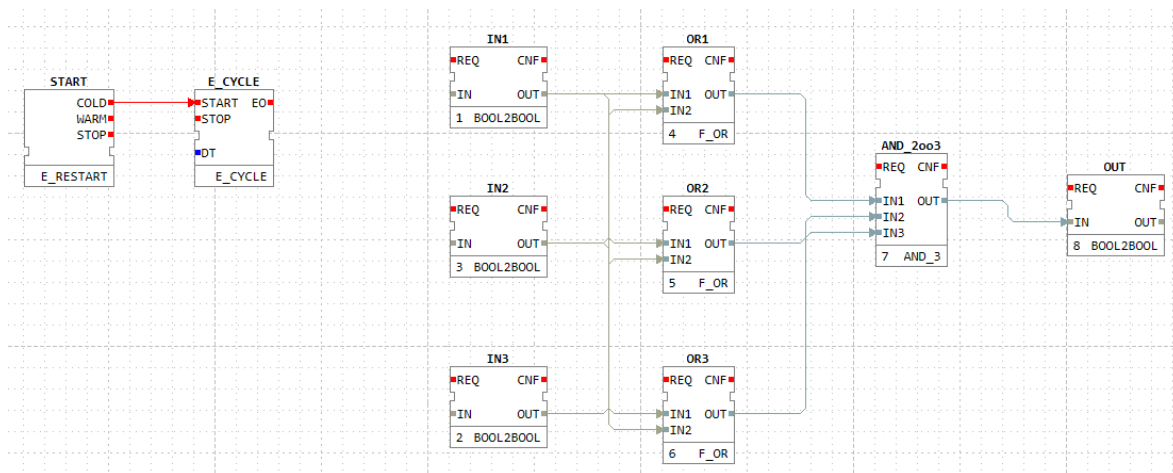


Рисунок 60 – ФБ CYCLE генерирует циклические события при поступлении на него события START

Необходимо указать период генерации событий в «DT» ФБ, равный 1000 ms. После этого, ФБ будет генерировать события на выходе «EO» с указанной периодичностью.

Для того, чтобы все ФБ были последовательно обработаны, необходимо соединить последовательно входы каждого ФБ (см. [Рисунок 61](#)). Последовательность соединений, будет определять последовательность выполнения ФБ.

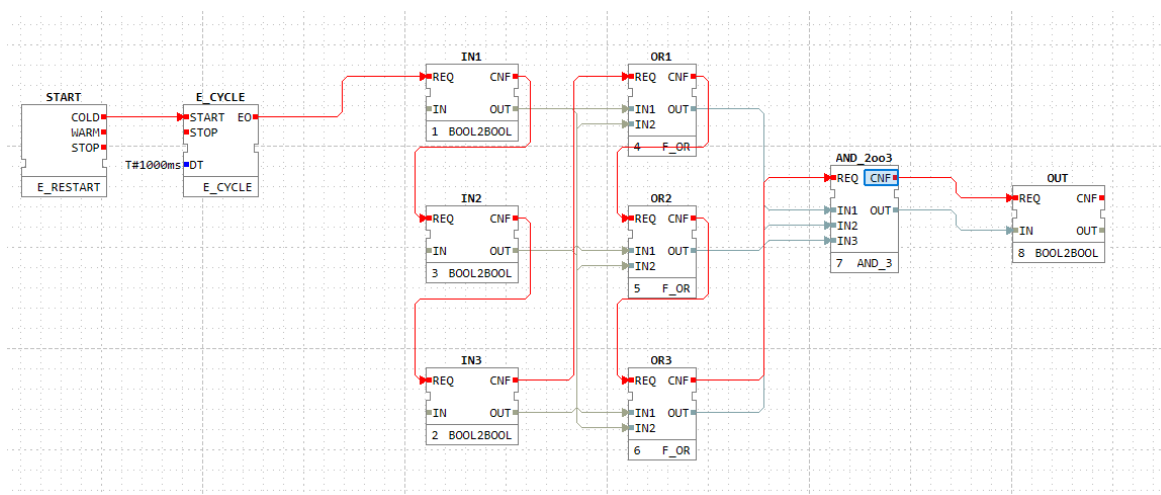


Рисунок 61 – Соединить последовательно входы каждого ФБ

6.7.3. Разветвление и замыкание событий

При создании схем алгоритмов необходимо избегать создания двух ситуаций:

- разветвление событий;
- замыкание событий.

Разветвление событий представлено на рисунке [Рисунок 62](#). При этой ситуации возникает неопределённость порядка выполнения блоков.

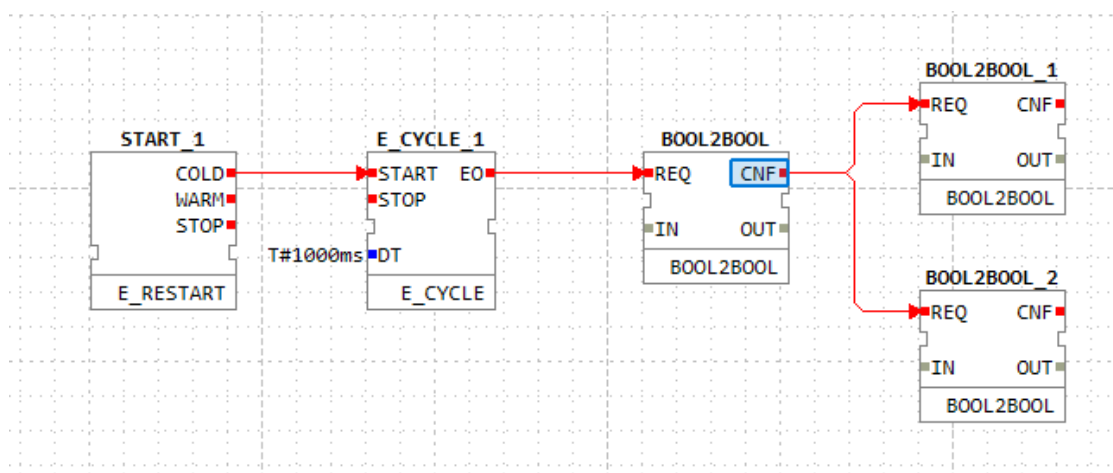


Рисунок 62 – Разветвление событий

«Замыкание» событий представлено на рисунке [Рисунок 63](#). При этой ситуации блоки будут обрабатываться без паузы – сразу после выполнения всей очереди событий, это приведёт к 100% нагрузке на ресурс.

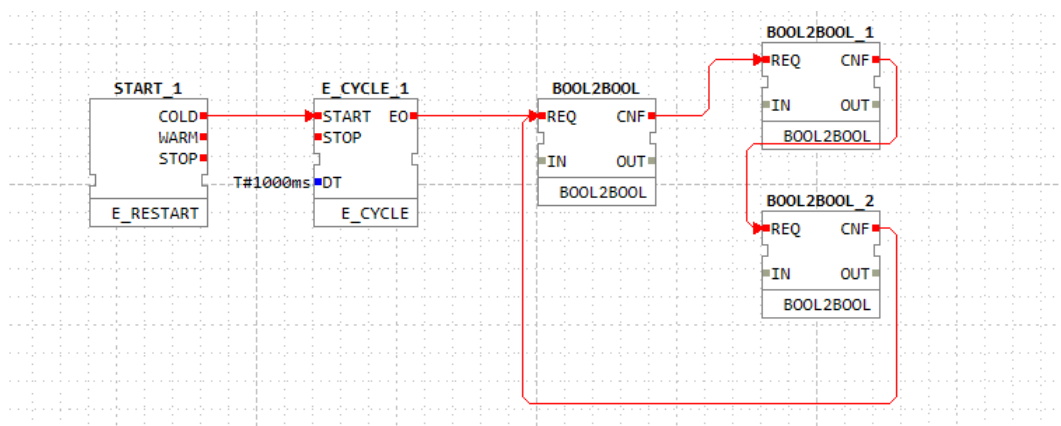


Рисунок 63 – «Замыкание» событий

6.8. Создание соединений между контурами

Для создания соединений между контурами, необходимо использовать объект Reference (см. [Рисунок 64](#)).

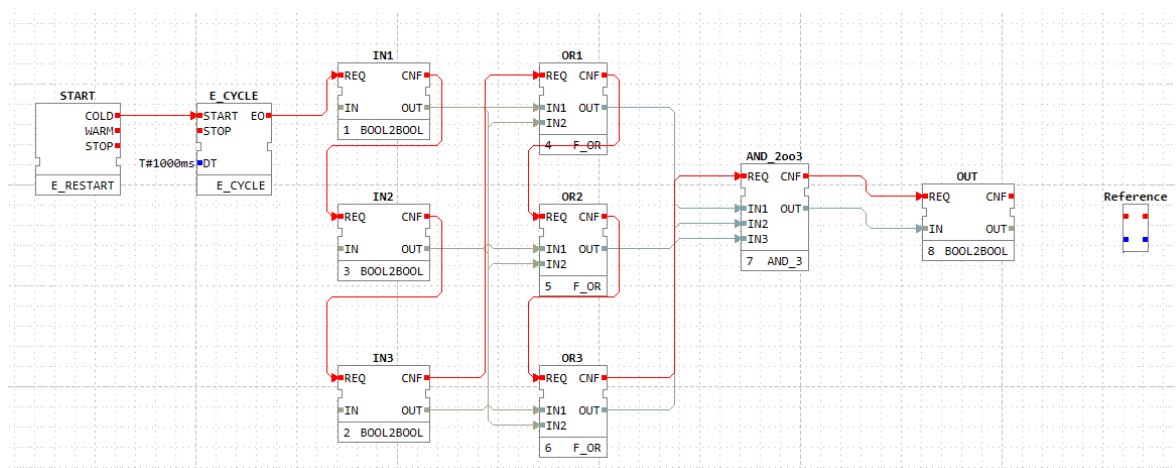


Рисунок 64 – Объект Reference

Связи могут быть как событийные, так и с данными. Поэтому используется как один, так и другой порт. Чтобы передать соединение, необходимо сначала установить объект Reference в контур с источником данных, и соединить эти данные. В момент соединения объект Reference изменит своё имя с адресом этого параметра BLOCK.PARAM (см. [Рисунок 65](#)).

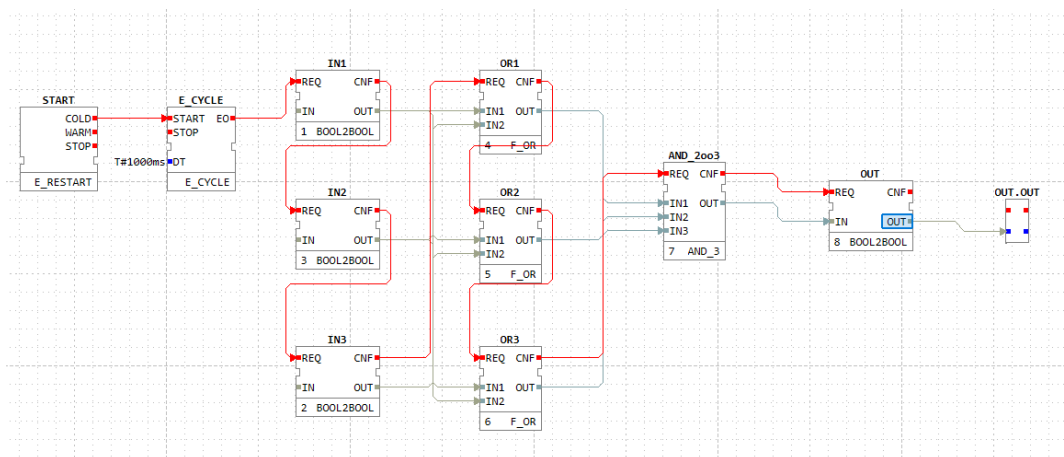


Рисунок 65 – Изменение имени ФБ Reference

Выход данного объекта Reference можно использовать в качестве источника данных для других ФБ, см рисунок [Рисунок 66](#).

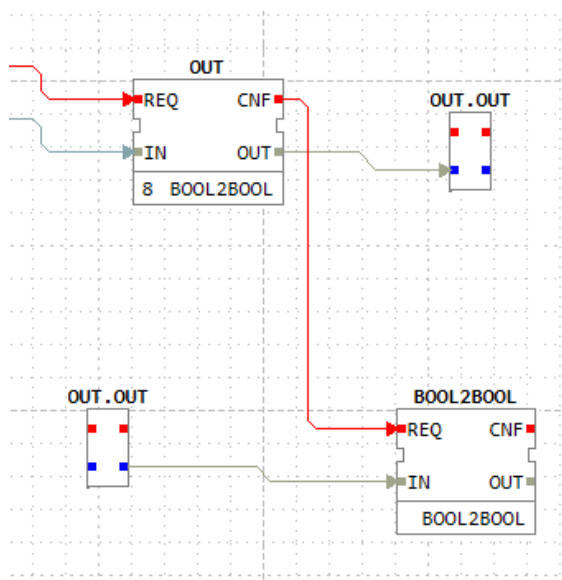


Рисунок 66 – Источник данных для других ФБ

Ссылка данного объекта поддерживает только одно соединение: данных или событий.

Для создания соединения между двумя контурами нужно скопировать объект Reference на стороне источника и вставить рядом с приёмником, соединив выходной порт с необходимым портом блока-приёмника (см. [Рисунок 67](#)).

В момент вставки в другой контур объект Reference примет уникальное имя ПРИЛОЖЕНИЕ.КОНТУР.БЛОК.ПАРАМЕТР.

6. Разработка проекта

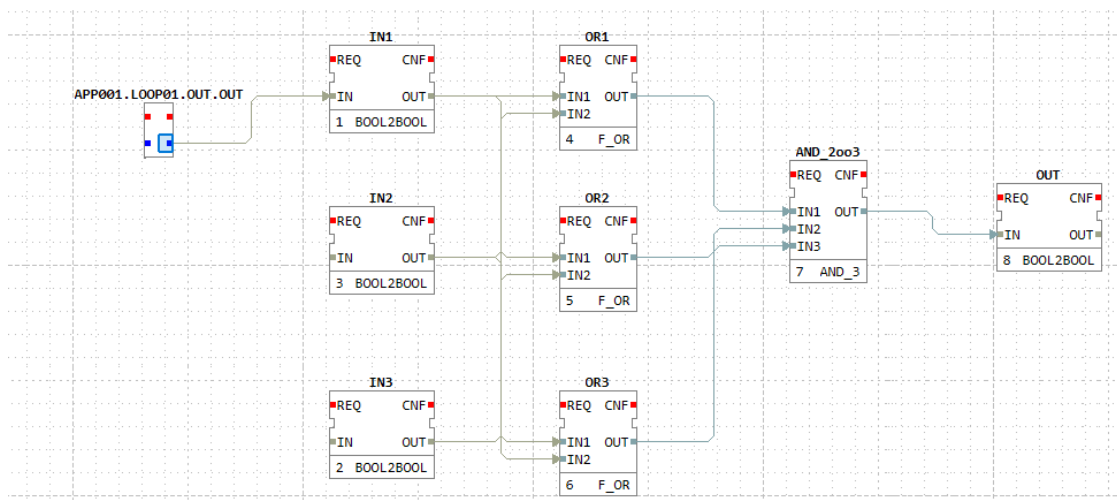


Рисунок 67 – Соединение объекта Reference с другим контуром

При изменении имени Блока, Контура или Приложения в источнике данных, происходит изменение имени объекта Reference в контурах (см. рисунки [Рисунок 68](#), [Рисунок 69](#)).

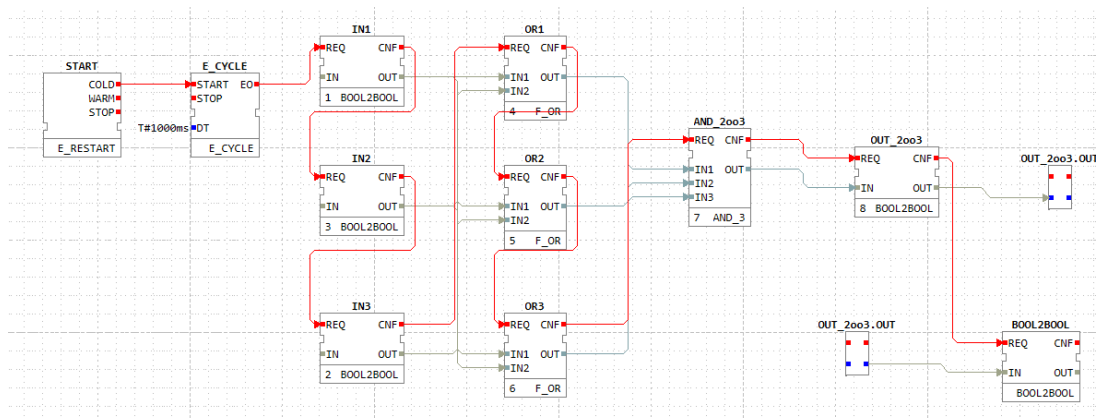


Рисунок 68 – Изменение имени объекта первого контура

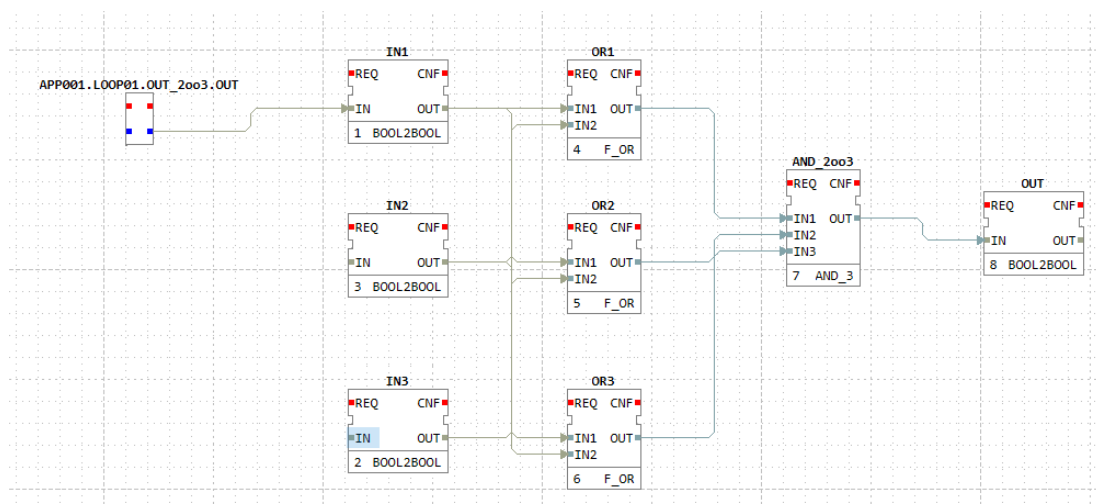


Рисунок 69 – Изменение имени объекта второго контура

7. Отладка проекта

7.1. Запуск среды исполнения (vcont)

Для загрузки логики и отладки логики в контроллере, необходимо запустить Vcont (см. [Рисунок 70](#)). Запуск должен быть выполнен с правами администратора.

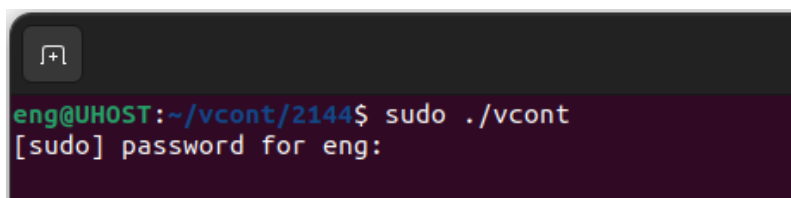


Рисунок 70 – Запуск файла виртуального контроллера (Vcont)

Данное окно после запуска закрывать не нужно!

После запуска среда исполнения доступна через локальный сетевой адрес с портом 61499. Данный порт установлен по умолчанию.

7.2. Он-лайн команды

7.2.1. Загрузка проекта

Для начала работы с ресурсом, необходимо убедиться, что он запущен. Контроллер должен быть доступен по IP-адресу и порту, указанному в свойствах ресурса (см. [Рисунок 71](#)).

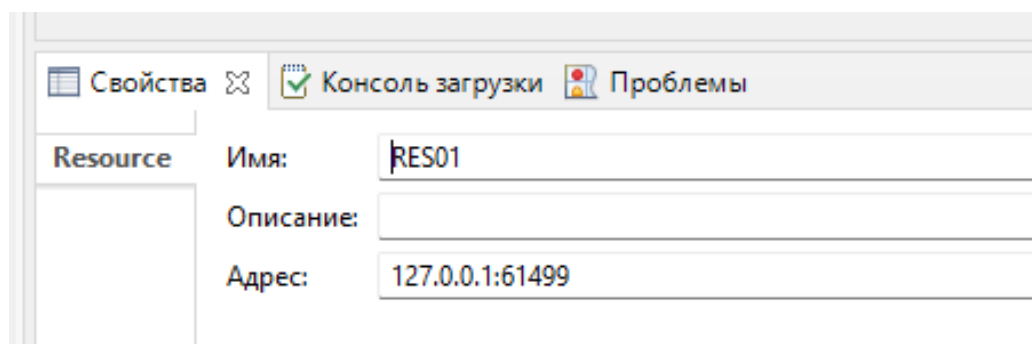


Рисунок 71 – Свойства ресурса

Загрузка ресурса осуществляется через контекстное меню. Для этого необходимо выделить ресурс, и из контекстного меню выбрать команду «Загрузить» (см. [Рисунок 72](#)).

7. Отладка проекта

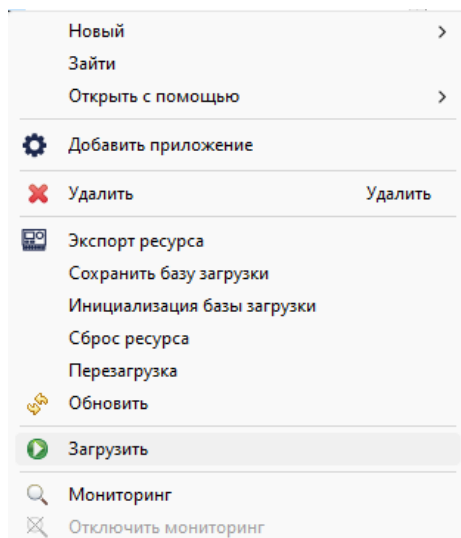


Рисунок 72 – Загрузка ресурса

После нажатия происходит загрузка ресурса. Проверить параметры загрузки можно в разделе «Консоль загрузки» (см. [Рисунок 73](#)).

Успешность загрузки можно проконтролировать в Консоли загрузки:

- в конце списка команд должны присутствовать сообщения по запуску всех задач, которые есть в ресурсе (команда «START»)
- отсутствие любых ошибок в процессе загрузки

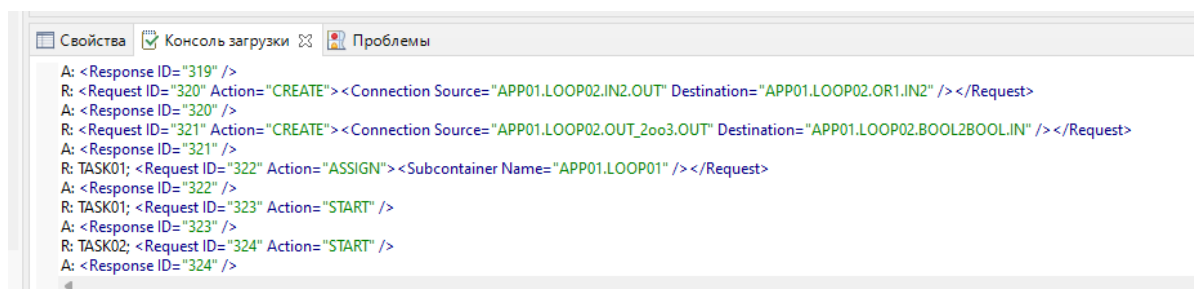


Рисунок 73 – Консоль загрузки

7.2.2. Сохранение базы загрузки

Данная команда выполняет сохранение параметров для «теплого» старта ресурса. Для сохранения базы загрузки, необходимо выделить ресурс и из контекстного меню выбрать команду «Сохранить базу загрузки».

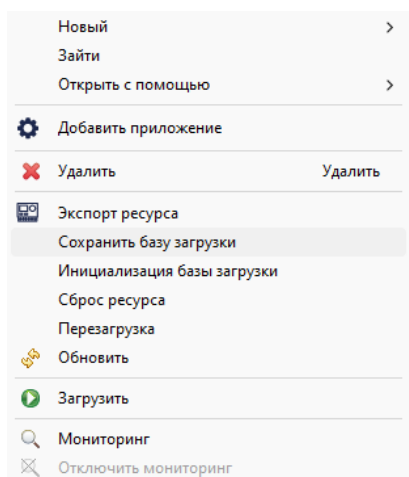


Рисунок 74 – Сохранение базы загрузки

Подтвердить выполнение команды – нажать «Ок» (см. [Рисунок 75](#)).

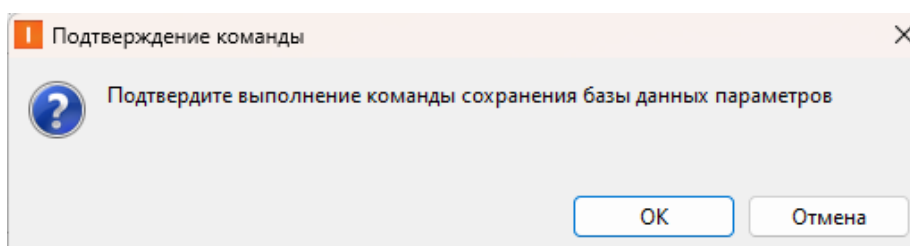


Рисунок 75 – Подтверждение выполнение команды

Проконтролировать выполнение команды можно в разделе «Консоль загрузки» (см. [Рисунок 76](#)).

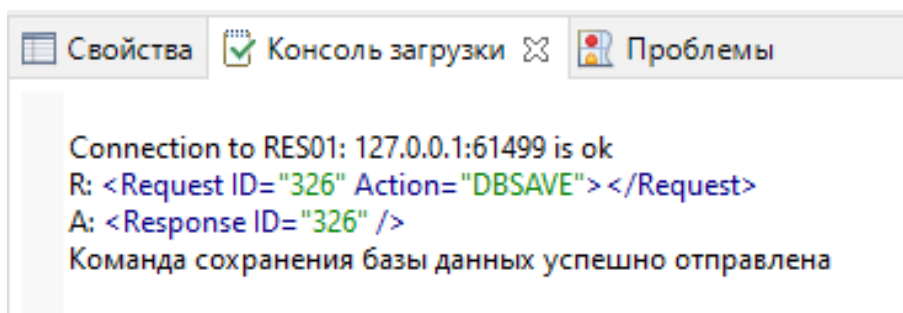


Рисунок 76 – Контроль выполнения команды

7.2.3. Инициализация базы загрузки

Чтобы очистить данные «теплого» старта необходимо выбрать команду «Инициализация базы загрузки» (см. [Рисунок 77](#)).

7. Отладка проекта

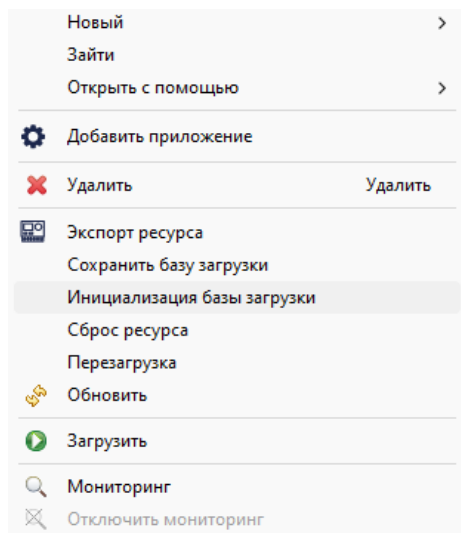


Рисунок 77 – Команда «Инициализация базы загрузки»

Подтвердить выполнение команды – нажать «Ок» (см. [Рисунок 78](#)).

Внимание, после выполнения команды ресурс будет перезагружен.

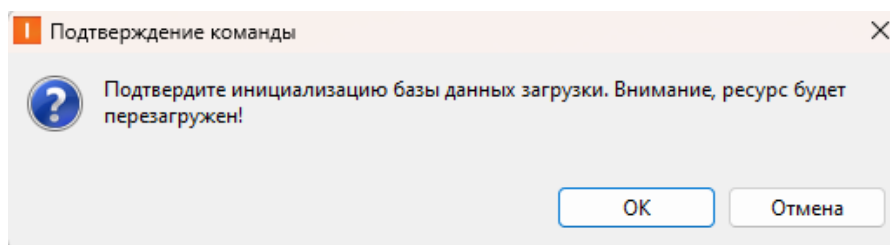


Рисунок 78 – Подтверждение выполнения команды

Проконтролировать выполнение команды можно в разделе «Консоль загрузки» (см. [Рисунок 79](#)).

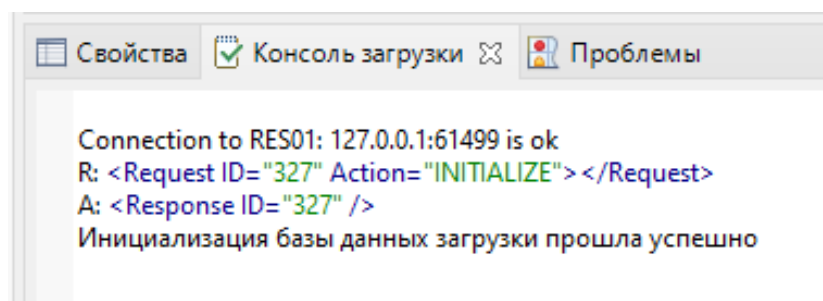


Рисунок 79 – Контроль выполнения команды

После инициализации базы загрузки, ресурс перезагружается с нулевыми данными.

7.2.4. Сброс ресурса

Команда «Сброс ресурса» очищает базу загрузки, удаляет файл загрузки и перезагружает ресурс.

Для выполнения команды, необходимо выделить ресурс, выделить ресурс и из контекстного меню выбрать данную команду (см. [Рисунок 80](#)).

Внимание, после выполнения команды ресурс будет перезагружен.

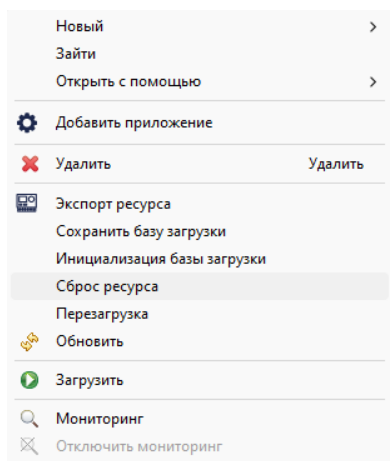


Рисунок 80 – Команда «Сброс ресурса»

Подтвердить выполнение команды – нажать «Ок» (см. [Рисунок 81](#)).

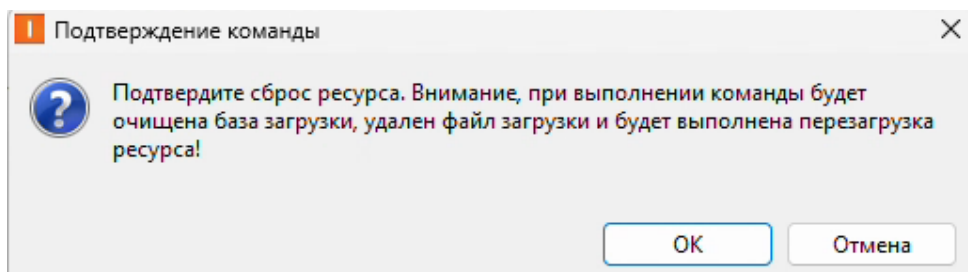


Рисунок 81 – Подтверждение выполнения команды

Произойдет сброс ресурса.

7.2.5. Перезагрузка

Для выполнения команды «Перезагрузка», необходимо выделить ресурс, выделить ресурс и из контекстного меню выбрать данную команду (см. [Рисунок 82](#)).

7. Отладка проекта

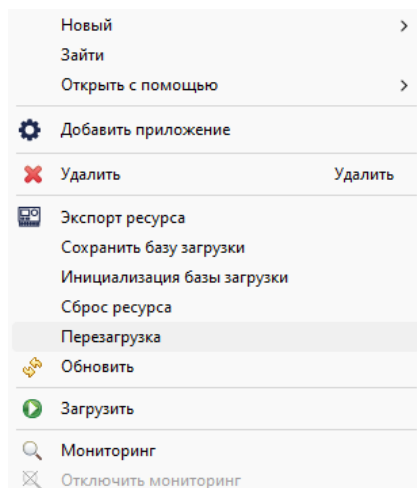


Рисунок 82 – Команда «Перезагрузка»

Подтвердить выполнение команды – нажать «Ок» (см. [Рисунок 83](#)).

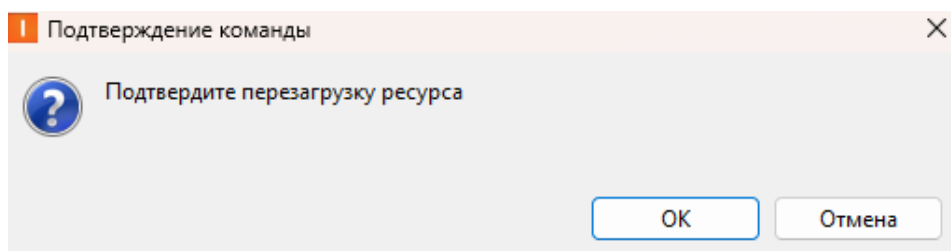


Рисунок 83 – Подтверждение выполнения команды

В этот момент, происходит выключение ресурса и служба, которая настроена для выполнения Runtime, заново запустит ресурс.

7.3. Режим мониторинга

7.3.1. Включение мониторинга

Включить мониторинг можно двумя способами:

- Через кнопку «Включить мониторинг» на панели быстрого доступа (активна при выборе ресурса) (см. [Рисунок 84](#))

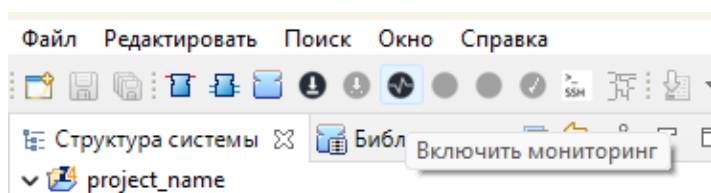


Рисунок 84 – Включение мониторинга

- Через выбор команды «Мониторинг» в контекстном меню ресурса (см. [Рисунок 85](#)).

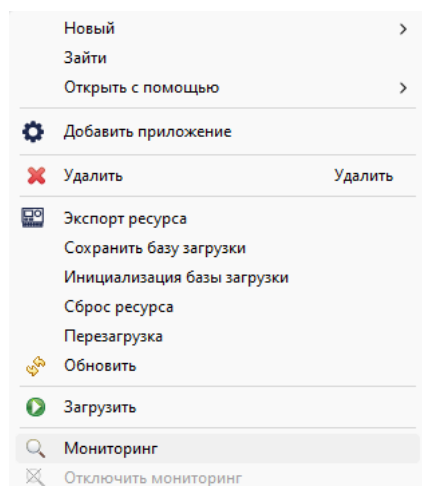


Рисунок 85 – Команда «Мониторинг»

7.3.2. Добавление параметров в мониторинг

Для отображения он-лайн значений параметров функциональных блоков нужно в окне редактора контура выделить необходимые ФБ и в контекстном меню блоков выбрать команду «Мониторинг» (см. [Рисунок 86](#)).

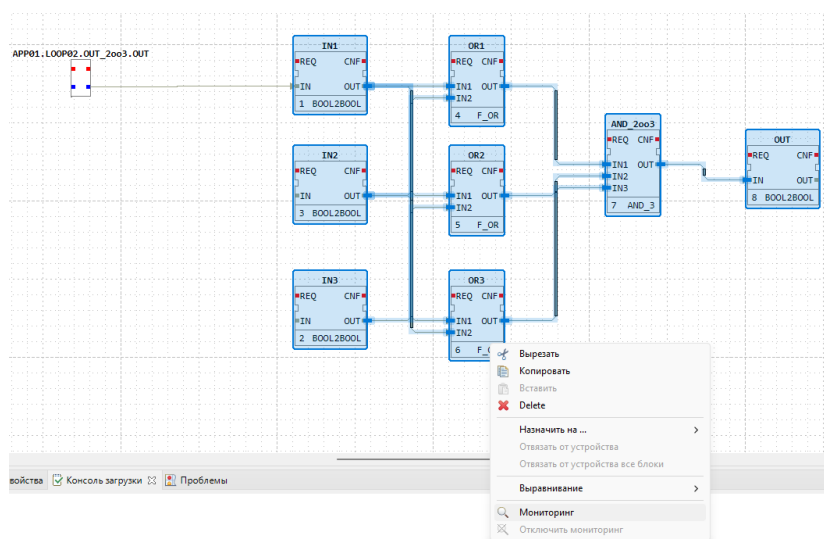


Рисунок 86 – Команда «Мониторинг»

Для того, чтобы увидеть значения параметров ресурс должен быть загружен. По умолчанию фон значений в режиме он-лайн будет жёлтым (см. [Рисунок 87](#)).

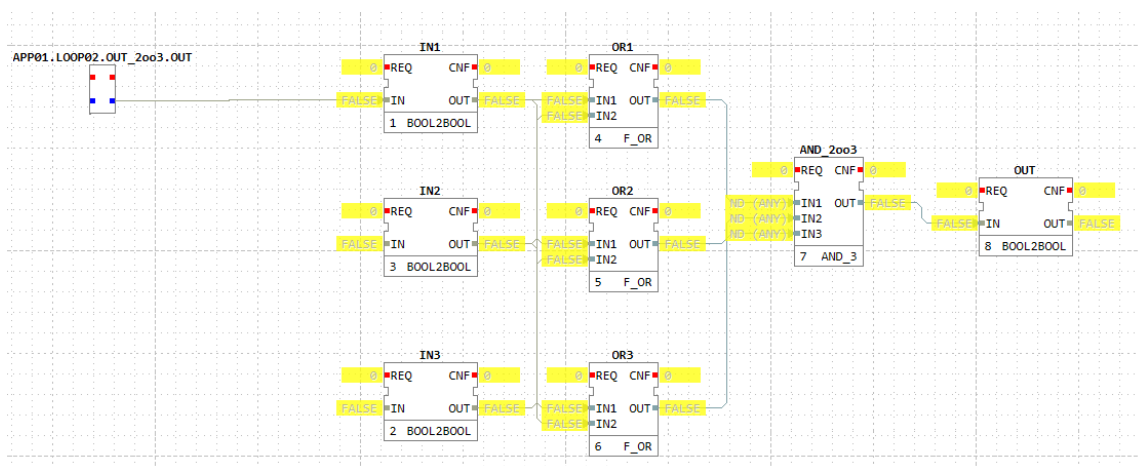


Рисунок 87 – Режим мониторинга ФБ

7.3.3. Изменение входных параметров в он-лайн режиме

Значения входных не соединённых параметров блоков можно изменять в режиме он-лайн (см. [Рисунок 88](#)).

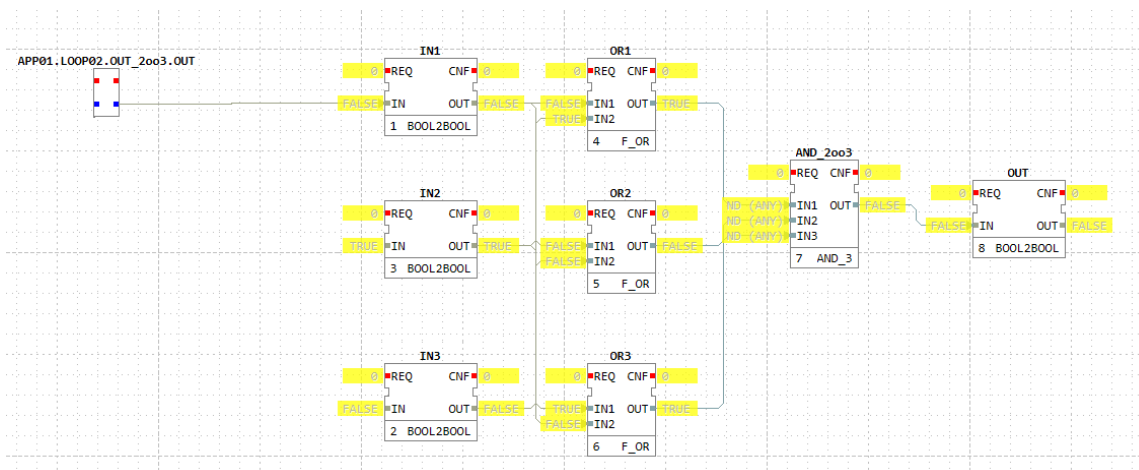


Рисунок 88 – Изменение значений ФБ

7.3.4. Форсирование переменных

Значения любых параметров могут быть принудительно изменены в режиме он-лайн. Данная команда называется форсированием (Force). Для форсирования переменных ФБ необходимо выбрать один из параметров – входную или выходную переменную и через контекстное меню выбрать команду «Force» (см. [Рисунок 89](#)).

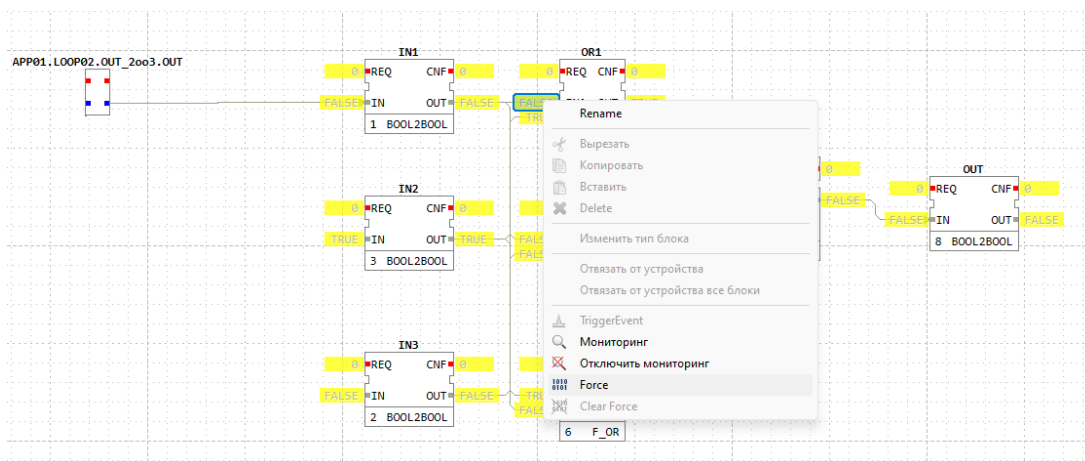


Рисунок 89 – Режим «Force»

Откроется окно для ввода нового значения переменной (см. [Рисунок 90](#)). Для изменения значения переменной, необходимо ввести необходимое значение. Далее нужно подтвердить значение, нажав «Ок». Значение переменной изменится (см. [Рисунок 91](#)).

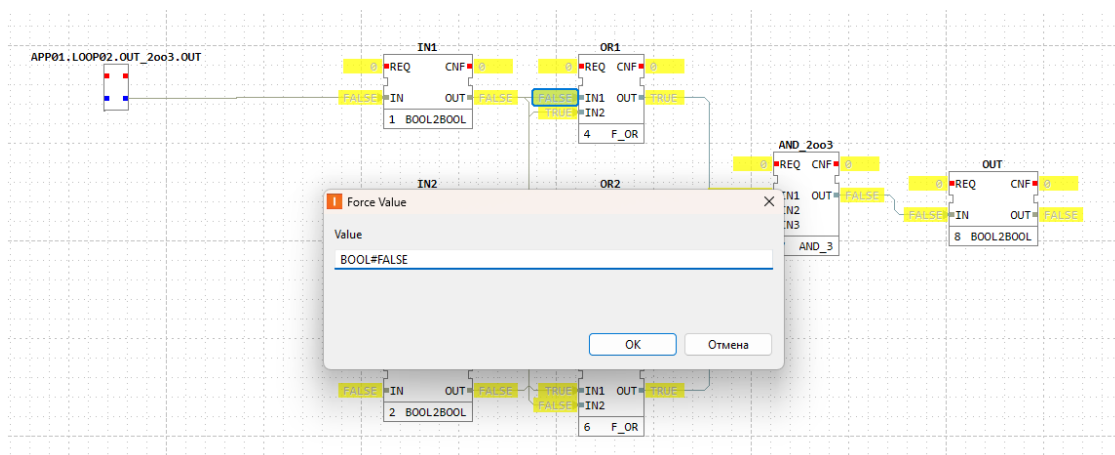


Рисунок 90 – Выбор значения переменной

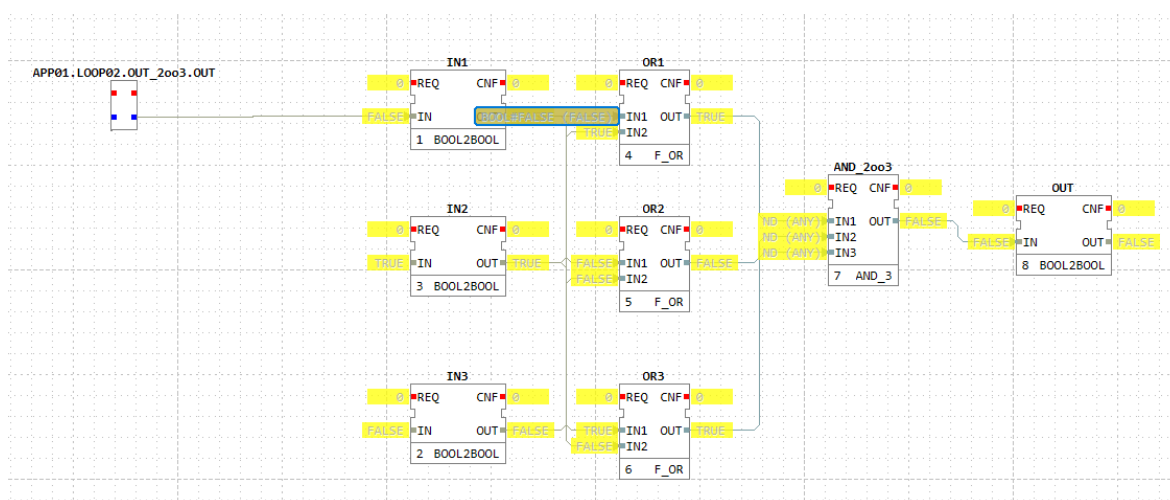


Рисунок 91 – Изменение значения переменной

7.3.5. Принудительная генерация событий

Перед применением принудительной генерации событий, необходимо убедиться, что задача контура является событийной. Для этого необходимо перейти в необходимый ресурс во вкладку «Задачи» (см. [Рисунок 92](#)).

| Задачи Modbus TCP Сеть Приложение | | |
|-----------------------------------|-------------|-----------------------|
| Сведения о задачах | | |
| Имя интерфейса | Тип задачи | Период выполнения, мс |
| TASK01 | Циклический | 500 |
| TASK02 | Событийный | |

Рисунок 92 – Проверка типа задачи

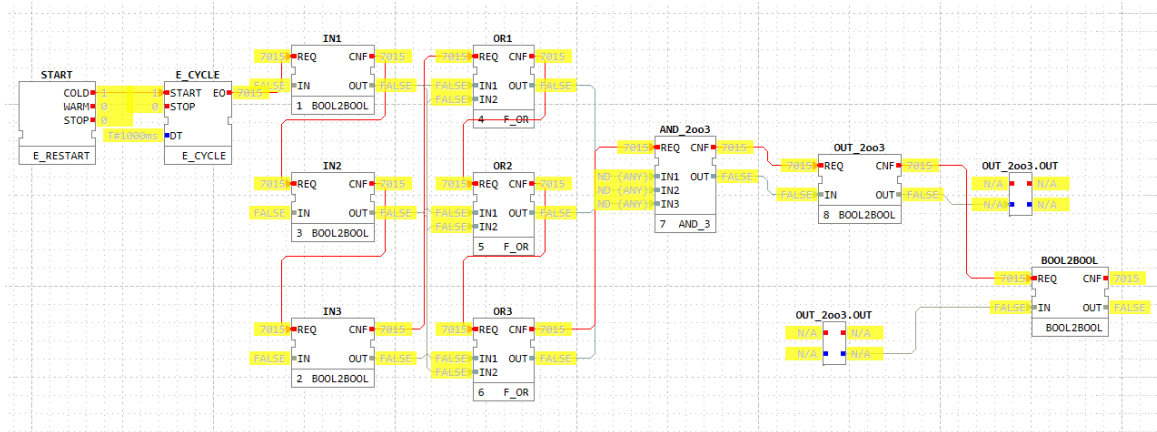


Рисунок 93 – Режим мониторинга событийного контура

Для генерации события нужно выбрать событийный параметр и в контекстном меню выбрать команду TriggerEvent (см. [Рисунок 94](#)). После этого, сгенерируется событие ФБ, которое увеличит счетчик событий на один (см. [Рисунок 95](#)).

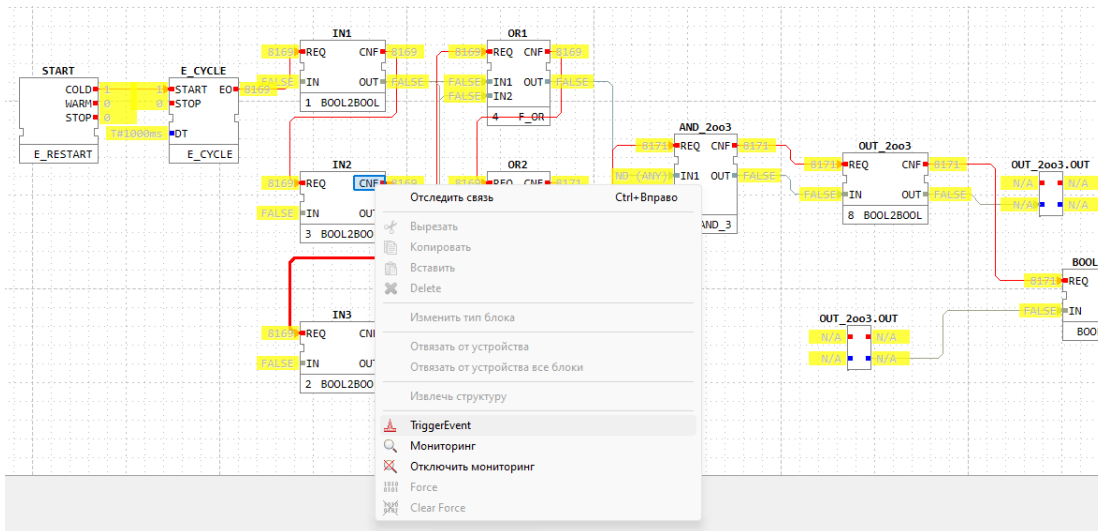


Рисунок 94 – Генерация вручную событий

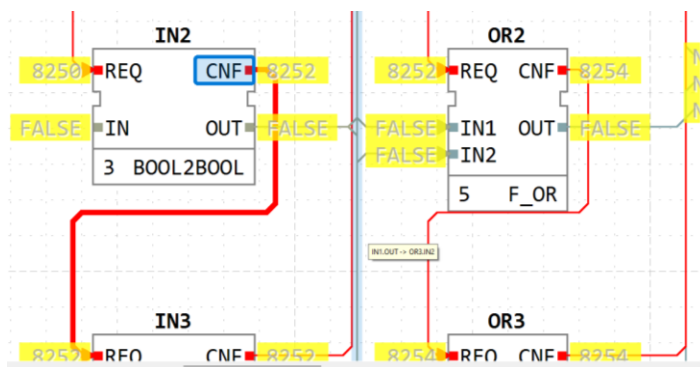


Рисунок 95 – Расхождение в значениях входа и выхода ФБ IN2

Показания событийных входов – не буквальное значение событийного входа, а количество событий, которое пришло на вход событийного параметра или вышло из событийного параметра. С каждым событием связан некоторый алгоритм, который находится внутри блока. И значение показывает сколько раз был выполнен алгоритм, который был связан с этим событийным входом.

8. Настройка контроллера

8.1. Загрузочный файл

Загрузочный файл представляет из себя набор команд, которые Runtime использует для создания базы данных. Этот набор команд идентичен набору команд, которые посылаются в режиме он-лайн при выполнении команды «Загрузка» для ресурса. Список команд загрузки также можно видеть в окне «Консоль загрузки» вместе с ответами от контроллера.

В момент старта контроллер считывает загрузочный файл и выполняет команды по созданию алгоритмов, которые в нем описаны.

Для создания загрузочного файла, необходимо выделить ресурс и из контекстного меню выбрать команду «Экспорт ресурса» (см. [Рисунок 96](#)).

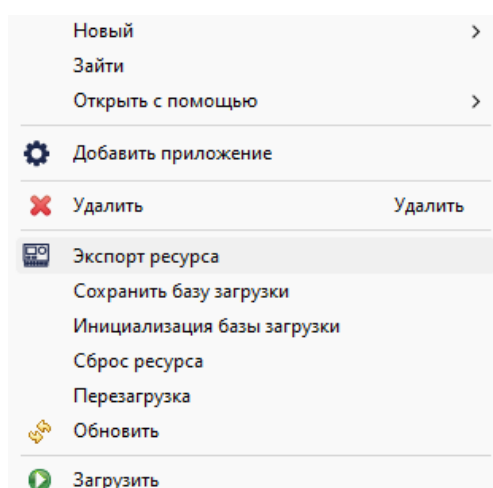


Рисунок 96 – Команда «Экспорт ресурса»

При выполнении команды создаётся файл в корне проекта с названием RES01.fboot (см. [Рисунок 97](#)).

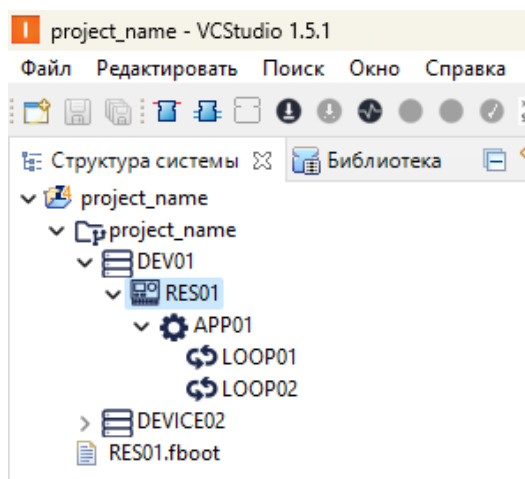


Рисунок 97 – Выполнение команды «Экспорт ресурса»

Для того, чтобы контроллер мог загрузить этот файл, его нужно переименовать в `vcont.fboot`. Данный файл нужно перенести в директорию с исполняемым файлом для `vcont`.

9. Обслуживание

9.1. Восстановление проекта из резервной копии

Настройка создания резервных копий, находится по пути: «Окно» – «Параметры» – «Общие» – «Рабочая область» – «Локальная история» (см. [Рисунок 98](#)).

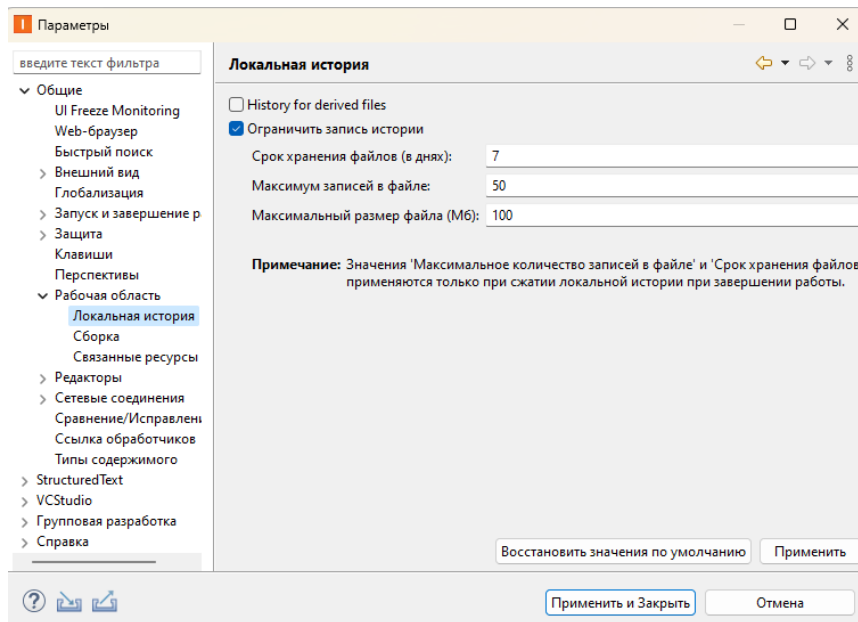


Рисунок 98 – Настройка создания резервных копий

Резервные копии создаются каждый раз при сохранении проекта.

Для того, чтобы восстановить проект из резервной копии необходимо из контекстного меню проекта в дереве навигатора выбрать команду «Replace With» – «Локальная история» (см. [Рисунок 99](#)). Появится список дат для замены проекта (см. [Рисунок 100](#))

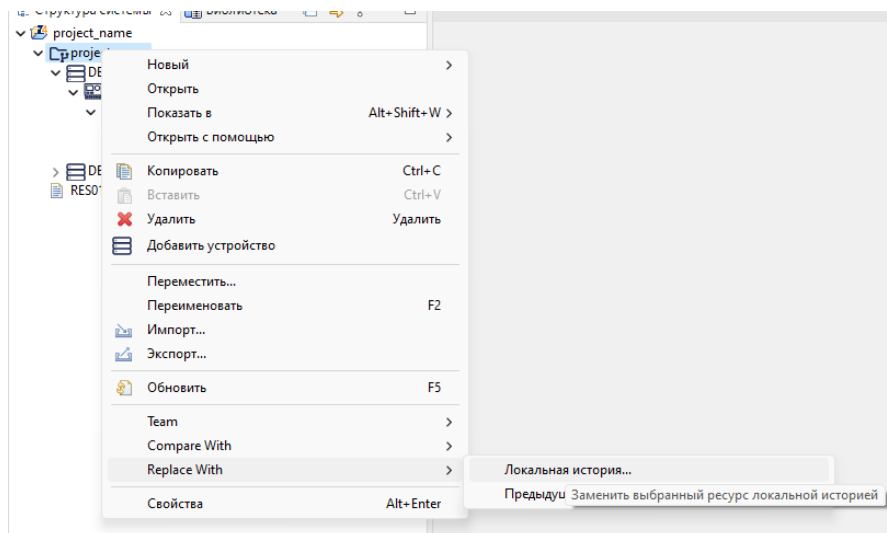


Рисунок 99 – Команда «Replace With» – «Локальная история»

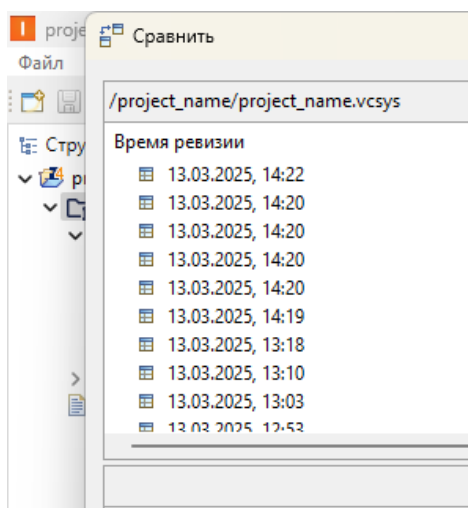


Рисунок 100 – Список дат для замены

Для восстановления копии проекта нужно выбрать дату резервной копии и нажать кнопку «Заменить».

10. Коммуникационные возможности

В настоящий момент система поддерживает следующие протоколы:

Modbus TCP (Master/Slave), OPCUA Client/Server.

10.1. Modbus TCP Server

Для предоставления данных внешним клиентам может использоваться встроенный Modbus TCP Server.

Modbus TCP сервер должен быть создан пользователем. Несколько серверов может быть создано для изоляции данных между устройствами.

Для каждого созданного Modbus TCP сервера используется стандартное пространство адресов (см. [Таблица 3](#)).

Таблица 3. Карта адресов сервера Modbus TCP

| Имя регистра | Modbus Адрес | Размер | Доступ | Функция | Типы данных |
|--------------|--------------|--------|--------|------------|-------------|
| QX0-QX65535 | 0-65535 | 1 бит | Запись | 01, 05, 15 | BOOL |
| IX0-IX65535 | 0-65535 | 1 бит | Чтение | 02 | BOOL |
| IW0-IW65535 | 0-65535 | 16 бит | Чтение | 04 | INT, WORD |
| MW0-MW29999 | 0-65535 | 16 бит | Запись | 03, 06, 16 | INT, WORD |
| MD0-MD9999 | 30000-49998 | 32 бит | Запись | 03, 06, 16 | DINT, REAL |
| ML0-ML3750 | 50000-65000 | 64 бит | Запись | 03, 06, 16 | LINT, LREAL |

10.1.1. Конфигурирование Modbus Server

Сервер добавляется в редакторе ресурса во вкладке Modbus TCP (см. [Рисунок 101](#)). Пользователь указывает следующие параметры:

Таблица 4. Параметры Modbus TCP сервера

| Параметр | Описание | Пример |
|-----------------|---|---------|
| Имя интерфейса | Название Modbus TCP сервера. К привязке к этому имени будет осуществляться публикация и чтение данных от параметров функциональных блоков | MBSRV01 |
| IP-адрес | IP адрес клиента, который может подключиться к серверу. Если указан 0.0.0.0, то подключиться к серверу может любой клиент | 0.0.0.0 |
| Физический порт | Сетевой порт Modbus TCP сервера | 1512 |

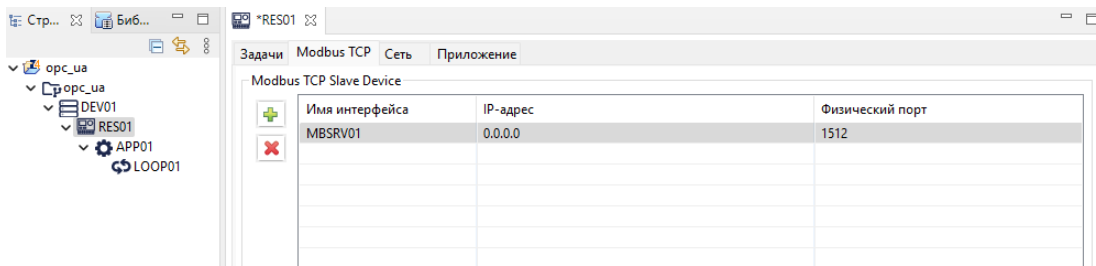


Рисунок 101 – Добавление Modbus TCP сервера

10.1.2. Привязка параметров к регистрам Modbus TCP сервера

Для того, чтобы связать входной или выходной параметр функционального блока с регистром сервера Modbus, нужно параметру определить адрес регистра Modbus TCP сервера (см. Рисунок 102).

Адрес указывается в формате <MODBUS_SERVER_NAME>.<TYPE#><ADDRESS>:
MODBUS_SERVER_NAME – имя сервера, указанное пользователем в редакторе ресурса

TYPE# – тип данных, используется для инициализации типа Any. Если инициализация типа не требуется, то может не указываться.

ADDRESS – адрес параметра в пространстве сервера Modbus (см. Таблица 3. Карта адресов)

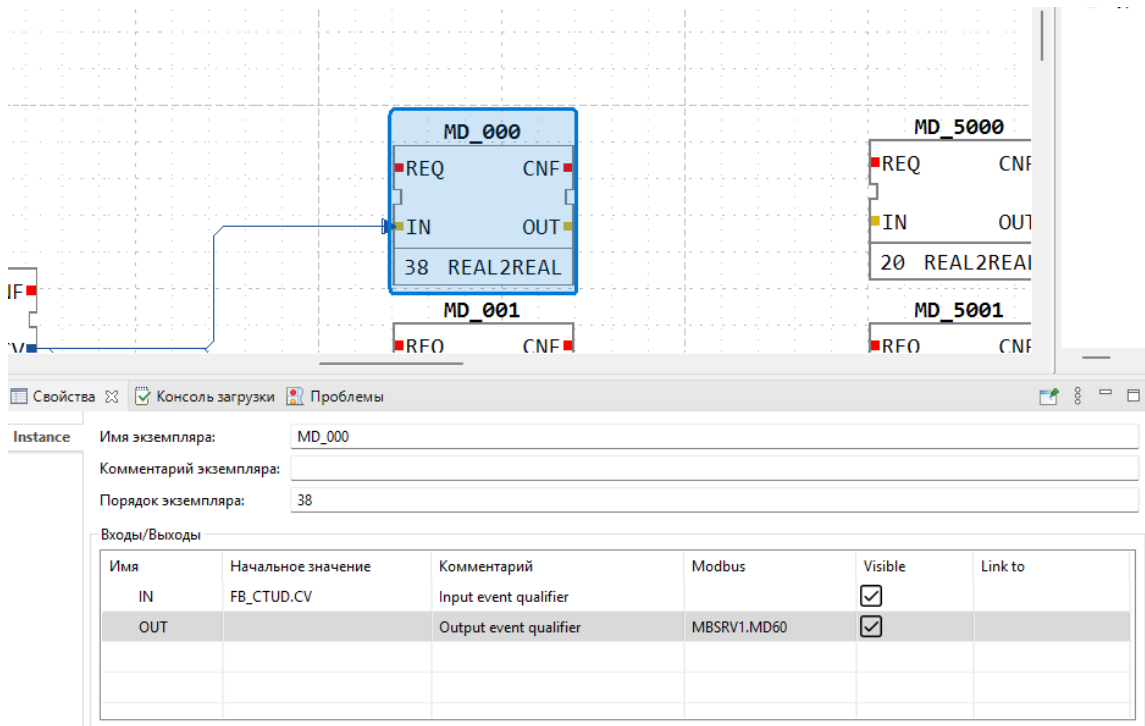


Рисунок 102 - Привязка параметра к регистру Modbus TCP сервера

10.1.1. Правила чтения/записи регистров входных параметров

Следующие правила определяют направление записи:

Таблица 5 – Правила чтения/записи регистров входных параметров

| Событие | Чтение из регистра Modbus TCP сервера | Запись в регистр Modbus TCP сервера |
|---|---|---|
| Старт функционального блока | | Функциональный блок записывает своё начальное значение в регистр Modbus |
| Текущее выполнение функционального блока | Функциональный блок каждое своё выполнение считывает значение регистра Modbus во входной параметр | |
| Изменение входного параметра пользователем в режиме мониторинга | | Одномоментно происходит запись в регистр Modbus |
| Форсирование переменной | | Постоянно с периодом обработки блока происходит запись в регистр Modbus |

Установка параметра Modbus TCP для входного параметра и создание соединения с этим же параметром от другого блока несовместимы

10.2. Modbus TCP Client

Для опроса внешних устройств по протоколу Modbus TCP в контроллере используются следующие блоки: для записи данных MBWRITE и для чтения данных MBREAD (см. [Рисунок 103](#)).

Блоки MBWRITE и MBREAD работают только в событийной задаче.

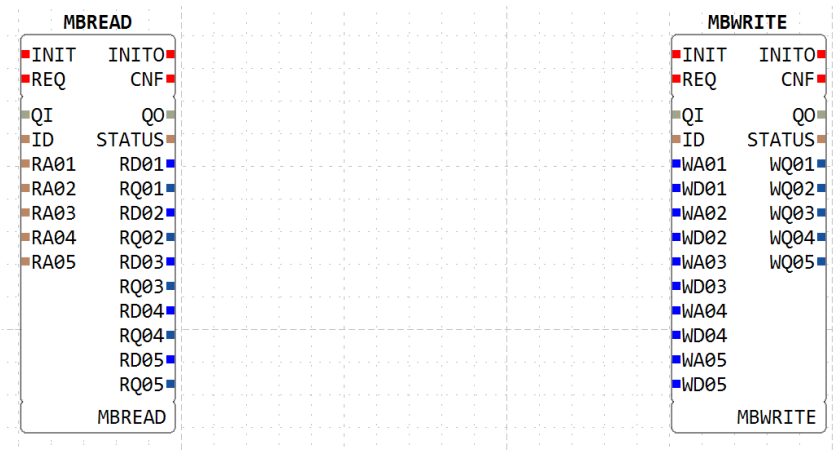


Рисунок 103 – ФБ – MBREAD (чтение) и MB WRITE (запись)

10.2.1. Видимость параметров данных

Блоки MBWRITE и MBWRITE по умолчанию отображает не все входы/выходы данных. Чтобы отобразить необходимое количество входов/выходов, нужно выполнить команду из контекстного меню функционального блока «Видимость».

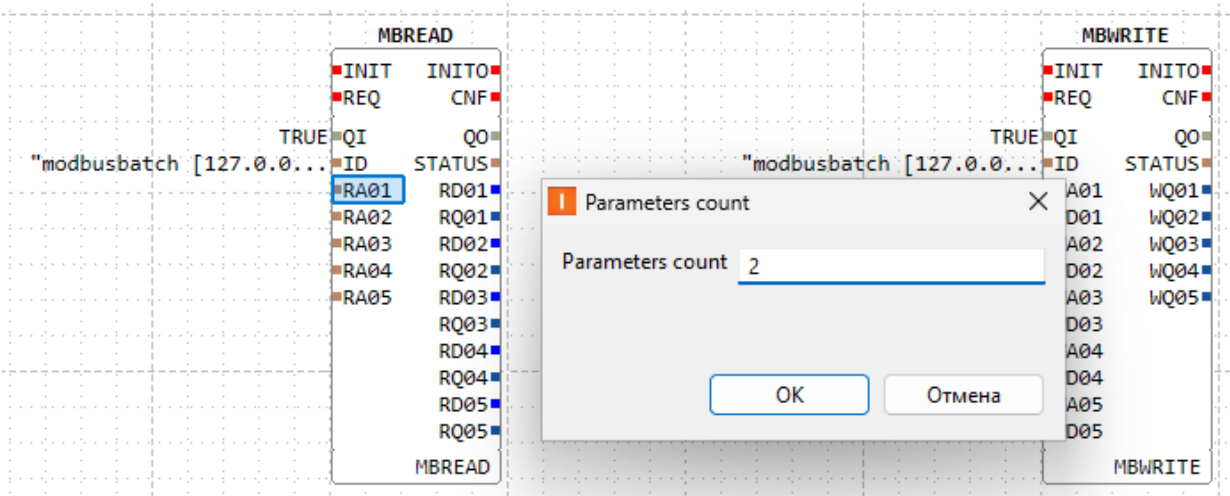


Рисунок 104 – Количество параметров

10.2.2. Инициализация типа ANY параметров

Тип входных и выходных данных блоков MBREAD/MBWRITE является типом ANY. Для корректного типы данных должны быть инициализированы.

Инициализация типов возможна двумя способами:

1. Создание соединения с параметром определённого типа (см. [Рисунок 105](#))

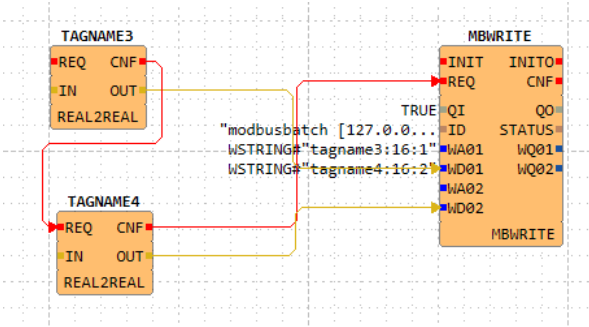


Рисунок 105 – Инициализация входа WD01 в REAL

2. Инициализация типа данных через публикацию данных на внутренний Modbus Server (см. [Рисунок 106](#))

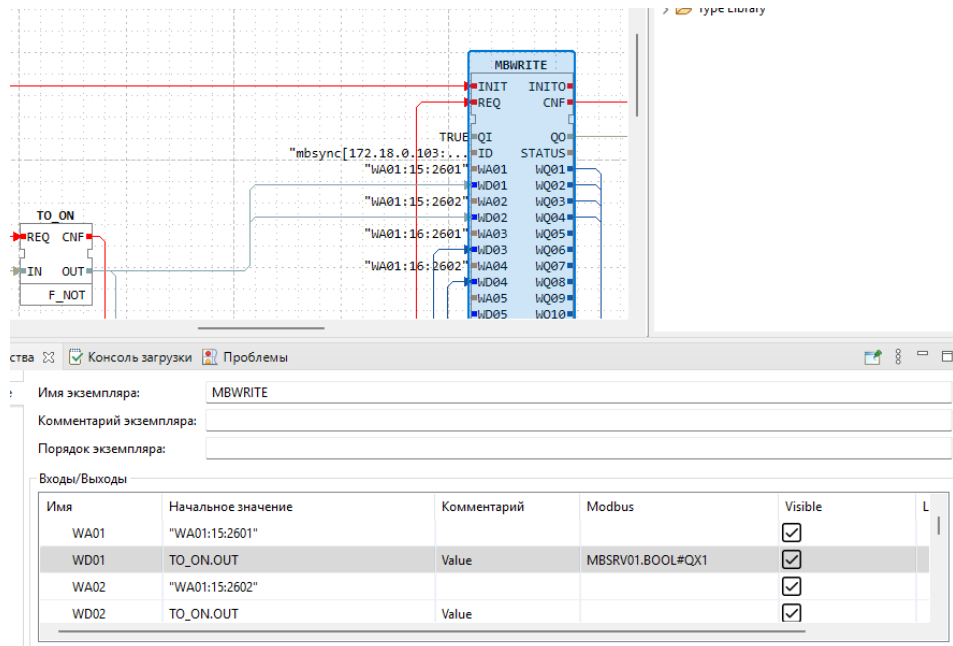


Рисунок 106 – Инициализация входа WD01 в BOOL

Ниже в таблице представлено соответствие поддерживаемых типов данных и функций Modbus:

| Тип регистра | Функция записи | Функция чтения | Тип параметра | Размер |
|---------------------------------|----------------|----------------|---------------|---------|
| Discrete Output Coils | 05, 15 | 01 | BOOL | 1 bit |
| Input Discrete Input Contacts | - | 02 | BOOL | 1 bit |
| Analog Output Holding Registers | 06, 16 | 03 | INT | 32 bits |
| Analog Input Registers | - | 04 | INT | 32 bits |

10.2.3. Блок MBWRITE

Для записи данных по протоколу Modbus TCP используется блок MBWRITE. Ниже в таблице представлены входные и выходные параметры блока:

| Параметр | Описание | Пример значения |
|-----------|---|--------------------------------|
| INIT | Событийный вход, который используется для инициализации: если QI=TRUE, то происходит считывание адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | - |
| REQ | Событийный вход, который инициализирует отправку пакета для записи на устройство | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | IP:PORT:ID:OFFSET:TIMEOUT IP – ip адрес устройства PORT – сетевой порт устройства ID – Modbus адрес устройства OFFSET – начальный адрес 0 или 1 TIMEOUT – таймаут ответа от устройства | mbsync[127.0.0.1:502:1:0:2500] |
| WD01-WD99 | Адрес регистра для записи: TAGNAME:FUNCTION:ADDRESS TAGNAME – строка имени тэга (для буд. исп.) FUNCTION – функция модбас записи: 05: Single Coils 06: Single Discrete Inputs 15: Multiple Coils 16: Multiple Holding Registers ADDRESS – адрес регистра для записи | "WA01:16:2602" |
| WQ01-WQ99 | Write Quality – качество записанных данных 1 – хорошее значение, данные записаны «<>1» – ошибка записи | - |

Ниже представлен пример логики для записи данных (см. [Рисунок 107](#)).

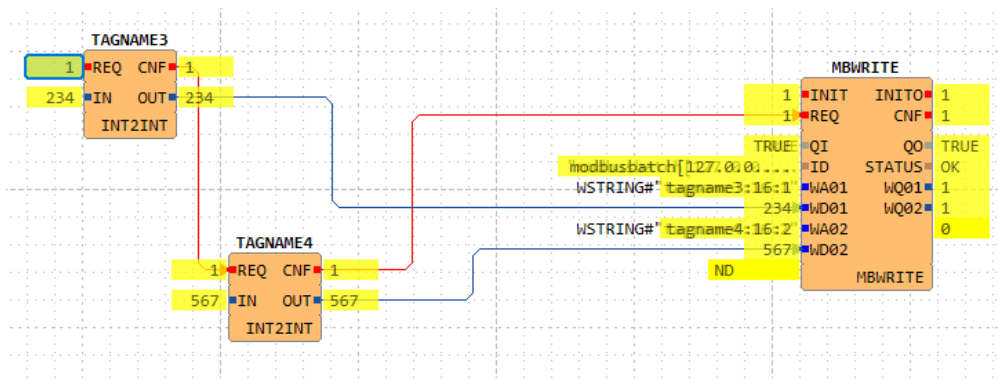


Рисунок 107 – Запись по протоколу Modbus TCP

10.2.4. Блок MBREAD

Для записи данных по протоколу Modbus TCP используется блок MBREAD. Ниже в таблице представлены входные и выходные параметры блока:

| Параметр | Описание | Пример значения |
|-----------|---|--------------------------------|
| INIT | Событийный вход, который используется для инициализации: если QI=TRUE, то происходит считывание адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | - |
| REQ | - | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | IP:PORT:ID:OFFSET:TIMEOUT IP – IP адрес устройства PORT – сетевой порт устройства ID – Modbus адрес устройства OFFSET – начальный адрес 0 или 1 TIMEOUT – таймаут ответа от устройства | mbsync[127.0.0.1:502:1:0:2500] |
| RA01-RA99 | Адрес регистра для записи: TAGNAME:SCANTIME:FUNCTION:ADDRESS TAGNAME – строка имени тэга (для буд. исп.) SCANTIME – частота опроса (мс) FUNCTION – функция модбас записи: 01: Coils 02: Discrete Inputs | "RA01:200:01:100" |

| Параметр | Описание | Пример значения |
|-----------|--|-----------------|
| | 03: Multiple Holding Registers 04: Input Registers ADDRESS – адрес регистра для записи | |
| RQ01-RQ99 | Read Quality – качество записанных данных 1 – хорошее значение, данные прочитаны «<>1» – ошибка чтения | - |

Ниже представлен пример логики для записи данных (см. [Рисунок 108](#)).

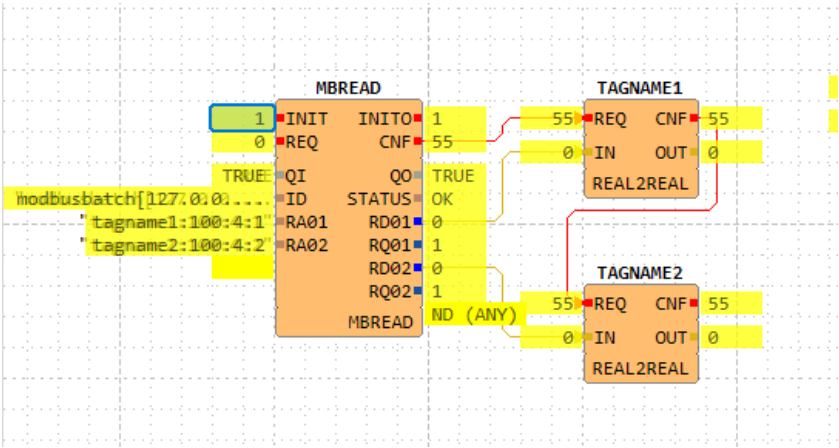


Рисунок 108 – Чтение по протоколу Modbus TCP

10.3. OPC UA Client

Для чтения и записи данных во внешний OPC UA Сервер используется блок CLIENT и блок SUBSCRIBE (см. 85).

Блоки CLIENT и SUBSCRIBE являются блоками с изменяемым количеством входов/выходов. Для изменения количества входов или выходов см. раздел [6.6.7 Блоки с изменяемым количеством входов/выходов](#).

Блок Client должен содержать только входы или только выходы.

10.3.1. Блок CLIENT

Таблица 6 Параметры блока CLIENT

| Параметр | Описание | Пример значения |
|----------|---|-----------------|
| INIT | Событийный вход, который используется для инициализации: если QI=TRUE, то происходит считывание | - |

| Параметр | Описание | Пример значения |
|----------|---|---|
| | адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | |
| REQ | Запрос на запись/чтение данных в OPC UA | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | описание сервера и точек в пространстве адресов OPC UA: "opc_ua[COMMAND; opc.tcp://localhost:4840#; TAGNAME_SD_1,Node:i=ADDRESS_SD_1; TAGNAME_SD_2,Node:i=ADDRESS_SD_2;]" COMMAND – WRITE/READ TAGNAME_SD_1 – путь и имя точки 1 Node – пространство адресов ADDRESS_SD_1 – глобальное имя точки, синтаксис s=TAG1 или i=123456 | "opc_ua[WRITE; opc.tcp://localhost:4840#; /Objects/TAGS/TAG1,0:s=TAG1; /Objects/TAGS/TAG2,0:s=TAG2]" |
| SD_1... | Вход для передаваемых данных, должен быть обязательно подключен к блоку определённого типа | - |
| RD_1... | Выход для считываемых данных, должен быть обязательно подключен к блоку определённого типа | - |
| QO | Качество записи, 1 – "ок" | - |
| STATUS | Статус работы блока | - |

Схема для чтения данных с удаленного сервера представлена на рисунке (см. [Рисунок 109](#)).

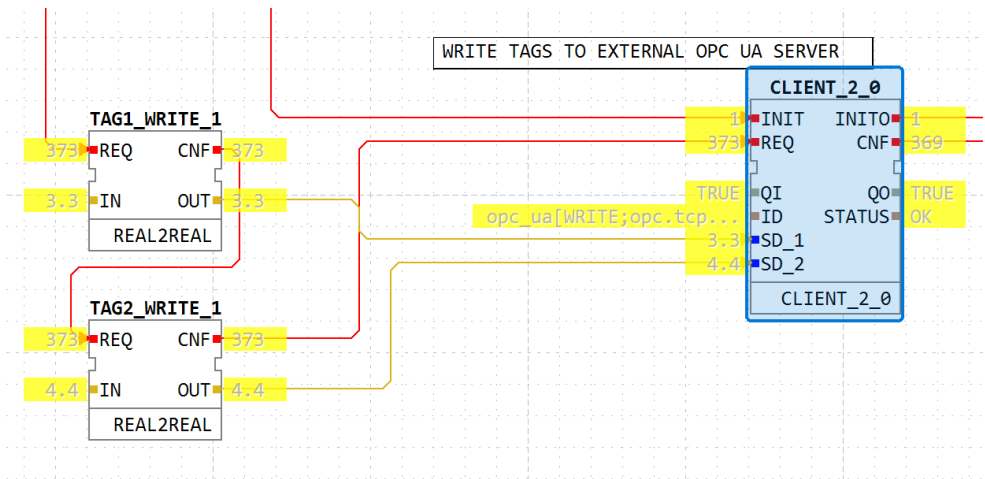


Рисунок 109 – Чтение данных через блок CLIENT

Схема для записи данных на удаленный сервер представлена на рисунке (см. [Рисунок 110](#)).

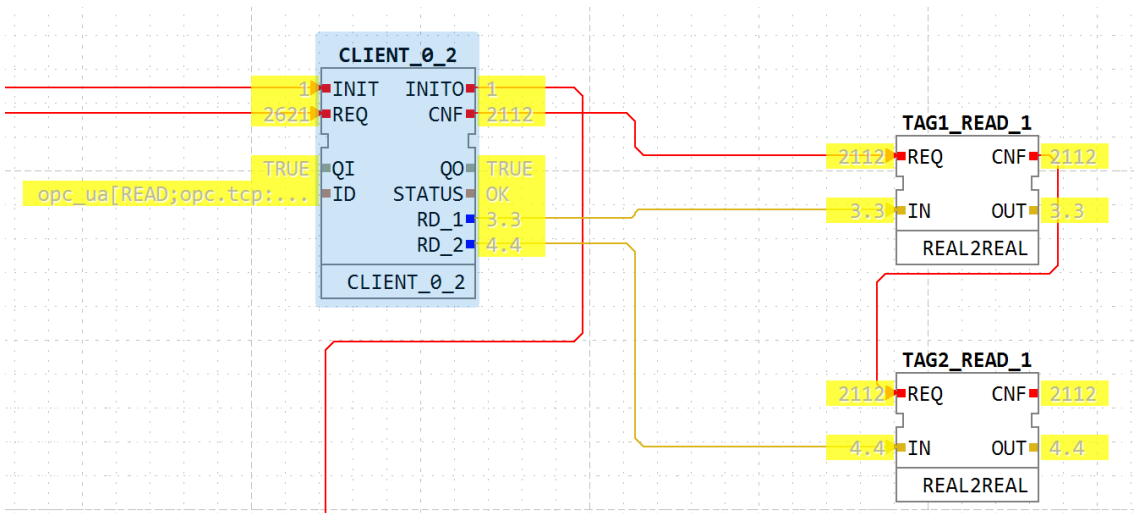


Рисунок 110 – Запись данных через блок CLIENT

10.3.2. Блок SUBSCRIBE

В отличие от блока CLIENT блок SUBSCRIBE получает данные только при их обновлении на сервере, в то время как, блок CLIENT запрашивает данные по событию со стороны контроллера.

Таблица 7 Параметры блока SUBSCRIBE

| Параметр | Описание | Пример значения |
|----------|--------------------------|-----------------|
| INIT | Событийный вход, который | - |

| Параметр | Описание | Пример значения |
|----------|---|---|
| | используется для инициализации: если QI=TRUE, то происходит считывание адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | |
| RSP | Не используется | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | описание сервера и точек в пространстве адресов OPC UA: "opc_ua[SUBSCRIBE; opc.tcp://localhost:4840#; TAGNAME_SD_1,Node:i=ADDRESS_SD_1; TAGNAME_SD_2,Node:i=ADDRESS_SD_2;]" TAGNAME_SD_1 – путь и имя точки 1 Node – пространство адресов ADDRESS_SD_1 – глобальное имя точки, синтаксис s=TAG1 или i=123456 | "opc_ua[SUBSCRIBE; opc.tcp://localhost:4840#; /Objects/TAGS/TAG1,0:s=TAG1; /Objects/TAGS/TAG2,0:s=TAG2]" |
| RD_1... | Выход для считываемых данных, должен быть обязательно подключен к блоку определённого типа | - |
| QO | Качество записи, 1 – "ок" | - |
| STATUS | Статус работы блока | - |

Схема для чтения данных по подписке с удаленного сервера представлена на рисунке (см. [Рисунок 111](#)).

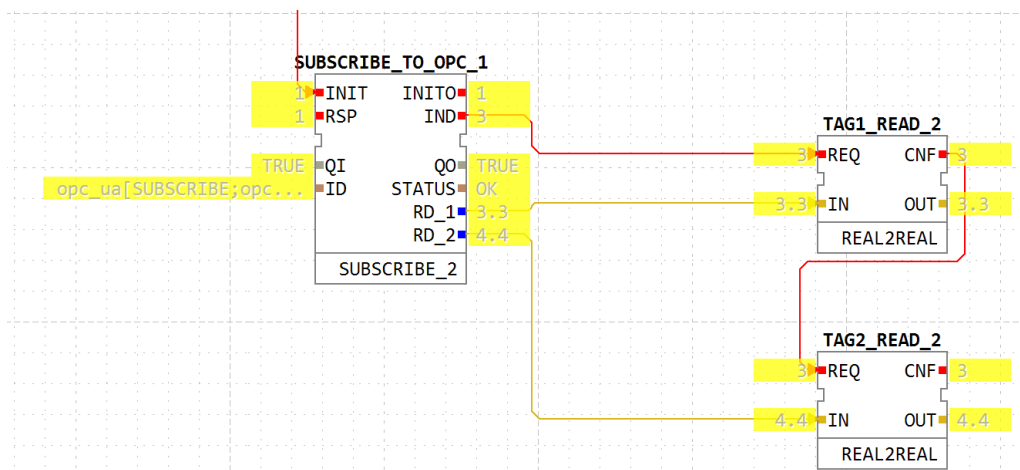


Рисунок 111 – Чтение данных через блок SUBSCRIBE

10.4. OPC UA Server

Для предоставления данных внешним устройствам по протоколу OPC UA в контроллере используются следующие блоки: для записи данных PUBLISH и для чтения данных SUBSCRIBE (см. 85). Блоки работают только в событийных задачах.

10.4.1. Блок PUBLISH

Блок используется для записи данных по протоколу OPC UA. Входные параметры блока имеют тип ANY. Для инициализации типа данных они должны быть обязательно подключены к параметрам определённого типа данных (см. [Рисунок 112](#)).

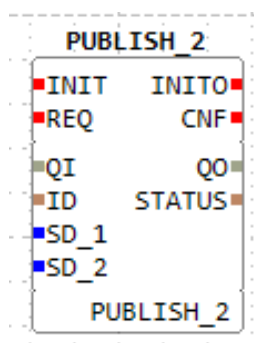


Рисунок 112 – ФБ PUBLISH

Таблица 8 Параметры блока PUBLISH

| Параметр | Описание | Пример значения |
|----------|--|--|
| INIT | Событийный вход, который используется для инициализации: если QI=TRUE, то происходит считывание адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | - |
| REQ | Запрос на запись данных в OPC UA | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | описание точек в пространстве адресов OPC UA «opc_ua[write;/objects/ TAGNAME_SD_1,Node:s=ADDRESS_SD_1; /objects/ TAGNAME_SD_2,Node:i=ADDRESS_SD_2;]» TAGNAME_SD_1 – путь и имя точки Node – пространство адресов | «opc_ua [write; /objects/QT01,0:s=QT01; /objects/QT02,0:i=123456]» |

| | | |
|---------|---|---|
| | ADDRESS_SD_1 – глобальное имя точки, синтаксис s=TAG1 или i=123456 | |
| SD_1... | Вход для данных, должен быть обязательно подключен к блоку определённого типа | - |
| QO | Качество записи, 1 – “ок” | - |
| STATUS | Статус работы блока | - |

Полностью сконфигурированный и работающий блок представлен на рисунке (см. [Рисунок 113](#)).

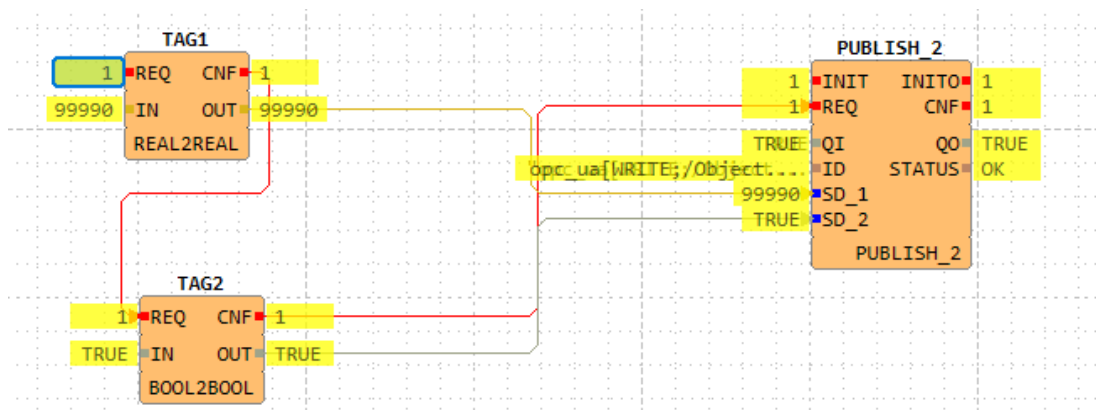


Рисунок 113 – Нажимаем правой кнопкой «мыши» на REQ и выбираем Trigger Event

10.4.2. Блок SUBSCRIBE

Блок используется для чтения данных по протоколу OPC UA. Выходные параметры блока имеют тип ANY. Для инициализации типа данных они должны быть обязательно подключены к параметрам определённого типа данных (см. [Рисунок 114](#)).

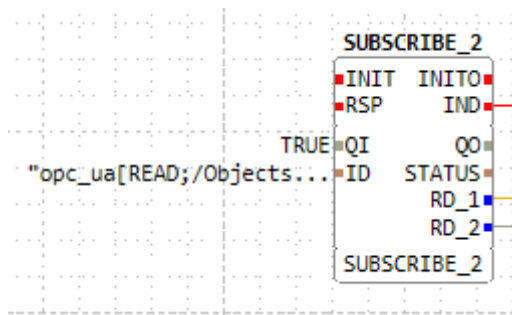


Рисунок 114 – ФБ SUBSCRIBE

Таблица 9 Параметры блока SUBSCRIBE

| Параметр | Описание | Пример значения |
|----------|--|--|
| INIT | Событийный вход, который используется для инициализации: если QI=TRUE, то происходит считывание адреса устройства (ID) и выполняется подключение к устройству, если QI=FALSE, то – обрыв подключения | - |
| RSP | Не используется | - |
| QI | Управляющий вход: подключение (TRUE) или отключение (FALSE) | TRUE/FALSE |
| ID | описание точек в пространстве адресов OPC UA «opc_ua[write;/objects/ TAGNAME_SD_1,Node:s=ADDRESS_SD_1; /objects/ TAGNAME_SD_2,Node:i=ADDRESS_SD_2;]» TAGNAME_SD_1 – путь и имя точки Node – пространство адресов ADDRESS_SD_1 – глобальное имя точки, синтаксис s=TAG1 или i=123456 | «opc_ua [write; /objects/QT01,0:s=QT01; /objects/QT02,0:i=123456]» |
| RD_1... | Вход для данных, должен быть обязательно подключен к блоку определённого типа | - |
| QO | Качество чтения, 1 – “ок” | - |
| STATUS | Статус работы блока | - |

Полностью сконфигурированный и работающий блок представлен на рисунке (см. [Рисунок 115](#)).

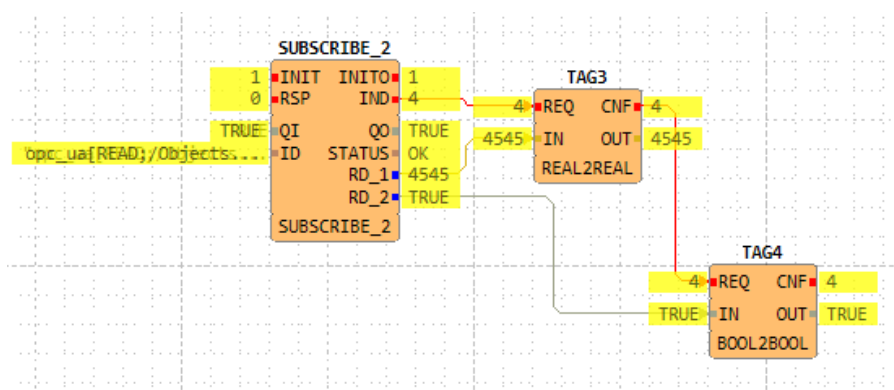


Рисунок 115 – Блок Subscribe для чтения данных по протоколу OPC UA